Collision avoidance based on separating planes for feet trajectory generation

Stanislas Brossette, Pierre-Brice Wieber

Collision avoidance based on separating planes for feet trajectory generation

Stanislas Brossette[1], Pierre-Brice Wieber[1]

Abstract— **In this paper, we present a formulation of the collision avoidance constraints that relies on the use of separating planes instead of a distance function. This formulation has the advantage of being defined and continuously differentiable in every situation. Because it introduces additional variables to the optimization problems, making it bigger and potentially slowing down its resolution, we propose a different resolution method that takes advantage of the independence of the variables, to form two subproblems that can be solved efficiently in an alternate problem fashion. We present some preliminary results using this approach in order to highlight its potential and promises in terms of convergence speed and robustness.**

I. Introduction

When a legged robot takes a step, it is crucial to ensure that the trajectory of the leg, and in particular the trajectory of the feet, avoids unwanted collisions with the environment. In [1] the collision-free feet trajectory of a four-legged robot is generated in the vertical plane containing the start and goal positions while staying away from the convex hull of the obstacles with a safety distance. That trajectory is later refined in order to avoid collisions with other parts of the legs, like the knee or shin. In humanoid robotics, most tasks require to consider collision avoidance, for walking tasks, the trajectory of the feet must avoid obstacles [2], [3], and for other tasks, such as ones performed in the context of multi-contact locomotion, collisions need to be avoided between all the bodies of the robot that are not contact bearing and the environment. This impacts the formulations of problems of posture generation [4] as well as motion planning and control [5]. Collision avoidance can also be used with virtual obstacles, for example when the location of a potentially mobile obstacle is known with an uncertainty [6].

In general, collisions are avoided by ensuring that the distance between bodies is always positive or greater than a safety distance. Schulman et al. [7] presented in great details how to ensure collision avoidance in the context of motion planning and showcases applications on various scenarios with different robots. The distance between two objects is equal to the distance between their two closest points and finding those points is the tricky part. For general convex shaped objects, computing the location of the two closest points is most commonly done by using the Gilbert-Johnson-Keerthi (GJK) algorithm presented in [8]. The distance function for a pair of objects is based on the results of that algorithm, and a particularly important property of this function is that it is not continuously differentiable unless one of the objects is convex, and the other is strictly convex (proofs can be found in [9] and [10]). Non-differentiability of the distance function can lead to convergence issues when using smooth optimization(see [11]). This issue is tackled in [12], where a method is proposed to generate (offline) a strictly convex approximation of the shape of an object with a bounding volume composed of patches of spheres and tori called STP-BV. That method requires the user to define a minimum radius of the STP-BV, which will impact the 'stiffness' of the evolution of the gradient of the distance functions.

Conversely to computing the distance between objects, it is often useful to compute the penetration depth between two colliding objects. Like during an optimization process, when at some iterations some objects are colliding and that quantity is useful to separate them. Or when solving scenarios involving virtual obstacles, with which a limited interpenetration can be tolerated. The abovementioned method for computing the distance between two object does not extend to situations where the objects are in collision and one needs to compute the interpenetration depth, which is the minimum translation needed to separate the objects. That quantity is usually computed using the Expanding Polytope Algorithm (EPA) [13], which is not continuously differentiable. Both the distance and penetration functions involved in the constraint are non-differentiable [7], and so is their sum. One must be especially careful when handling the transition point where the two objects are in contact without interpenetration.

Overall, the use of a distance function between objects to ensure collision avoidance is not particularly convenient. In this paper, we propose a different approach based on the idea that, in vertue of the *'Hyperplane separation theorem'*[14], if two convex objects are not colliding, there exists a plane separating them. Similar approaches are often used to detect interpenetration in computer graphics and to separate sets with Support Vector Machines[15], but to our knowledge have never been used for the purpose of generating collision-free trajectories in robotics. For each pair of objects, we search for a separating plane between them. The constraint of positive distance between two objects becomes a set of constraints requiring all the vertices of the objects to be above or below a separating plane. By relaxing this constraint, we show that it is possible to use this approach even when the objects are in interpenetration. Such a formulation has the advantage of being defined and differentiable in

[1]Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP*, LJK, 38000 Grenoble, France

This work has been funded by the PSPC Romeo 2 project and EU H2020 Comanoid Research and Innovation Action (RIA).

The authors thank Adrien Escande for his advice and fruitful conversations on the topic of this contribution.

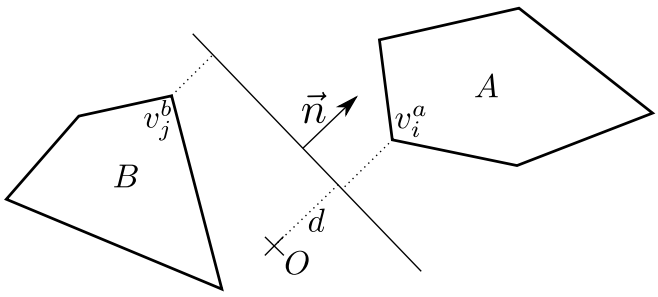

Fig. 1. Illustration of the collision avoidance constraint between bodies A and B separated by plane $P = \{\vec{n}, d\}$

all cases(separated and interpenetrating objects), and being simpler than the usual method, using only one type of constraint and algorithm instead of two. Using this formulation, collision-free trajectory generation problems can be solved by a state-of-the-art nonlinear optimization solver, which we illustrate on some academic examples.

We aim at using this method to generate safe foot trajectory for humanoid robots. We present an alternative solving approach taking advantage of the structure of our collision avoidance constraint, in which, instead of solving one nonlinear problem, we decompose the resolution into two simpler problems. A nonlinear problem in the variables associated with the locations of the objects, that can become a quadratic problem(QP) is common cases, and a linear problem(LP) in the variables associated with the locations of the separating planes, those problems are then solved alternatively, which drastically reduces the computation times.

II. Collision avoidance constraint with separating planes

A. Collision avoidance constraint

Given two bodies A and B, which convex hulls are respectively composed of the sets of vertices $\mathcal{V}_A = \{v_0^a, \cdots, v_{m_a}^a\}$ and $\mathcal{V}_B = \{v_0^b, \cdots, v_{m_b}^b\}$. We define a plane P by its normal n and its orthogonal distance d to the origin of the world O. We illustrate these notations is figure Fig. 1. A plane $P = \{\vec{n}, d\}$ separating A and B must satisfy the following constraints:

$$\begin{cases} \forall v \in \mathcal{V}_A,\ v \cdot n \geq d \\ \forall v \in \mathcal{V}_B,\ v \cdot n \leq d \\ \|n\| = 1 \end{cases} \tag{1}$$

Solving problem (1) allows to ensure that the plane P is between A and B if they are separated. When they are in interpenetration, the problem (1) is unfeasible. To deal with that infeasibility and ensure that the problem is always feasible, we can considere using an elastic mode by adding a relaxation term r to our constraints, which gives the following problem:

$$\min_{n,d,r} \|r\| \quad \text{s.t.} \begin{cases} \forall v \in \mathcal{V}_A,\ 0 \leq v \cdot n - d + r \\ \forall v \in \mathcal{V}_B,\ 0 \leq -v \cdot n + d + r \\ \|n\| = 1 \end{cases} \tag{2}$$

When the problem (1) is feasible, r is null, otherwise, r's minimization will drive the plane to minimize the violation of both constraints of problem (1) simultaneously. The solution is then the plane $P : \{\vec{n}, d\}$ such that a translation of A by $r\vec{n}$ and of B by $-r\vec{n}$ makes the two objects separated by P. In other words, it gives the minimum penetration depth $2r$ and the direction of the penetration is given by $\vec{n}$.

When imposing collision avoidance constraints, it is usual to require a safety distance 2β between the objects, our formulation straightforwardly extends to account for that:

$$0 \leq v \cdot n - d + r \text{ becomes } \beta \leq v \cdot n - d + r \tag{3}$$

B. Collision avoidance constraint in continuous-time trajectory

The constraint presented above allows to ensure the separation of two objects instantaneously. However, when generating a collision-free trajectory, ensuring that collisions are avoided instantaneously at every time step does not guarantee that there is no collision inbetween time steps.

In order to ensure that a collision-free trajectory exists between successive body positions $B(t_k)$ and $B(t_{k+1})$, we require that a plane P_{jk} exists that separates each obstacle O_j (defined by its position o_j and set of vertices $\mathcal{W}_j$) from the convex hull of the two successive bodies, which approximates the volume swept by B during the time interval $[t_k, t_{k+1}]$, as illustrated in figure Fig. 2. Like in [7], we ignore the effect of rotations of B on the swept volume as their influence in minor and can be encompassed conservatively by the use of a safety distance. This is equivalent to finding a plane that separates $B(t_k)$ from O_j and $B(t_{k+1})$ from O_j simultaneously. r_{jk} denotes the relaxation term associated with the collision avoidance constraint with O_j during time interval $[t_k, t_{k+1}]$. We denote x the set of problem's variables, such that $x(t_k) = x_k$ and the locations of a vertex v of B at time t is $v(x(t))$

The continuous-time collision avoidance problem is:

$$\min. \sum_{j,k} \|r_{jk}\| \quad \begin{cases} \forall v \in \mathcal{V}_B,\ \beta \leq v(x_k) \cdot n_{jk} - d_{jk} + r_{jk} \\ \forall v \in \mathcal{V}_B,\ \beta \leq v(x_{k+1}) \cdot n_{jk} - d_{jk} + r_{jk} \\ \forall w \in \mathcal{W}_j,\ \beta \leq -(o_j + w) \cdot n_{jk} + d_{jk} + r_{jk} \\ \|n_{jk}\| = 1 \end{cases} \tag{4}$$

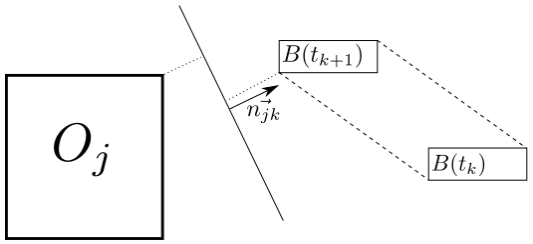

Fig. 2. Illustration of the collision avoidance constraint between successive bodies B_{j1} and B_{j2} and obstacle O_{j0}

C. Complete nonlinear problem

By adding a cost function $f(x)$ and concatenating the continuous-time collision avoidance constraints on all time interval, we can formulate a problem on the whole trajectory of a body B as an optimization problem:

$$\min_{x,d,n} f(x) + \alpha_R \|r\|$$
$$\text{s.t. } \forall j,k \begin{cases} \forall v \in \mathcal{V},\ 0 \leq v(x_k) \cdot n_{jk} - d_{jk} + r_{jk} \\ \forall v \in \mathcal{V},\ 0 \leq v(x_{k+1}) \cdot n_{jk} - d_{jk} + r_{jk} \\ \forall w \in \mathcal{W}_j,\ 0 \leq -(o_j + w) \cdot n_{jk} + d_{jk} + r_{jk} \\ \|n_{jk}\| = 1 \end{cases} \tag{5}$$

Due to the relaxation of the collision avoidance constraints, if a collision-free trajectory is not found, the result is a trajectory that minimizes penetrations. The choice of norm used on r governs the way the amount of penetration is minimized. With an l1-norm, the sum of all penetrations is minimized, while with an infinity-norm $\|.\|_\infty$, only the maximum penetration is minimized, disregarding all smaller penetrations. Using the l1-norm $\|.\|_1$ requires the addition of as many variables r_{jk} as there are pairs $\{j,k\}$. Using the infinity-norm $\|.\|_\infty$ a single variable r is sufficient. A careful choice of the coefficient α_R allows to penalize exactly the violation of the non-collision avoidance constraint.

III. Resolution

A. Resolution with NLP

In its generic form, the nonlinear problem (5) can be solved with a state-of-the-art nonlinear solver. To be rid of the $\|n_j\| = 1$ constraints, we used PGSolver [4], a nonlinear solver on manifolds and search for a solution on a manifold where the variables representing the normals n are elements of S^2 and thus always have unit-norm.

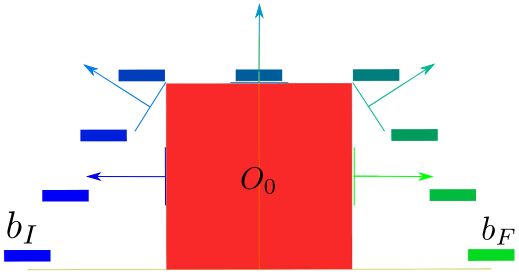

Fig. 3. Result obtained by resolution of the nonlinear problem with PGSolver

In Fig. 3, we present a result obtained by solving (5), where the cost function f approximates the total distance traveled by the body, with $b_k(x)$ denoting the 3D position of B at time t_k:

$$f(x) = \sum_k \|b_{k+1}(x) - b_k(x)\|^2 \tag{6}$$

In this example, the trajectory is discretized in 8 time-steps for a body B going from position b_I (blue) to position b_F (green) while avoiding collisions with a cubic obstacle O_0(red). One can observe how the separating planes are located: When possible, the planes coincide with the faces of the obstacle, some planes are tilted to allow reducing the distance between successive objects while maintaining collision avoidance.

Solving problem (5) as is with a nonlinear solver is convenient, but it is fairly slow (see section IV). This is most likely due to the fact that it works with the entire set of variables: bodies and planes locations and all the nonlinear constraints. In the next section, we propose a different resolution method in which the variables are separated.

B. Resolution with alternate problems

We consider simplifying this problem by formulating two separate subproblems in which the variables related to the bodies $\{x\}$ and the planes $\{n,d\}$ are alternately fixed, and solving those two problems iteratively until a solution is found.

We denote PBP(n,d) the problem with fixed planes, and PBB(x) the problem with fixed bodies. By construction, PBP(n,d) is a linear program that can be efficiently solved by an LP algorithm. PBB(x) is a nonlinear problem, that can be solved by a nonlinear solver. In some cases, like for foot trajectory planning, it can be formulated as a quadratic problem with linear constraints, thus making it efficiently solvable with a QP algorithm.

We propose to use a method of alternate problems to solve (5). This consists in computing PBP(n,d) for a given set of planes locations, solving it to obtain a set of bodies locations that will be used to compute PBB(x). Solving PBB(x) gives new values of planes locations to compute the next iteration of PBP(n,d). And repeating this process until convergence, as described in algorithm 1.

Algorithm 1 Alternate problems resolution

```
Initialize n and b
while not converged do
    compute PBB(x)
    {n, d} ←solve PBB(x)
    compute PBP(n, d)
    {x} ←solve PBP(n, d)
end while
```

1) Problem with fixed bodies: When bodies variables are fixed, problem (5) can be formulated as a linear program. All the constraints remain expressed in the same way, except for the norm-1 constraint on the normal's variables n_j. Enforcing the norm of n_j to be 1 cannot be done exactly with linear constraints. But in fact, the amplitude of the vector n_j does not matter, as long as it is not null. Only the direction of n_j and the product $d_j n_j$ describe perfectly the plane P_j. Indeed, a plane described by $\{n_j, d_j\}$ is equivalent to one described by $\{\alpha n_j, \frac{d_j}{\alpha}\}$ for any nonnull α.

To ensure that n_j is nonnull and maintains a norm of the magnitude of 1, we bound the scalar product between its current value n_j(which is a problem's variable) and its value obtained as result of the previous iteration denoted

n_j^{prev}(which is a constant). The lower bound $n_{\min}$ needs to be strictly positive and the upper bound can be 1. Additionally, to prevent divergence of the components of n_j, they can all be bounded between -1 and 1.

$$n_{\min} \leq n_j^{\text{prev}} \cdot n_j \leq 1 \tag{7}$$

We obtain the following relaxed linear problem:

$$\begin{aligned} &\min_{n,d,r} \|r\| \\ &\text{s.t. } \forall j,k \begin{cases} \forall v \in \mathcal{V} : 0 \leq v(x_k) \cdot n_{jk} - d_{jk} + r_{jk} \\ \forall v \in \mathcal{V} : 0 \leq v(x_{k+1}) \cdot n_{jk} - d_{jk} + r_{jk} \\ \forall w \in \mathcal{W}_j : 0 \leq -w \cdot n_{jk} + d_{jk} + r_{jk} \\ n_{\min} \leq n_{jk}^{prev} \cdot n_{jk} \leq 1 \end{cases} \end{aligned} \tag{8}$$

Notice that the problems for each $\{j,k\}$ are only coupled by the relaxation variable. We can separate them into M independent LP problems of smaller sizes.

$$\forall j,k \begin{cases} \min\limits_{n_{jk},d_{jk},r_{jk}} r_{jk} \\ \text{s.t. } \begin{cases} \forall v \in \mathcal{V} : 0 \leq v(x_k) \cdot n_{jk} - d_{jk} + r_{jk} \\ \forall v \in \mathcal{V} : 0 \leq v(x_{k+1}) \cdot n_{jk} - d_{jk} + r_{jk} \\ \forall w \in \mathcal{W}_j : 0 \leq -w \cdot n_{jk} + d_{jk} + r_{jk} \\ n_{\min} \leq n_{jk}^{prev} \cdot n_{jk} \leq 1 \end{cases} \end{cases} \tag{9}$$

When r_{jk} is positive, it is a measure of interpenetration, whereas when it is negative, it is a measure of the minimum distance between the plane and the two objects it separates. Therefore, by removing the norm on r_{jk} in the cost function, the separating plane is driven to maximize that distance and thus placing the plane at equidistance from both objects it separates.

In order to avoid numerical issues, the solution plane is replaced by an equivalent plane with unit-norm normal before computing the next problem:

$$\{n_{jk}, d_{jk}\} \leftarrow \left\{ \frac{n_{jk}}{\|n_{jk}\|}, d_{jk}\|n_{jk}\| \right\} \tag{10}$$

This prevents the norms of the n_{jk} vectors to gradually converge towards zero.

Then the value of d_{jk} can be changed as in (11) so that the planes are located as close as possible to the obstacle, giving as much freedom as possible for the choice of location of the bodies in PBP(n,d).

$$d_{jk} = \max_{w \in \mathcal{W}_j} ((w + o_j) \cdot n_{jk}) \tag{11}$$

2) Problem with fixed planes: When the planes are fixed, the third set of constraint of problem (5) can be reduced to a single constraint as follows:

$$\begin{gathered} \forall w \in \mathcal{W}_j,\ 0 \leq -(o_j + w) \cdot n_{jk} + d_{jk} + r_{jk} \\ \text{becomes} \\ \max_{w \in \mathcal{W}_j} (w \cdot n_{jk}) \leq -n_{jk} \cdot o_j + d_{jk} \end{gathered} \tag{12}$$

The problem to solve becomes:

$$\begin{aligned} &\min_{x,r} f(x) + \alpha_R \|r\| \\ &\text{s.t.} \\ &\forall j,k \begin{cases} \forall v \in \mathcal{V},\ 0 \leq v(x_k) \cdot n_{jk} - d_{jk} + r_{jk} \\ \forall v \in \mathcal{V},\ 0 \leq v(x_{k+1}) \cdot n_{jk} - d_{jk} + r_{jk} \\ \max\limits_{w \in \mathcal{W}_j} (w \cdot n_{jk}) \leq -n_{jk} \cdot o_j + d_{jk} + r_{jk} \end{cases} \end{aligned} \tag{13}$$

This problem can be solved with a state-of-the-art nonlinear solver.

In the event where f is quadratic and the positions of the vertices are linear in the variables x_k, the problem (13) becomes a quadratic program, which can be solved efficiently with a QP solver, and more simplifications of the problem can be made. For example, when the location of the vertices can be defined only with the position b_k of the body at time t_k:

$$\forall v \in \mathcal{V} : v(x_k) = b_k + v \tag{14}$$

Then a modification similar to (12) can be applied to the first two sets of constraints of (13), giving:

$$\begin{aligned} &\min_{x,r} f(x) + \alpha_R \|r\| \\ &\text{s.t.} \\ &\forall j,k \begin{cases} -\min\limits_{v \in \mathcal{V}_k} (v \cdot n_{jk}) \leq n_{jk} \cdot b_k - d_{jk} + r_{jk} \\ -\min\limits_{v \in \mathcal{V}_{k+1}} (v \cdot n_{jk}) \leq n_{jk} \cdot b_{k+1} - d_{jk} + r_{jk} \\ \max\limits_{w \in \mathcal{W}_j} (w \cdot n_{jk}) \leq -n_{jk} \cdot o_j + d_{jk} + r_{jk} \end{cases} \end{aligned} \tag{15}$$

This problem can be simplified further by eliminating the constant d_{jk} across the constraints of problem (15) to get problem (16):

$$\begin{aligned} lb_k &\triangleq \max_{w \in \mathcal{W}_j} (w \cdot n_{jk}) + n_{jk} \cdot o_j - \min_{v \in \mathcal{V}_k} (v \cdot n_{jk}) \\ lb_{k+1} &\triangleq \max_{w \in \mathcal{W}_j} (w \cdot n_{jk}) + n_{jk} \cdot o_j - \min_{v \in \mathcal{V}_{k+1}} (v \cdot n_{jk}) \\ &\min_{x,r} f(x) + \alpha_R \|r\| \\ &\text{s.t. } \forall j,k \begin{cases} lb_k \leq n_j \cdot b_k + r_{jk} \\ lb_{k+1} \leq n_{jk} \cdot b_{k+1} + r_{jk} \end{cases} \end{aligned} \tag{16}$$

With the formulation (16) the problem features only three linear constraints per separating plane.

C. Problem initialization

Because the problem to solve is non-convex, we cannot escape the risk of finding a local minima as solution. Therefore, it is important to guide the resolution towards a satisfactory minima. This can be done by wisely choosing the initial guess. An obvious initial guess for the positions of the bodies is a sequence of evenly spaced positions on a straight line between the initial and final position. This initialization can then be modified depending on the application. For example, in the case of planning a foot trajectory for a walking robot, we can add a vertical motion of the foot that goes as high as the maximum height of a step $h_{\max}$.

$$b_k^0 = b_I + k\frac{b_F - b_I}{N} + h_{\max} \sin \frac{k\pi}{N} \vec{z} \tag{17}$$

Once the initial guess for the bodies position is computed, we can infer an initial guess for the planes by positioning them between the obstacle and the pair of bodies that they are meant to separate.

$$\begin{aligned} n_{jk} &= \frac{b_k+b_{k+1}}{2} - o_j \\ d_{jk} &= (o_j + \frac{n_{jk}}{2}) \cdot \frac{n_{jk}}{\|n_{jk}\|} \\ n_{jk} &\leftarrow \frac{n_{jk}}{\|n_{jk}\|} \end{aligned} \tag{18}$$

D. Convergence criterion

The optimization process can be stopped when a stationary position is found, e.g. when the solutions of PBB and PBP are the same (or close enough in norm) across two successive iterations. Once a stationary point is found for PBB, it is possible that the solutions of PBP are not stationary. In that case, two successive PBP are identical(because the positions of the bodies are unchanged), except for the constraint (7) which depends on the normal found in the previous iteration of PBP. Because of this constraint, a cycling on the solutions of PBP can occur, in which case, a stationary point can never be found. All the elements in such a cycle are solutions of the problem. Thus, we consider that a solution is reached once the positions of the bodies (solutions of PBB) are stationary.

IV. Results and Experimentations

In this section, we present some results obtained with the proposed approach. All the computations of the following experiments are performed on a single thread of an Intel(R) Core(TM) i7-3840QM CPU at 2.80GHz, with 16Go of RAM. In the cases of resolution with alternate problems, one iteration of the algorithm is counted per loop of the algorithm, e.g. PBB and PBP are solved at each iteration. Our goal is to solve problems of foot trajectory planning for humanoid robots. In such problems, only the translations of the foot are usually searched for and the locations of the vertices of the foot can be obtained through (14). Thus, in this section, we only focus on that type of problems in which PBP(n, d) is a QP. In all the following problems, the cost function is chosen to be a combination of the distance traveled by the bodies and their jerks:

$$f(x) = \alpha_P \|b_{k+1} - b_k\|^2 + \alpha_J \|\dddot{b}_k\|^2 \tag{19}$$

A. Large cube avoidance

In this first example, we consider the problem of a collision-free trajectory planning for a foot stepping above a large cubic obstacle(red).

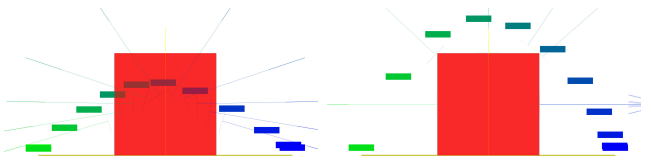

Fig. 4. left: initial guess; right: solution found after 6 iterations

Figure Fig. IV-A illustrates the solution to this problem found by our alternate resolution algorithm in 6 iterations. The feet is represented by a rectangular prism, its initial(blue) and final(green) positions are on opposite sides of the cube, and the initial velocities and accelerations are null. A (linear) constraint requiring the bodies to be above the ground(yellow plane) is added to the problem. The left picture represents the initial guess, and the right one the solution found. A resolution of this problem with a nonlinear solver took 26 iterations, each of these iterations requiring the resolution of a QP more than twice larger than the one solved in the alternate resolution. A QP of size n is solved in approximately $o(n^3)$, thus, solving a single QP that is twice larger should take about 8 times longer. In practice, the problem is solved in 5ms with the alternate problem method and 100ms with the nonlinear solver. The resolution method that we proposed in section III-B is thus much faster than a resolution with nonlinear solver as presented in section III-A.

B. Path through a small opening

Reaching convergence can become more difficult with problems where the initial guess violates the constraints, e.g. penetrates the obstacles. In order to study the robustness of our method to poor initialization, we design an environment such that the optimal trajectory for the bodies goes through a small opening between obstacles, and the initial guess proposed in section III-C is in collision with several obstacles.

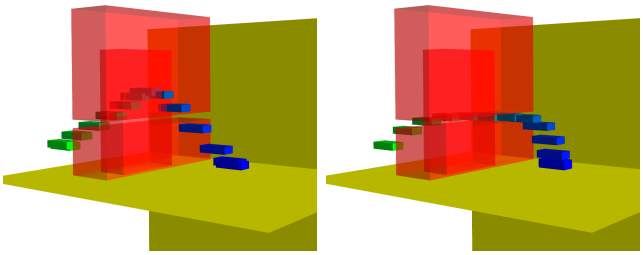

Fig. 5. left: initial guess; right: solution found after 10 iterations

The alternate resolution finds a solution in only 10 iterations while the nonlinear solver takes 81 iterations to converge. Effectively taking respectively 4ms and 3.71s to converge. This example shows that even with a complex scenario where a good initial guess is difficult to find, the alternate resolution approach is still able to find a solution and is robust to penetrations in the initial guess.

Furthermore, the narrowness of the opening in which the body must pass would be an issue for an approach using STP-BV because it relies on the use of a bounding volume that makes the body thicker, thus potentially preventing it from crossing tight fitting openings. Methods based on a distance and interpenetration calculation are also likely to fail in that kind of situations because when crossing the opening, the distance function of the collision avoidance constraint would reach a gradient discontinuity. Our method overcomes those issues.

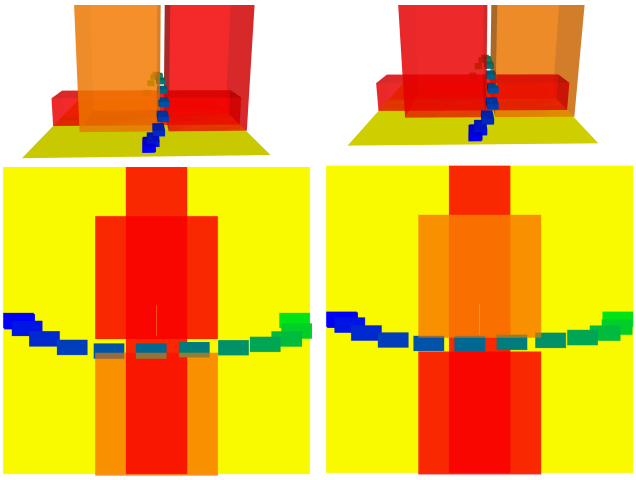

Fig. 6. Solution trajectories to problems with virtual obstacles. Virtual obstacles are in orange and non-virtual ones in red

C. Virtual obstacle

A slight modification of our problems formulation allows to easily setup a form of hierarchy between the obstacles of the problem. Some obstacles can be considered as virtual ones, in the sense that they are avoided if possible, but only if avoiding them does not impact negatively the avoidance of non-virtual obstacles. This can be a useful feature in real scenarios where some obstacles are actually virtual, like the comfort zone of a human. Whereas others, like a physical wall that cannot be avoided shouldn't be considered as virtual. We discriminate through virtual and non-virtual obstacles by manipulating the relaxation terms. We consider two different relaxation terms r_o and r_v, respectively for obstacles and virtual obstacles. They replace the relaxation terms used in the constraints of problem(16), and the cost function is changed to emulate a lexicographic optimization: $\text{lex}\min.(\|r_o\|, \|r_v\|, f(b))$. This is achieved by using an exact penalization of the terms of the cost function:

$$\min_{b, r_o, r_v} \; f(b) + \alpha_o \|r_o\| + \alpha_v \|r_v\| \tag{20}$$

We illustrate this in the following problem where two obstacles(one virtual and one non-virtual) are too close to each other for the feet to pass through the gap between them. We present the solutions to two problems that differ by the choice of the right or left obstacle to be virtual. One can observe that in both cases the solution trajectory violates the non-collision constraint only with the virtual obstacle.

V. Conclusion

In this proceeding, we present a formulation of the collision avoidance constraint that is based on the notion of separating planes, instead of a distance between objects, as is usually done. The collision avoidance between two bodies is ensured by the existence of a separating plane between them. Unlike usual methods, our approach formulates the same way whether the bodies are separated or in interpenetration, and is continuously differentiable, which is an appreciated feature when running smooth optimization algorithms. Optimization problems featuring this type of constraints can be solved as is with nonlinear solvers, but we also propose an alternative resolution method to solves them more efficiently by separating the problem into two subproblems that can be solved iteratively until convergence. We tested our method on several problems of trajectory planning. The use of our custom resolution method with our collision avoidance constraint formulation shows very promising results in terms of robustness and convergence speed. This indicate that such methods could potentially be used on more complex problems, and may be fast enough to be used online to compute and update trajectories for real robots and in particular for computing feet trajectories for walking robots.

References

[1] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Learning, planning, and control for quadruped locomotion over challenging terrain," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 236–258, 2011.

[2] K. Nishiwaki, J. Chestnutt, and S. Kagami, *Planning and Control of a Humanoid Robot for Navigation on Uneven Multi-scale Terrain*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 401–415.

[3] J. J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Footstep planning among obstacles for biped robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, 2001, pp. 500–505 vol.1.

[4] S. Brossette, A. Escande, G. Duchemin, B. Chretien, and A. Kheddar, "Humanoid posture generation on non-Euclidean manifolds," in *IEEE-RAS International Conference on Humanoid Robots*, 2015, pp. 352–358.

[5] O. Stasse, A. Escande, N. Mansard, S. Miossec, P. Evrard, and A. Kheddar, "Real-time (self)-collision avoidance task on a hrp-2 humanoid robot," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, 2008, pp. 3200–3205.

[6] B. Bouilly, T. Simeon, and R. Alami, "A numerical technique for planning motion strategies of a mobile robot in presence of uncertainty," in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, vol. 2. IEEE, 1995, pp. 1327–1332.

[7] J. Schulman, Y. Duan, J. Ho, a. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *International Journal of Robotic Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[8] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi, "A fast procedure for computing the distance between complex objects in three-dimensional space," *IEEE Journal of Robotics and Automation*, vol. 4, no. 2, pp. 193–203, 1988.

[9] E. Gilbert and D. Johnson, "Distance functions and their application to robot path planning in the presence of obstacles," *IEEE Journal on Robotics and Automation*, vol. 1, no. 1, pp. 21–30, 1985.

[10] A. Escande, S. Miossec, and A. Kheddar, "Continuous gradient proximity distance for humanoids free-collision optimized-postures," in *Humanoid Robots, 2007 7th IEEE-RAS International Conference on*. IEEE, 2007, pp. 188–195.

[11] J.-F. Bonnans, J. C. Gilbert, C. Lemaréchal, and C. A. Sagastizábal, *Numerical optimization: theoretical and practical aspects*. Springer Science & Business Media, 2006.

[12] A. Escande, S. Miossec, M. Benallegue, and A. Kheddar, "A strictly convex hull for computing proximity distances with continuous gradients," *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 666–678, June 2014.

[13] G. van den Bergen, *Collision detection in interactive 3D environments*, ser. The Morgan Kaufmann Series in Interactive 3D Technology, D. H. Eberly, Ed. Morgan Kaufmann Publishers, 2004.

[14] K. Border, "Separating hyperplane theorems," Caltech, Tech. Rep.

[15] C. Cortes and V. Vapnik, *Support-Vector Networks*, 1995.