Ballistic motion planning

Mylène Campana, Jean-Paul Laumond

Ballistic motion planning

Mylène Campana[1,2] Jean-Paul Laumond[1,2]

***Abstract*— This paper addresses the motion planning problem for a jumping point-robot. Each jump consists in a ballistic motion linking two positions in contact with obstacle surfaces. A solution path is thus a sequence of parabola arcs. The originality of the approach is to consider non-sliding constraints at contact points: slipping avoidance is handled by constraining takeoff and landing velocity vectors to belong to 3D friction cones. Furthermore the magnitude of these velocities is bounded. A ballistic motion lying in a vertical plane, we transform the 3D problem into a 2D one. We then solve the motion equations. The solution gives rise to an exact steering method computing a jump path between two contact points while respecting all constraints. The method is integrated into a standard probabilistic roadmap planner. Probabilistic completeness is proven. Simulations illustrate the performance of the approach.**

I. INTRODUCTION

Geometrical motion planning is a well-known problem [1]. Today, most motion planners are inspired by the random sampling seminal approaches [2], [3]. They can also be adapted for systems that handle special types of paths (e.g. spline-based approach [4]), or which embed special constraints (e.g. constrained optimization [5]).

In this paper we consider the ballistic motion planning for a jumping robot in environments containing slipping surfaces. It is well-known that a ballistic motion results in a parabola trajectory. According to the Coulomb friction law, a sufficient condition for the robot not to slide during its takeoff is that the contact force belongs to the so-called friction cone. This latter property extends to the landing phase. We consider a simple point mass robot with simplified contact dynamics: we assume that the robot is submitted to an impulse force as soon as it lands, so that the transition between landing and take-off is instantaneous. This gives rise to a discontinuity between the contact forces and the contact velocities. In this simplified contact model, we consider that the robot is not sliding as soon as its landing and take-off velocity vectors belong to the friction cone. Note that the impulse force is also contained in the friction cone. The constraint on the velocity direction is then named non-sliding constraint. Moreover we assume that the robot has limited energy resources, which limit the velocity at takeoff. The landing velocity is also constrained to avoid requiring to dissipate too much energy (and damaging the robot). These energy restrictions are materialized by velocity vector magnitude limitations during the takeoff and landing phases. Constraints on the velocity vector magnitudes are named velocity constraints.

*This work has been supported by the project ERC Advanced Grant 340050 Actanthrope

[1]{mcampana, jpl}@laas.fr
CNRS, LAAS, 7 avenue du colonel Roche, F–31400 Toulouse, France
[2]Univ. de Toulouse, LAAS, F–31400 Toulouse, France

How to plan a collision-free path satisfying both sliding and velocity constraints in such a context? This is the question addressed by this paper.

Ballistic motion planning has been relatively little addressed. [6] shows a one-legged robot hopping while keeping balance. [7] is processing jumps on a miniature robot to climb horizontal stairs. In [8] a multi-articulated gymnast robot jumps above obstacles on a horizontal ground, while taking into account the whole-body angular momentum. In [9]–[13] ballistic motion is considered from a character animation viewpoint. Focus is done on jump motion preparation and adaptation. [12] presents a simulated humanoid robot that is running and jumping on a horizontal platform. The takeoff leg angle and intensity are computed to cross the large gap. Motion planning and obstacle avoidance are considered in [14]. The method computes a sequence of parabolas. It consists in tuning the parabola heights in order to reach different levels while avoiding obstacles.

This paper does not consider the full dynamics of articulated avatars. It is restricted to point-robots. With respect to the state of the art, the contribution is to account for slipping prevention as well as takeoff and landing velocity limitations. Furthermore, the proposed approach applies on 3D environments and rough terrains without any restriction.

We formally state our problem in Section II. In Section III, we detail the unconstrained parabola trajectory equations. Then, we present and solve the non-sliding constraints as well as the velocity limitations in Section IV. Finally, this resolution is integrated into a motion planner (Section V) whose probabilistic completeness is proven. Simulations are provided in Section VI.

II. PROBLEM STATEMENT

Let us consider a point-robot moving in a 3D environment. The robot begins from a starting position $\mathbf{c}_s$ and wants to reach a goal position $\mathbf{c}_g$, only by performing jumps from one contact to another. Both $\mathbf{c}_s$ and $\mathbf{c}_g$ are assumed to be in contact with the environment. There is no distinction between grounds and obstacles. The purpose of this paper is to determine a sequence of jumps, under the following assumptions:

- The robot is modeled by a point mass m of position $\mathbf{c}$ with respect to the origin.
- The only force that applies to the robot during a jump is $m\mathbf{g}$.
- Contact phases are instantaneous, so that the velocity at a contact point is discontinuous, i.e. robot takeoff results from an impulsion.

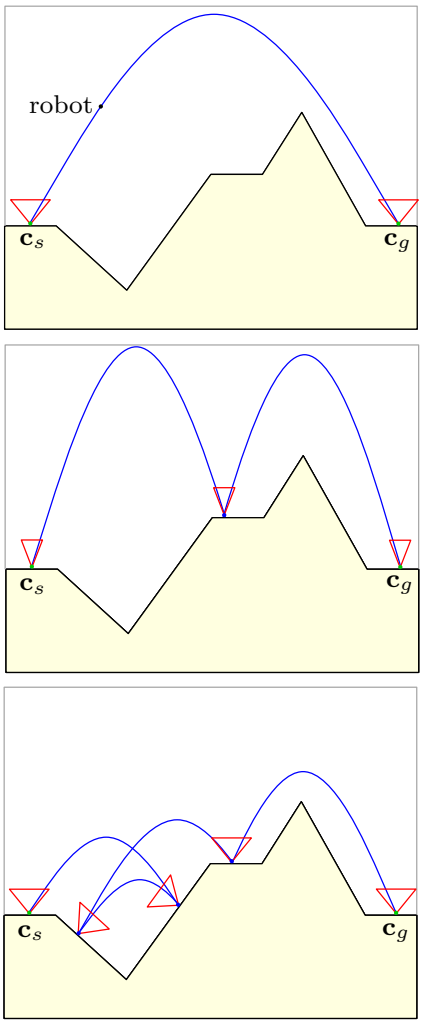

Fig. 1. Three sequences of parabolas between $\mathbf{c}_s$ and $\mathbf{c}_g$ positions for different constraints. In the case illustrated in the middle, friction cones are narrower than the top ones, so that the parabola on top is not admissible anymore and a waypoint has to be used. On the bottom case, the initial velocity has been limited compared to the top case, resulting in a sequence with numerous waypoints.

- The non-sliding constraint is modeled by a friction cone. We assume that a position is non-sliding as soon as its arrival and restart velocity vectors belong to the friction cone. Moreover, the surface material is uniform in the environment, i.e. the non sliding constraints can be modeled everywhere by a friction cone with constant coefficient. We denote by μ the tangent of the cone half-angle.
- Takeoff and landing velocity magnitudes are bounded by the same value, so that an admissible jump path can be traveled in reverse.
- There is no constraint on the energy balance between a jump and the next one.
- The robot cannot collide the obstacles.

Fig. 1 illustrates the effects of the friction and velocity constraints on the existence of parabola sequences. The following section reminds the basics of ballistic motion and details the equations of parabolas linking two points.

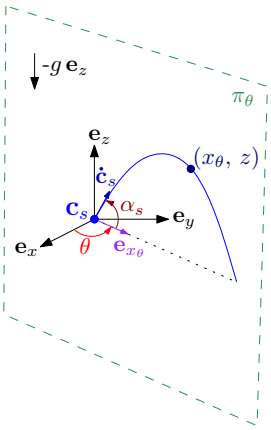

Fig. 2. The parabola always belongs to the plane π_θ defined by $(\mathbf{c}_s; \mathbf{e}_{x_\theta}; \mathbf{e}_z)$, where $\mathbf{e}_{x_\theta} = \cos(\theta)\mathbf{e}_x + \sin(\theta)\mathbf{e}_y$.

III. UNCONSTRAINED BALLISTIC MOTION

A. Accessible space of ballistic motion

We denote the global frame basis by $(\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z)$. When the Newton second law of motion is integrated with respect to time for a ballistic shot from the $\mathbf{c}_s$ position with a $\dot{\mathbf{c}}_s$ initial velocity, the following robot trajectory is obtained:

$$\mathbf{c}(t) = -\frac{g}{2}\, t^2\, \mathbf{e}_z + \dot{\mathbf{c}}_s\, t + \mathbf{c}_s \tag{1}$$

Let $(x, y, z)^T$ be the coordinates of $\mathbf{c}$. The ballistic motion belongs to a vertical plane denoted by π_θ. The orientation of the plane is given by the initial velocity components as follows:

$$\theta = \text{atan2}(\dot{y}_s, \dot{x}_s) \in [-\pi; \pi]$$

Considering $\mathbf{\Theta} = [\cos(\theta)\ \sin(\theta)\ 0]^T$, we introduce the following variable changes:

$$x_\theta = \mathbf{c}.\mathbf{\Theta}, \quad x_{\theta_s} = \mathbf{c}_s.\mathbf{\Theta}$$

$$x_{\theta_g} = \mathbf{c}_g.\mathbf{\Theta}, \quad \dot{x}_{\theta_s} = \dot{\mathbf{c}}_s.\mathbf{\Theta}$$

Thus from (1), one can rewrite the main equations of motion determining the robot coordinates $(x_\theta, z)^T$ in π_θ (see Fig. 2):

$$z = -\frac{g}{2}\frac{(x_\theta - x_{\theta_s})^2}{\dot{x}_{\theta_s}^2} + \frac{\dot{z}_s}{\dot{x}_{\theta_s}}(x_\theta - x_{\theta_s}) + z_s \tag{2}$$

$$\frac{\dot{z}}{\dot{x}_{\theta_s}} = -g\frac{x_\theta - x_{\theta_s}}{\dot{x}_{\theta_s}^2} + \frac{\dot{z}_s}{\dot{x}_{\theta_s}} \tag{3}$$

Let us denote the takeoff angle by $\alpha_s = \text{atan2}(\dot{z}_s, \dot{x}_{\theta_s})$ and the velocity value $||\dot{\mathbf{c}}_s||$ by v_s. Equations (2)-(3) highlight the two parameters α_s and $\dot{x}_{\theta_s}$ that determine a parabola in π_θ. For instance, Fig. 3 presents the parabola beams when v_s (resp. α_s) is fixed. This can also be viewed as the accessible space of the robot performing ballistic motions.

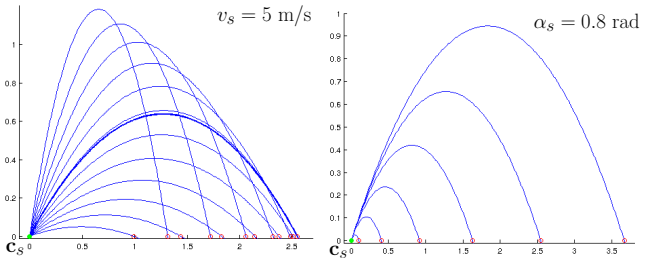

Fig. 3. Accessible space from $\mathbf{c}_s = (0,0)^T$ when varying α_s (left), or when varying initial velocity v_s (right). On left, the bold parabola leads to the maximal range at given initial velocity, and is obtained for $\alpha_s = \frac{\pi}{4}$.

B. Goal-oriented ballistic motion

Now we want our robot to reach the goal position $\mathbf{c}_g$ with a jump starting from $\mathbf{c}_s$. Therefore, the value $\theta = \text{atan2}(y_g - y_s, x_g - x_s)$ is now known. Let X_θ equal $x_{\theta_g} - x_{\theta_s}$ and Z equal $z_g - z_s$. Since Z is fixed, it appears that from (2), α_s is the only remaining variable to compute the parabola beam leading to $\mathbf{c}_g$. In fact, the initial velocity $\dot{x}_{\theta_s}$ can be obtained with the following equation:

$$\dot{x}_{\theta_s} = \sqrt{\frac{gX_\theta^2}{2(X_\theta \tan(\alpha_s) - Z)}} \tag{4}$$

Equation (4) implies that $\dot{x}_{\theta_s}$ is only defined for top-curved parabolas, which is consistent with the gravity. Non-physically-feasible parabolas as down-curved ones are not considered. Therefore we impose:

$$\text{atan2}(Z, X_\theta) < \alpha_s < \frac{\pi}{2} \tag{5}$$

An example of a goal-oriented parabola beam is presented Fig. 4. Finally, we denote a parabola starting from $\mathbf{c}_s$ and its parameters by $\mathcal{P}_s(\theta, \alpha, v)$.

IV. BALLISTIC MOTION WITH CONSTRAINTS

So far we have defined a beam of feasible parabolas to connect two positions. In this section, the non-sliding and velocity constraints are introduced, and the resulting reduction of the space of admissible parabola beams is detailed.

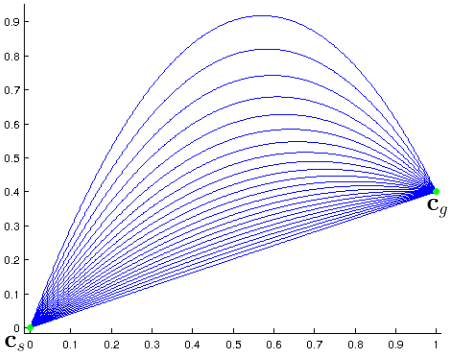

Fig. 4. Physically-feasible parabolas linking $\mathbf{c}_s$ and $\mathbf{c}_g$, for multiple values of α_s in $[0.91; 1.27]$ rad.

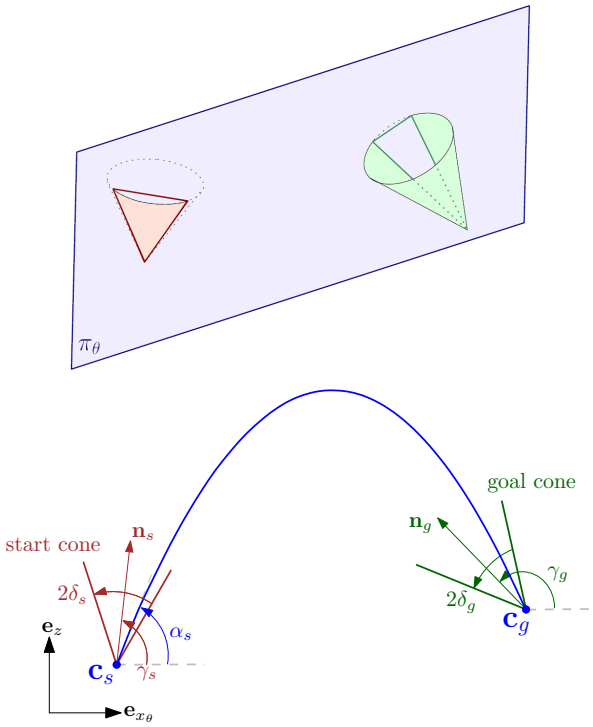

Fig. 5. (Top) Representation of the intersection between π_θ and two 3D cones. (Bottom) 2D cones resulting of the intersection and an example of parabola belonging to both cones.

A. Non-sliding constraints

1) 2D reduction: Let us consider two points $\mathbf{c}_s$ and $\mathbf{c}_g$ at the contact of environment surfaces. Non-sliding constraints impose the robot to land and take off along velocity vectors that belong to 3D friction cones of apexes based on $\mathbf{c}_s$ and $\mathbf{c}_g$.

Since the motion has to lie in a vertical plane π_θ, the problem of computing a parabola between the two 3D friction cones is reduced to a 2D problem. Corresponding 2D cones result from the intersections of the 3D cones with the plane π_θ (see Fig. 5 top). If one of both intersection sets is reduced to a point, then there is no possible jump between $\mathbf{c}_s$ and $\mathbf{c}_g$. Otherwise, let us denote their half-apex angles[1] respectively by δ_s and δ_g, and their directions projected in π_θ respectively by $\mathbf{n}_s$ and $\mathbf{n}_g$. Note that δ_s and δ_g may be smaller that $\arctan(\mu)$. For the 2D start cone of direction $\mathbf{n}_s = (n_{x_s}, n_{y_s}, n_{z_s})^T$, let us denote by γ_s the angle between the cone direction and the horizontal line:

$$\gamma_s = \text{atan2}(n_{z_s}, n_{x_s}\cos(\theta) + n_{y_s}\sin(\theta))$$

γ_g is similarly defined, respectively to the 2D goal cone. Thus the problem of slipping avoidance is reduced to the problem of finding a parabola going through the 2D cones (see Fig. 5 bottom).

2) Constraint formulation: Since the equation of a parabola starting at $\mathbf{c}_s$ and ending at $\mathbf{c}_g$ only depends on α_s, the four constraints are just formulated relatively to α_s.

[1]Detailed computation of δ_s and δ_g is presented in the appendix file.

Algorithm 1 Resolution of the landing cone constraint.

Output: Defined constraint bounds α_2^-, α_2^+

if $\gamma_g > 0$ **then**
  $\alpha_g^- = \gamma_g - \pi - \delta_g$
  $\alpha_g^+ = \gamma_g - \pi + \delta_g$
  if $\alpha_g^+ < -\frac{\pi}{2}$ **then**
    No solution
  else
    $\alpha_2^- = \arctan(\frac{2Z}{X_\theta} - \tan(\alpha_g^+))$
    if $\alpha_g^- > -\frac{\pi}{2}$ **then**
      $\alpha_2^+ = \arctan(\frac{2Z}{X_\theta} - \tan(\alpha_g^-))$
    else
      α_2^+ not defined
else
  $\alpha_g^- = \gamma_g + \pi - \delta_g$
  $\alpha_g^+ = \gamma_g + \pi + \delta_g$
  if $\alpha_g^+ > \frac{\pi}{2}$ **then**
    No solution
  else
    $\alpha_2^+ = \arctan(\frac{2Z}{X_\theta} - \tan(\alpha_g^-))$
    if $\alpha_g^+ < \frac{\pi}{2}$ **then**
      $\alpha_2^- = \arctan(\frac{2Z}{X_\theta} - \tan(\alpha_g^+))$
    else
      α_2^- not defined

For the non-sliding takeoff constraint, the inequalities on α_s are immediate:

$$\alpha_1^- \leq \alpha_s \leq \alpha_1^+ \quad \text{with} \quad \begin{cases} \alpha_1^- = \gamma_s - \delta_s \\ \alpha_1^+ = \gamma_s + \delta_s \end{cases}$$

To express the three remaining constraints according to α_s, (2)-(3) are brought back to the parabola origin $\mathbf{c}_s$. Thus constraints are still expressed as inequalities:

$$\alpha_i^- \leq \alpha_s \leq \alpha_i^+ \ , \ i \in \{2..4\}$$

For the landing cone constraint, different cases appear, depending on the accessibility of the cone. They are tackled by Algorithm 1, which returns the constraint bounds (α_2^-, α_2^+). Note that one of the bounds may not exist, and that the constraint may also not be satisfied.

B. Velocity constraints

Takeoff velocity limitation is expressed as $v_s \leq V_{max}$. Equation (2) leads to:

$$gX_\theta^2 \tan(\alpha_s)^2 - 2X_\theta V_{max}^2 \tan(\alpha_s) + gX_\theta^2 + 2ZV_{max}^2 \leq 0 \quad (6)$$

$$\Delta = V_{max}^4 - 2gZV_{max}^2 - g^2X_\theta^2$$

If $\Delta < 0$, (6) has no solution. In other words, it means that the goal position is not reachable with an initial velocity satisfying the limitation. In the case where the constraint is solvable, we write:

$$\begin{cases} \alpha_3^- = (V_{max}^2 - \sqrt{\Delta})/gX_\theta \\ \alpha_3^+ = (V_{max}^2 + \sqrt{\Delta})/gX_\theta \end{cases}$$

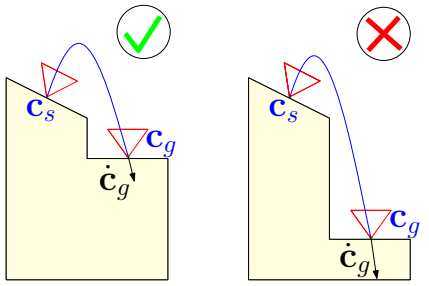

Fig. 6. The landing velocity limitation allows to reject parabolas that have a too important impact velocity magnitude v_g (right).

The same argument can be applied for the landing velocity limitation (see Fig. 6). Using (3)-(4), we can rewrite the constraint equation $v_f \leq V_{max}$ as:

$$gX_\theta^2 \tan(\alpha_s)^2 - (4X_\theta Zg + 2X_\theta V_{max}^2) \tan(\alpha_s) + gX_\theta^2 + 2ZV_{max}^2 + 4gZ^2 \leq 0 \quad (7)$$

$$\Lambda = V_{max}^4 + 2gZV_{max}^2 - g^2X_\theta^2$$

If $\Lambda < 0$, (7) has no solution. Otherwise, we write:

$$\begin{cases} \alpha_4^- = (V_{max}^2 + 2gZ - \sqrt{\Lambda})/gX_\theta \\ \alpha_4^+ = (V_{max}^2 + 2gZ + \sqrt{\Lambda})/gX_\theta \end{cases}$$

A symmetry property of the parabola implies that, given one parabola from $\mathbf{c}_s$ to $\mathbf{c}_g$ determined by $\dot{\mathbf{c}}_s$, the same parabola can be obtained from $\mathbf{c}_g$ to $\mathbf{c}_s$ with $-\dot{\mathbf{c}}_g$ as initial velocity. In the latter case, the contact velocity becomes $-\dot{\mathbf{c}}_s$. We use this property to apply the same bound V_{max} for the takeoff and landing velocities. Thus the parabola can be traveled both ways without violating the velocity limitation constraints.

C. Constraints collection and solution existence

The domains where constraints are satisfied are convex. Thus, we intersect these domains to determine if an interval $]\alpha_s^-; \alpha_s^+[$ of α_s values complying with all the constraints exists. The interval bounds are given by:

$$\begin{cases} \alpha_s^- = \max(\alpha_1^-, \alpha_2^-, \alpha_3^-, \alpha_4^-) \\ \alpha_s^+ = \min(\alpha_1^+, \alpha_2^+, \alpha_3^+, \alpha_4^+) \end{cases}$$

Note that (5) has to be simultaneously satisfied to consider an admissible parabola. Fig. 7 presents an illustration of this constraint intersection. Constraint bounds $(\alpha_i^-, \alpha_i^+)_{i \in \{1..4\}}$ are used to plot parabolas, representing the domains where constraints are satisfied. The intersection of these domains leads to the set of possible solutions.

Finally, the existence of an admissible jump between two points is guaranteed as soon as:

- Neither of the intersections between both friction cone and π_θ is reduced to a point.
- (α_s^-, α_s^+) are defined and $\alpha_s^- \leq \alpha_s^+$.

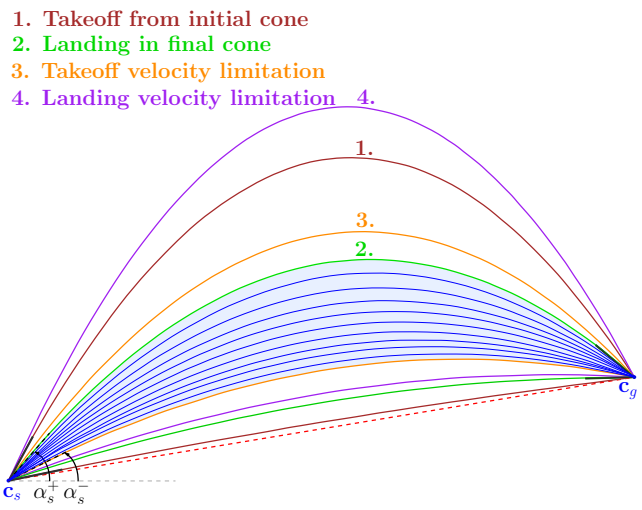

Fig. 7. Illustration of the constraints on a practical example: each constraint bound is used as α_s and represents a bold parabola. Between these bounds, the constraint is satisfied, out of them not. The constraint intersection is given by the bounds (α_s^-, α_s^+) and illustrated by the gray zone: blue parabolas belonging to it are admissible solutions to the problem.

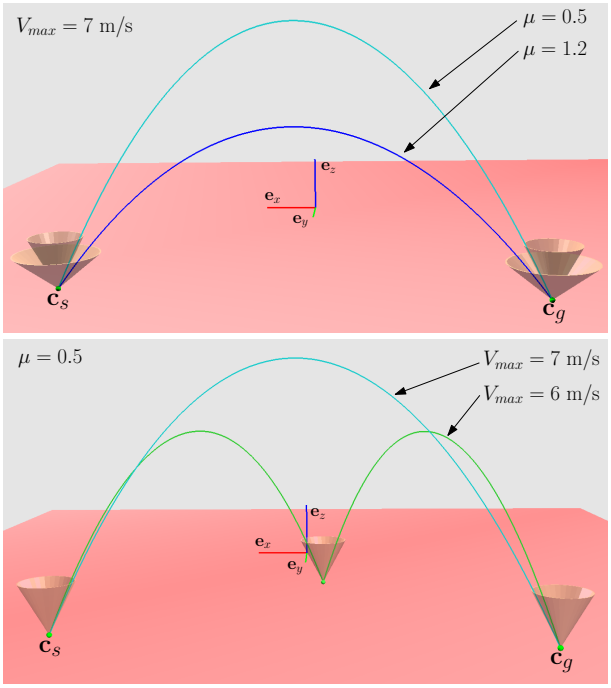

Fig. 8. Two parabola examples linking $\mathbf{c}_s$ to $\mathbf{c}_g$ with different constraints. (Top) large and narrow friction cones are considered, forcing the solution parabola to be adapted. (Bottom) with a large velocity limitation, $\mathbf{c}_g$ can be directly reached. Otherwise, an intermediate position has to be considered.

The two conditions are necessary and sufficient. The interval $]\alpha_s^-; \alpha_s^+[$ gives a simple parametrization of the solution beam. Choosing α_s as the middle $0.5(\alpha_s^- + \alpha_s^+)$ allows to optimize the distance to the constraints, e.g. to be far from the limits of the friction cones, and so far from sliding. Fig. 8 illustrates the constraint effects on a simple example.

Topological Property: Let us consider an admissible parabola $\mathcal{P}_s(\theta, \alpha, v)$ starting at $\mathbf{c}_s$ and ending at $\mathbf{c}_g$. There exists a neighbor $\mathcal{N}_s$ (resp. $\mathcal{N}_g$) of $\mathbf{c}_s$ (resp. $\mathbf{c}_g$) such that any pair of points ($\mathbf{c}_s^*$,$\mathbf{c}_g^*$) belonging to $\mathcal{N}_s \times \mathcal{N}_g$ can be linked by an admissible parabola.

Proof: Let us consider the parabola family $\{\mathcal{P}_s(\theta + e, \alpha + e, v + e),\ e \in]-\epsilon, \epsilon[\}$ starting at $\mathbf{c}_s$. The functions giving the parabola parameters from the coordinates of $\mathbf{c}_g$ are continuous. Therefore such a family spans a neighborhood of $\mathbf{c}_g$. Let $\mathbf{c}_g^*$ be a point of this neighbor and $\mathcal{P}_s(\theta + e^*, \alpha + e^*, v + e^*)$ the parabola from $\mathbf{c}_s$ to $\mathbf{c}_g^*$. $\mathbf{c}_g^*$ may be chosen close enough from $\mathbf{c}_g$ to guarantee that e^* is small enough, and then $\mathcal{P}_s(\theta + e^*, \alpha + e^*, v + e^*)$ is admissible. Because the construction is symmetric, let us consider the same parabola as starting at $\mathbf{c}_g^*$ and ending at $\mathbf{c}_s$. We get a new parametrization of the same parabola, i.e. $\mathcal{P}_g(\theta^*, \alpha^*, v^*)$. By using the same argument as above, the parabola family $\{\mathcal{P}_g(\theta^* + e, \alpha^* + e, v^* + e),\ e \in]-\epsilon, \epsilon[\}$ starting at $\mathbf{c}_g^*$ spans a neighbor of $\mathbf{c}_s$. The property holds for any point $\mathbf{c}_g^*$ sufficiently close to $\mathbf{c}_g$. Therefore, there exists a neighbor $\mathcal{N}_s$ (resp. $\mathcal{N}_g$) of $\mathbf{c}_s$ (resp. $\mathbf{c}_g$) such that any pair of points ($\mathbf{c}_s^*$,$\mathbf{c}_g^*$) belonging to $\mathcal{N}_s \times \mathcal{N}_g$ can be linked by an admissible parabola. ■

V. MOTION PLANNING

To find a sequence of parabola arcs between an initial position $\mathbf{c}_s$ and a final one $\mathbf{c}_g$, we use a simple PRM-based probabilistic roadmap planner [2] (see Algorithm 2). Note that the roadmap can contain cycles. The planner builds a roadmap in the 3D space by randomly sampling contact positions according to a standard uniform law (RANDOMSAMPLE), and by linking them (STEER) with admissible collision-free parabola arcs. The roadmap construction is over as soon as either a path linking $\mathbf{c}_s$ and $\mathbf{c}_g$ is found (ARECONNECTED), or computation time is over. Then, the function FINDSHORTESTPATH explores the roadmap to return the shortest path sequence, in terms of sum of parabola lengths.

The function BEAM is described in Algorithm 3. It computes the interval of takeoff angles that generate constrained parabolas to link $\mathbf{c}_s$ and $\mathbf{c}_g$. The algorithm starts by calculating each cone and plane π_θ intersection, and continues only if

Algorithm 2 Probabilistic roadmap planner for ballistic motion planning.

Input: *environment*, $\mathbf{c}_s$, $\mathbf{c}_g$, μ, V_{max}
Output: Collision-free solution *sequence* to problem

```
path ← STEER(c_s, c_g)
finished ← ARECONNECTED(c_s, c_g)
while not(finished) do
    c_random ← RANDOMSAMPLE()
    ADDTOROADMAP(c_random)
    for c_node ∈ Roadmap do
        path ← STEER(c_node, c_random)
        ADDTOROADMAP(path)
    end for
    finished ← ARECONNECTED(c_s, c_g)
return sequence ← FINDSHORTESTPATH()
```

Algorithm 3 BEAM($\mathbf{c}_s, \mathbf{c}_g$): Computes the parabola beam represented by the takeoff angle interval $]\alpha_s^-;\alpha_s^+[$.

Input: $\mathbf{c}_s,\ \mathbf{c}_g,\ \mu,\ V_{max}$
Output: Interval of takeoff angles I_{beam}

```
cone_s^2D ← COMPUTEINTERSECTION(cone_s^3D, π_θ)
cone_g^2D ← COMPUTEINTERSECTION(cone_g^3D, π_θ)
if ISREDUCED(cone_s^2D) or ISREDUCED(cone_g^2D) then
    return I_beam ← ∅
(α_i^- α_i^+, fail)_{i∈{1..4}} ← COMPUTECONSTRAINTS()
if fail = true then return I_beam ← ∅
α_s^- ← max(α_1^-, α_2^-, α_3^-, α_4^-)
α_s^+ ← min(α_1^+, α_2^+, α_3^+, α_4^+)
return I_beam ←]α_s^-; α_s^+[
```

Algorithm 4 STEER($\mathbf{c}_s, \mathbf{c}_g$): Steering method based on a constrained parabola. Returns a collision-free path linking $\mathbf{c}_s$ and $\mathbf{c}_g$. Otherwise, returns an empty path.

Input: $\mathbf{c}_s,\ \mathbf{c}_g,\ \mu,\ V_{max},\ n_{limit}$
Output: Collision-free parabola path $path$

```
n ← 0
I_beam ← BEAM(c_s, c_g)
if ISEMPTY(I_beam) then return path ← emptyPath
else ]α_s^-; α_s^+[ ← I_beam
α_s ← 0.5(α_s^- + α_s^+)
path ← COMPUTEPARABOLA(c_s, c_g, α_s)
while HASCOLLISIONS(path) and n < n_limit do
    α_s ← DICHOTOMY(]α_s^-; α_s^+[)
    path ← COMPUTEPARABOLA(c_s, c_g, α_s)
    n ← n + 1
if HASCOLLISIONS(path) then path ← emptyPath
return path
```

both intersections are not reduced to the cone apexes. Then, takeoff angle bounds related to constraints are computed as in Section IV. A boolean $fail$ conveys the feasibility of constraints, i.e. if one constraint cannot be satisfied, $fail$ is set to $false$. At this stage, an admissible parabola exists if the global constraint bounds verify $\alpha_s^- \leq \alpha_s^+$.

Then the steering method STEER detailed in Algorithm 4 selects a takeoff angle α_s and tests the corresponding parabola for collisions. If the parabola is not collision-free (HASCOLLISIONS), then we proceed by dichotomy on the interval $]\alpha_s^-;\alpha_s^+[$ until the resolution threshold n_{limit} is reached. In the worst case, the DICHOTOMY function allows to span the almost entire parabola beam. Doing so, the algorithm is probabilistically complete as proven by the following property.

Convergence Property: Let us consider a sequence of collision-free ballistic jumps between two points $\mathbf{c}_s$ and $\mathbf{c}_g$ in a given environment. Let us assume that the entire path is away about ϵ from the obstacles. Then the probability for Algorithm 2 to find a sequence of collision-free ballistic jumps between $\mathbf{c}_s$ and $\mathbf{c}_g$ converges to 1 when running time tends to infinity.

Proof: The property is a direct consequence of the topological property exposed in Section IV. Indeed let us consider a sequence of collision-free ballistic jumps between two points $\mathbf{c}_s$ and $\mathbf{c}_g$. Let $\mathbf{c}_i$ and $\mathbf{c}_{i+1}$ two consecutive points in the sequence. $\mathbf{c}_i$ and $\mathbf{c}_{i+1}$ are linked by a collision-free parabola $\mathcal{P}_i$. From the topological property, there are two neighbors $\mathcal{N}_i$ (resp. $\mathcal{N}_{i+1}$) of $\mathbf{c}_i$ (resp. $\mathbf{c}_{i+1}$) such that any pair of points $(\mathbf{c}_i^*,\mathbf{c}_{i+1}^*)$ belonging to $\mathcal{N}_i \times \mathcal{N}_{i+1}$ can be linked by an admissible parabola $\mathcal{P}_i^*$. Because Algorithm 2 tends to sample the environment uniformly, the probability to sample two points in $\mathcal{N}_i$ and $\mathcal{N}_{i+1}$ respectively tends to 1 when time tends to infinity. $\mathcal{N}_i$ and $\mathcal{N}_{i+1}$ can be arbitrarily small. As a consequence, $\mathcal{P}_i^*$ can be arbitrarily close to $\mathcal{P}_i$. Because $\mathcal{P}_i$ is away about ϵ from the obstacles, $\mathcal{P}_i^*$ is guaranteed to be collision-free. ■

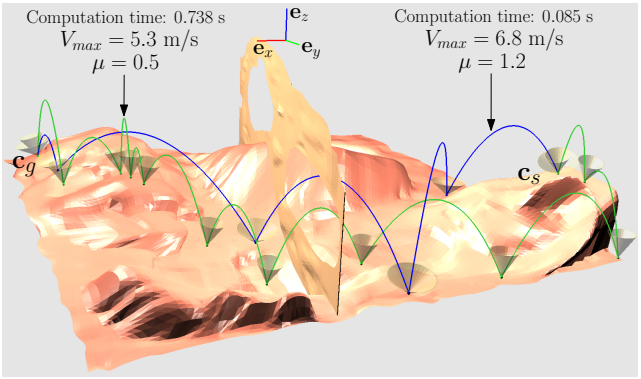

Fig. 9. Full path planning results in an environment containing two windows to cross from right to left. The green solution is more constrained, so it results in a longer sequence of parabolas. Number of triangles: 47 733.

VI. RESULTS

The ballistic motion planner has been implemented on the software Humanoid Path Planner [15] and tested in 3D environments containing sliding surfaces. In all described examples, the parameter n_{limit} from Algorithm 4 has been set to 6.

We have planned sequences of parabolas for a point-robot in three environments. For each example, we consider weak and strong constraints. The results are shown[2] in Fig. 9, 10 and 11. Solutions under strong constraints tend to increase the number of waypoints. It is not only a consequence of the velocity limitation that forces to reach closer positions (see also Fig. 8 bottom), but it is also a result of the cone narrowness. In fact, as it is shown in Fig. 8 top, narrow cones provide parabolas with higher heights, more likely to produce collisions or to exceed the environment bounds.

Table I presents the average performance results of the ballistic motion planner run on the Fig. 10 example. The velocity limitation is less restrictive in terms of computation time than the cone coefficient. However, the velocity limitation cannot be reduced without endangering the existence of a solution. In fact, the robot has to reach other platforms in order to find a solution path sequence.

[2]Movies of the trajectories are available in the attached video and at https://homepages.laas.fr/mcampana/drupal/content/ballistic-motion-planning.

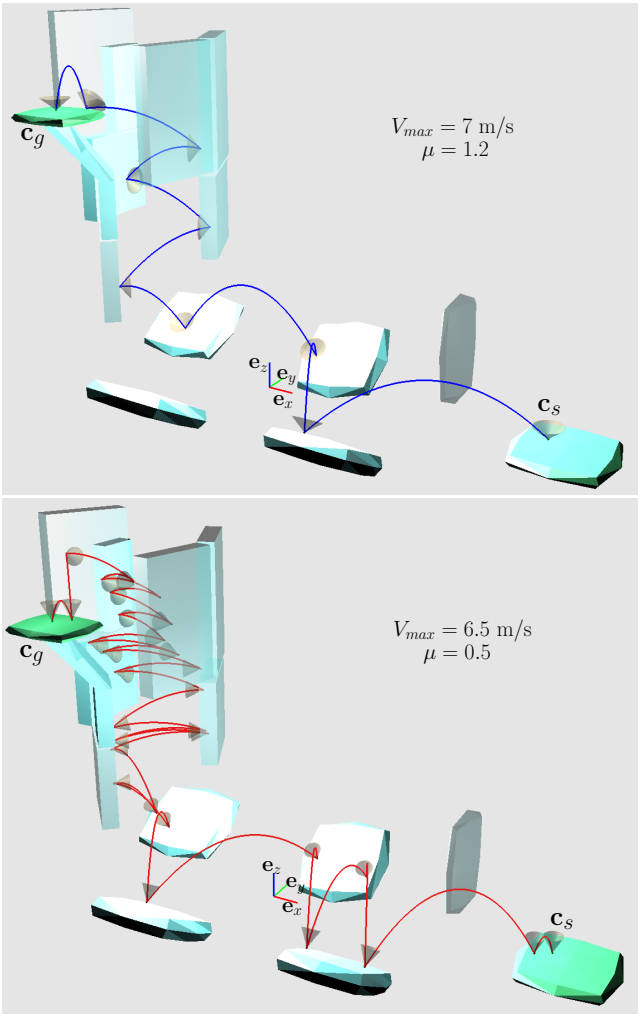

Fig. 10. Full path planning results in an environment containing platforms and a chimney. Top case involves large cones, bottom case narrow cones. Reducing μ prevents the creation of a parabola linking two low platforms with the same inclination. Number of triangles: 696.

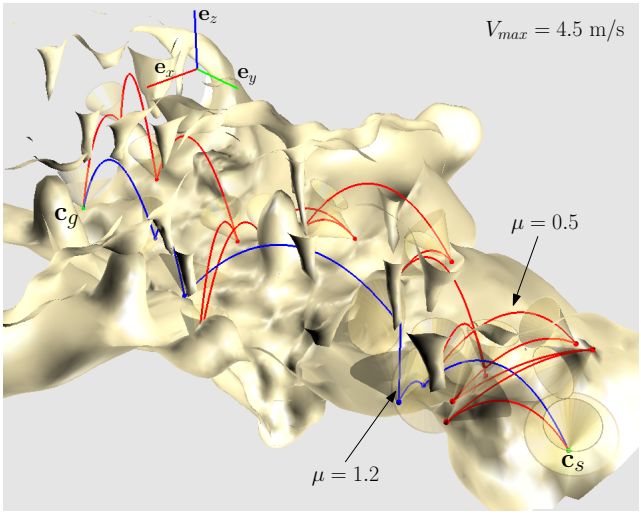

Fig. 11. Path planning results in a cave. The robot has to avoid the numerous stalactites, stalagmites and holes. Number of triangles: 33 513.

Parameters μ	Parameters V_{max}	Computation time (s)	Collision found	Roadmap nodes	Path length (m)
0.5	6.5 m/s	9.89	3270	1995	39.9
0.5	7 m/s	9.99	4002	1835	37.4
1.2	6.5 m/s	1.04	601	282	28.1
1.2	7 m/s	0.909	540	237	27.0

TABLE I

AVERAGES OF 40 BALLISTIC PLANNING OF THE EXAMPLE FIG. 10, FOR FOUR COMBINATIONS OF THE PARAMETERS.

VII. CONCLUSION

We presented a method that analytically computes a non-sliding jump for a point-robot, resulting in a parabola going from one friction cone to another. The method has been implemented as a steering method in a probabilistic-roadmap motion planner in order to determine a sequence of jumps between given start and goal positions.

We can easily extend the parabola-based steering method to devise a diffusing type of probabilistic planner, such as RRT [3]. Besides, our algorithm is the first stage of a more ambitious challenge. Our final purpose is to address dynamic motion planning for digital artifacts. The solution which we provide can be used to compute the center of mass path when the artifact is jumping. Now, it remains to consider more realistic models of contacts (e.g. multiple contacts involving feet and hands) and impacts (e.g. including energy balance). Moreover, the steering method we consider in the motion planner is assumed to be symmetric. This assumption is not realistic. Indeed, for a given parabola, the energy required to overcome the gravity effect from a position is greater than the energy to dissipate when landing at the same position. The extension of the motion planner to more realistic energetic models is the purpose of future developments.

REFERENCES

[1] J. C. Latombe, *Robot Motion Planning*. Boston, MA: Kluwer Academic Publishers, 1991.

[2] L. Kavraki, P. Svestka, J. C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Rob. Autom.*, vol. 12, no. 4, pp. 566–580, 1996.

[3] S. M. LaValle and J. J. Kuffner, Jr., "Rapidly-exploring random trees: Progress and prospects," in *Algorithmic and Computational Robotics: New Directions*, 2000, pp. 293–308.

[4] B. Lau, C. Sprunk, and W. Burgard, "Kinodynamic motion planning for mobile robots using splines," *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pp. 2427–2433, 2009.

[5] M. Zucker, N. Ratliff, A. Dragan, M. Pivtoraiko, M. Klingensmith, C. Dellin, J. Bagnell, and S. Srinivasa, "CHOMP: covariant hamiltonian optimization for motion planning," *Int. J. Rob. Res.*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[6] M. Raibert, M. Chepponis, and H. B. Brown, "Experiments in balance with a 3d one-legged hopping machine," *Int. J. Rob. Res.*, vol. 3, pp. 75–92, 1984.

[7] S. Stoeter and N. Papanikolopoulos, "Autonomous stair-climbing with miniature jumping robots," *IEEE Transactions on Systems, Man, and Cybernetics: Part B*, vol. 35, no. 2, pp. 313–325, 2005.

[8] E. Papadopoulos, I. Fragkos, and I. Tortopidis, "On robot gymnastics planning with non-zero angular momentum," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2007, pp. 1443–1448.

[9] A. Sulejmanpašić and J. Popović, "Adaptation of performed ballistic motion," *ACM Trans. Graph.*, vol. 24, no. 1, pp. 165–179, 2005.

[10] P. Reitsma and N. Pollard, "Perceptual metrics for character animation: Sensitivity to errors in ballistic motion," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 537–542, 2003.
[11] P. Reitsma, J. Andrews, and N. Pollard, "Effect of character animacy and preparatory motion on perceptual magnitude of errors in ballistic motion," *Comput. Graph. Forum*, vol. 27, no. 2, pp. 201–210, 2008.
[12] P. Wensing and D. Orin, "Development of high-span running long jumps for humanoids," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 222–227.
[13] ——, "Improved computation of the humanoid centroidal dynamics and application for whole-body control," *Int. J. Humanoid Rob. (Special Issue on Whole-Body Control)*, vol. 13, pp. 1–23, 2015.
[14] K. Yamane and K. W. Sok, "Planning and synthesizing superhero motions," in *Conference on Motion in Games*, 2010, pp. 254–265.
[15] F. Lamiraux, "Humanoid path planner," http://projects.laas.fr/gepetto/index.php/Software/Hpp, 2014.