Dynamic Non-Prehensile Object Transportation

Puttichai Lertkultanon and Quang-Cuong Pham
School of Mechanical and Aerospace Engineering
Nanyang Technological University, Singapore

***Abstract*—When possible, non-prehensile transportation (i.e. transporting objects without grasping them) can be faster and more efficient than prehensile transportation. However, the need to explicitly consider reaction and friction forces yields kinodynamic constraints that are difficult to take into account by traditional planning algorithms. Based on the recently developed Admissible Velocity Propagation algorithm, we propose here a fast and general non-prehensile transportation scheme. Our contribution is twofold. First we show how to cast the dynamic balance constraints of a 3D object (e.g. a bottle) into a form compatible with the AVP algorithm. Second, we extend the AVP-RRT algorithm into a more efficient AVP-biRRT algorithm, which makes use of the idea of concurrently growing two trees, one rooted at the starting configuration and one rooted at the goal configuration. We also show both in simulations and on a real robot how our algorithm allows planning fast and dynamic trajectories for the non-prehensile transportation of a bottle.**

I. Introduction

Robots can carry objects by either grasping them (prehensile transportation) or not (non-prehensile transportation). Prehensile transportation is the most common option since it allows controlling all the degrees of freedom of the carried object. However, when possible, non-prehensile transportation can be faster and more efficient since the often time-consuming grasping and ungrasping stages are entirely skipped. Moreover, in many applications, the objects to be carried are too soft, fragile or small to be adequately grasped (e.g. food, electronic components, etc.)

Unlike prehensile transportation, non-prehensile transportation relies entirely on reaction and friction forces to keep the object stationary with respect to the robot end-effector – usually a tray. Thus, planning robot trajectories for non-prehensile transportation – especially at high speeds – is very difficult since the reaction and friction forces need to be explicitly taken into account, resulting in *kinodynamic* constraints [1], [2] in addition to the traditional geometric constraints (joint limits, obstacle avoidance, etc.) associated with prehensile transportation.

There are two main approaches to planning with kinodynamic constraints. The first approach works directly in the *state space* of the robot (usually the joint angles + joint velocities) [2], [3]. As this approach involves searching in a space of size $2n$ where n is the number of robot joint angles, it can be highly inefficient in most practical applications.

The second approach *decouples* the problem: first, search for a collision-free *path* in the robot joint space of dimension n and second, find a time-parameterization of that path that satisfies the kinodynamic constraints [4]. However, a major drawback in this approach is that the path obtained in the first step may not have any valid time parameterization at all. One solution to this drawback is to constrain the planner to consider only quasi-statically feasible motions (i.e. motions that are feasible when executed at very low speeds). Obviously, any path obtained from the quasi-static planner is guaranteed to be time-parameterizable. However, all dynamically-feasible motions that are not quasi-statically feasible are overlooked. This results in longer trajectories and even considerable inefficiency in the cases when a quasi-static trajectory is difficult to find or does not exist at all [5].

Recently, a new method was proposed to address this drawback, allowing the decoupling approach to discover truly dynamic trajectories. This method is based on the Admissible Velocity Propagation (AVP) algorithm which, given an interval of reachable velocities at the *beginning* of a path, determine the interval of all velocities the system can reach *after* traversing that path while respecting the kinodynamic constraints. Combining AVP with standard sampling-based planners (e.g. RRT [6] or PRM [3]) allows finding feasible trajectories in problems where no quasi-static trajectories are feasible [5].

In this paper, we further develop the method of [5] in order to provide a fast and general non-prehensile transportation scheme. More precisely, our contribution is twofold. First, in section II, we show how to cast the dynamic balance constraints of a 3D object (e.g. a bottle) into a form compatible with the AVP algorithm. Second, in section III, we extend the AVP-RRT algorithm of [5], which combines AVP with the uni-directional RRT [6], into a more efficient AVP-biRRT algorithm, which uses the idea of concurrently growing two trees, one rooted at the starting configuration and one rooted at the goal configuration. Then, in section IV, we show both in simulations and on a real robot how our algorithm allows planning fast and dynamic trajectories for the non-prehensile transportation of a bottle. Finally, we draw a brief conclusion in section V.

II. Derivation of constraints in the Bobrow form

Let $s : [0, T'] \to [0, T]$ be a path parameterization function which is increasing and twice-differentiable. A trajectory $\mathbf{q}(t) \in \mathbf{R}^n, t \in [0, T]$ can be written as a function of s as $\mathbf{q} = \mathbf{q}(s)$. Consequently, its time derivatives are

$$\dot{\mathbf{q}} = \mathbf{q}_s \dot{s} \tag{1}$$

and

$$\ddot{\mathbf{q}} = \mathbf{q}_s \ddot{s} + \mathbf{q}_s \dot{s}^2. \tag{2}$$

Next, the geometric and kinodynamic constraints of the robot can be expressed as functions of path parameter s. In

the sequel, an inequality constraint is said to be in the *Bobrow form* [7] if it is expressed as

$$\mathbf{a}(s)\ddot{s} + \mathbf{b}(s)\dot{s}^2 + \mathbf{c}(s) \leq 0. \tag{3}$$

Expressing constraints in the Bobrow form allows us to use the AVP algorithm, which is based itself on the Bobrow algorithm (see e.g. [4], [8]), in order to plan trajectories under kinodynamic constraints [5], [7].

A. Reducing the friction constraints to the Bobrow form

Consider a rectangular prism bottle of dimension $2d_x \times 2d_y \times 2h$. The position vector of the center of mass (COM) of the bottle, denoted by $\mathbf{p}_b$, with respect to inertial frame can be expressed as a function of the generalized coordinates $\mathbf{q}$ as $\mathbf{p}_b = \mathbf{r^p}(\mathbf{q})$. Differentiating $\mathbf{p}_b$ once and twice yields the expressions of velocity and acceleration of the COM as

$$\dot{\mathbf{p}}_b = \mathbf{r^p_q}\dot{\mathbf{q}} \tag{4}$$

and

$$\ddot{\mathbf{p}}_b = \mathbf{r^p_q}\ddot{\mathbf{q}} + \dot{\mathbf{q}}^T\mathbf{r^p_{qq}}\dot{\mathbf{q}}, \tag{5}$$

where $\mathbf{r^p_q} \in \mathbf{R}^{3\times n}$ and $\mathbf{r^p_{qq}} \in \mathbf{R}^{3\times n\times n}$ are the Jacobian matrix and the Hessian tensor of $\mathbf{r^p}$ with respect to $\mathbf{q}$. Plugging (1) and (2) into the expressions of $\dot{\mathbf{p}}_b$ and $\ddot{\mathbf{p}}_b$, these quantities can be written in terms of path parameter s, $\dot{s}$, and $\ddot{s}$ as

$$\dot{\mathbf{p}}_b = \mathbf{p}_{b_s}\dot{s} \tag{6}$$

and

$$\ddot{\mathbf{p}}_b = \mathbf{p}_{b_s}\ddot{s} + \mathbf{p}_{b_{ss}}\dot{s}^2, \tag{7}$$

where $\mathbf{p}_{b_s} = \mathbf{r^p_q}\mathbf{q}_s$ and $\mathbf{p}_{b_{ss}} = \mathbf{r^p_q}\mathbf{q}_{ss} + \mathbf{q}_s^T\mathbf{r^p_{qq}}\mathbf{q}_s$.

Next, consider Newton's second law of motion for the bottle

$$m_b\ddot{\mathbf{p}}_b = m_b\mathbf{g} + \mathbf{f} + \mathbf{N}, \tag{8}$$

where $\mathbf{f}$ is a friction force on the bottle-tray contact surface, $\mathbf{N}$ is a normal force, and $\mathbf{g}$ is the gravitational acceleration. Let $\mathbf{n}_z$ be a unit vector normal to the tray. The magnitude of the normal force, $N = \mathbf{n}_z^T\mathbf{N}$, is obtained as

$$N = m_b\mathbf{n}_z^T(\ddot{\mathbf{p}}_b - \mathbf{g}) \tag{9}$$

since $\mathbf{n}_z^T\mathbf{f} = 0$.

By using expressions in (5), (1), and (2), the non-negativity constraint on the normal force can then be expressed in the Bobrow form as

$$-N_s\ddot{s} - N_{ss}\dot{s} - N_0 \leq 0, \tag{10}$$

where $N_s = m_b\mathbf{n}_z^T\mathbf{p}_{b_s}$, $N_{ss} = m_b\mathbf{n}_z^T\mathbf{p}_{b_{ss}}$, and $N_0 = -m_b\mathbf{n}_z^T\mathbf{g}$.

Now, we turn to the expression of the friction force. From (8) and (9), the friction force can be written in terms of s, $\dot{s}$, and $\ddot{s}$ as

$$\mathbf{f} = \mathbf{f}_s\ddot{s} + \mathbf{f}_{ss}\dot{s}^2 + \mathbf{f}_0, \tag{11}$$

where $\mathbf{f}_s = m_b\mathbf{p}_{b_s} - N_s\mathbf{n}_z$, $\mathbf{f}_{ss} = m_b\mathbf{p}_{b_{ss}} - N_{ss}\mathbf{n}_z$, and $\mathbf{f}_0 = -m_b\mathbf{g} - N_0\mathbf{n}_z$.

However, the friction cone constraints $||\mathbf{f}|| \leq \mu N$ cannot be directly reduced to the Bobrow form due to its non-linearity in $\mathbf{f}$. Therefore, we use instead a more restrictive, linear version of the constraints where the friction cone is replaced by a friction pyramid (see e.g. [9]). The constraints become

$$-\frac{\mu}{\sqrt{2}}N \quad \leq \quad \mathbf{n}_x^T\mathbf{f} \quad \leq \quad \frac{\mu}{\sqrt{2}}N, \tag{12}$$

and

$$-\frac{\mu}{\sqrt{2}}N \quad \leq \quad \mathbf{n}_y^T\mathbf{f} \quad \leq \quad \frac{\mu}{\sqrt{2}}N, \tag{13}$$

where $\mathbf{n}_x$ and $\mathbf{n}_y$ are unit vectors in the directions of x- and y- axis of the reference frame of the tray respectively. Substituting (11) into inequalities (12) and (13) yields four friction constraints in the Bobrow form

$$(\mathbf{n}_x^T\mathbf{f}_s - \frac{\mu}{\sqrt{2}}N_s)\ddot{s} \;+\; (\mathbf{n}_x^T\mathbf{f}_{ss} - \frac{\mu}{\sqrt{2}}N_{ss})\dot{s}^2 + \;(\mathbf{n}_x^T\mathbf{f}_0 - \frac{\mu}{\sqrt{2}}N_0) \;\leq\; 0, \tag{14}$$

$$(-\mathbf{n}_x^T\mathbf{f}_s - \frac{\mu}{\sqrt{2}}N_s)\ddot{s} \;+\; (-\mathbf{n}_x^T\mathbf{f}_{ss} - \frac{\mu}{\sqrt{2}}N_{ss})\dot{s}^2 + \;(-\mathbf{n}_x^T\mathbf{f}_0 - \frac{\mu}{\sqrt{2}}N_0) \;\leq\; 0, \tag{15}$$

$$(\mathbf{n}_y^T\mathbf{f}_s - \frac{\mu}{\sqrt{2}}N_s)\ddot{s} \;+\; (\mathbf{n}_y^T\mathbf{f}_{ss} - \frac{\mu}{\sqrt{2}}N_{ss})\dot{s}^2 + \;(\mathbf{n}_y^T\mathbf{f}_0 - \frac{\mu}{\sqrt{2}}N_0) \;\leq\; 0, \tag{16}$$

and

$$(-\mathbf{n}_y^T\mathbf{f}_s - \frac{\mu}{\sqrt{2}}N_s)\ddot{s} \;+\; (-\mathbf{n}_y^T\mathbf{f}_{ss} - \frac{\mu}{\sqrt{2}}N_{ss})\dot{s}^2 + \;(-\mathbf{n}_y^T\mathbf{f}_0 - \frac{\mu}{\sqrt{2}}N_0) \;\leq\; 0. \tag{17}$$

B. Reducing the ZMP constraints to the Bobrow form

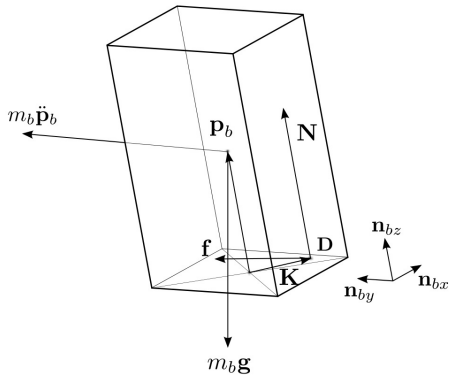

Fig. 1: Free body diagram of the bottle

Consider the total moment acting on the bottle about a point D on the surface of the tray

$$\mathbf{M}_b = \overline{\mathbf{Dp}_b} \times m_b(\mathbf{g} - \ddot{\mathbf{p}}_b) + \mathbf{M}, \tag{18}$$

where $\mathbf{M}$ is the time derivative of the angular momentum of the bottle. For the point $\mathbf{D}$ to be the ZMP, the total moment about $\mathbf{D}$ must have no tangential components, *i.e.*,

$$\mathbf{n}_z \times \mathbf{M}_b = 0. \tag{19}$$

Substituting (18) into (19) and applying the identity $\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) = (\mathbf{a}^T\mathbf{c})\mathbf{b} - (\mathbf{a}^T\mathbf{b})\mathbf{c}$ bring us to

$$\overline{\mathbf{Dp}_b} = \frac{1}{N}\left(m_b h \ddot{\mathbf{p}}_b + (\mathbf{n}_z \times \mathbf{M}) - m_b h \mathbf{g}\right), \tag{20}$$

Let $\mathbf{n}_{bx}$, $\mathbf{n}_{by}$ and $\mathbf{n}_{bz}$ be unit vectors in the direction of x-, y- and z- axes of the reference frame of the bottle respectively. Define $\mathbf{K} = h\mathbf{n}_z - \overline{\mathbf{Dp}_b}$ as a vector pointing from the center of the supporting area of the bottle to the ZMP (see Fig. 1). By using the expression of $\overline{\mathbf{Dp}_b}$ in (20) and the fact that $\mathbf{M}$ can be written as $\mathbf{M} = \mathbf{M}_s\ddot{s} + \mathbf{M}_{ss}\dot{s}^2$, one obtains the vector $\mathbf{K}$ in terms of s and its derivatives as

$$\mathbf{K} = \frac{\mathbf{K}_s\ddot{s} + \mathbf{K}_{ss}\dot{s}^2 + \mathbf{K}_0}{N_s\ddot{s} + N_{ss}\dot{s}^2 + N_0}, \tag{21}$$

where

$$\begin{aligned}
\mathbf{K}_s &= hN_s\mathbf{n}_{bz} - m_b h\mathbf{p}_{bs} - \mathbf{n}_{bz} \times \mathbf{M}_s, \\
\mathbf{K}_{ss} &= hN_{ss}\mathbf{n}_{bz} - m_b h\mathbf{p}_{bss} - \mathbf{n}_{bz} \times \mathbf{M}_{ss}, \text{ and} \\
\mathbf{K}_0 &= hN_0\mathbf{n}_{bz} + m_b h\mathbf{g}.
\end{aligned}$$

The constraints for the ZMP to lie inside the supporting area can be expressed as

$$-d_x \quad \leq \quad \mathbf{n}_{bx}^T\mathbf{K} \quad \leq \quad d_x \tag{22}$$

and

$$-d_y \quad \leq \quad \mathbf{n}_{by}^T\mathbf{K} \quad \leq \quad d_y. \tag{23}$$

Substituting (21) into inequalities (22) and (23) yields four ZMP constraints in the Bobrow form

$$\begin{aligned}
(\mathbf{n}_{bx}^T\mathbf{K}_s - d_x N_s)\ddot{s} \;+\; (\mathbf{n}_{bx}^T\mathbf{K}_{ss} - d_x N_{ss})\dot{s}^2 \\
+\; (\mathbf{n}_{bx}^T\mathbf{K}_0 - d_x N_0) \;\leq\; 0,
\end{aligned} \tag{24}$$

$$\begin{aligned}
(-\mathbf{n}_{bx}^T\mathbf{K}_s - d_x N_s)\ddot{s} \;+\; (-\mathbf{n}_{bx}^T\mathbf{K}_{ss} - d_x N_{ss})\dot{s}^2 \\
+\; (-\mathbf{n}_{bx}^T\mathbf{K}_0 - d_x N_0) \;\leq\; 0,
\end{aligned} \tag{25}$$

$$\begin{aligned}
(\mathbf{n}_{by}^T\mathbf{K}_s - d_y N_s)\ddot{s} \;+\; (\mathbf{n}_{by}^T\mathbf{K}_{ss} - d_y N_{ss})\dot{s}^2 \\
+\; (\mathbf{n}_{by}^T\mathbf{K}_0 - d_y N_0) \;\leq\; 0,
\end{aligned} \tag{26}$$

and

$$\begin{aligned}
(-\mathbf{n}_{by}^T\mathbf{K}_s - d_y N_s)\ddot{s} \;+\; (-\mathbf{n}_{by}^T\mathbf{K}_{ss} - d_y N_{ss})\dot{s}^2 \\
+\; (-\mathbf{n}_{by}^T\mathbf{K}_0 - d_y N_0) \;\leq\; 0.
\end{aligned} \tag{27}$$

C. Remarks

1) Shape of the bottle: For a bottle with arbitrary bottom shape, we can use the approximated model which has rectangular bottom instead.

2) Non-negativity of the normal force: Note that the inequality (10) is actually unnecessary. Summing two inequalities in (12) or (13) leads us to

$$0 \leq \sqrt{2}\mu N. \tag{28}$$

Therefore, if friction constraints are satisfied, (10) is automatically satisfied. Nevertheless, for the case when safety is a main concern, we may modify (10) to be

$$-N_s\ddot{s} - N_{ss}\dot{s} - N_0 + \epsilon \leq 0, \tag{29}$$

where ϵ is a small positive real number.

III. AVP-BIRRT

A. Main algorithm

The AVP algorithm developed in [5] allows extending the decoupling approach (first, plan a path considering geometric constraints, and second, time-parameterize that path considering kinodynamic constraints) to the *incremental* setting of the highly efficient sampling-based algorithms such as RRT and PRM.

However, in [5], the AVP algorithm was combined with the unidirectional RRT, which grows a single tree rooted at the starting configuration. BiRRT [2], by growing two trees, one rooted at the starting configuration and one rooted at the goal configuration, allows searching the robot joint space more thoroughly. Furthermore, using two trees will increase the probability of finding a path going through a narrow passage [10]. This will be particularly helpful in the cases when feasible quasi-static trajectories are difficult to find or do not exist at all.

We developed AVP_BACKWARD, which allows propagating admissible velocity intervals *backward* in time (note that this is not a trivial extension of AVP forward since the dynamic equations are not in general time-reversible). This allows us to obtain a bi-directional version of AVP-RRT introduced in [5]. The new algorithm, AVP-biRRT, is given in Algorithm 1.

Algorithm 1: AVP-biRRT

input : $\mathbf{q}_{\text{start}}$, $\mathbf{q}_{\text{goal}}$, $\dot{\mathbf{q}}_{\text{start}}$, $\dot{\mathbf{q}}_{\text{goal}}$, Bobrow form constraints
output: a trajectory

$\mathcal{U}_{\text{start}} \leftarrow$ VERTEX($\mathbf{q}_{\text{start}}$, $\dot{\mathbf{q}}_{\text{start}}$)
$\mathcal{U}_{\text{goal}} \leftarrow$ VERTEX($\mathbf{q}_{\text{goal}}$, $\dot{\mathbf{q}}_{\text{goal}}$)
$\mathcal{T}_0$.INITIALIZE($\mathcal{U}_{\text{start}}$)
$\mathcal{T}_1$.INITIALIZE($\mathcal{U}_{\text{goal}}$)
for $i = 1$ **to** N_{max} **do**
  $\mathcal{T}_{\text{start}} \leftarrow \mathcal{T}_{\text{mod}(i-1,2)}$
  $\mathcal{T}_{\text{end}} \leftarrow \mathcal{T}_{\text{mod}(i,2)}$
  $(\mathbf{q}_{\text{rand}}, \dot{\mathbf{q}}_{\text{rand}}) \leftarrow$ RANDOM_STATE()
  $\mathcal{U}_{\text{new}} \leftarrow$ EXTEND($\mathcal{T}_{\text{start}}$, $(\mathbf{q}_{\text{rand}}, \dot{\mathbf{q}}_{\text{rand}})$)
  if *EXTEND succeeds* **then**
    if *CONNECT($\mathcal{T}_{start}$, $\mathcal{T}_{end}$) succeeds* **then**
      return *COMPUTE_TRAJ($\mathcal{T}_{start}$, $\mathcal{T}_{end}$)*
    end
  end
end
return *failure*

A vertex $\mathcal{U}$ stored in a tree contains 4 entities which are state, vmin, vmax, and path. $\mathcal{U}$.state is a pair of a configuration and its time derivative. $\mathcal{U}$.vmin and $\mathcal{U}$.vmax indicate the reachable velocity interval from its parent. $\mathcal{U}$.path stores a path to $\mathcal{U}$ from its parent in the tree rooted at the starting configuration and stores a path from $\mathcal{U}$ to its parent in the tree rooted at the goal configuration.

In each iteration, the algorithm chooses a tree to be extended. Then, it tries to extend the tree to the newly sampled state using the EXTEND routine. If the tree is successfully extended, AVP-biRRT then tries to connect two trees together. The algorithm will terminate when either two trees are connected or the max number of iterations is exceeded.

The routine EXTEND called in the AVP-biRRT algorithm is described in (Algorithm 2).

Algorithm 2: EXTEND($\mathcal{T}$, $(\mathbf{q}_{\text{rand}}, \dot{\mathbf{q}}_{\text{rand}})$)

```
for j = 1 to n do
    U_near ← NEAREST(T, (q_rand, q̇_rand), j)
    P ← INTERPOLATE(U_near, (q_rand, q̇_rand))
    if T is T_0 then
        (v_min, v_max) ← AVP_FORWARD(P, U_near.vmin, U_near.vmax)
    else
        (v_min, v_max) ← AVP_BACKWARD(P, U_near.vmin, U_near.vmax)
    end
    if AVP succeeds then
        if P is collision-free then
            U_new ← VERTEX((q_rand, q̇_rand))
            U_new.path ← P
            U_new.vmin ← v_min
            U_new.vmax ← v_max
            T.ADD_VERTEX(U_new)
            return U_new
        end
    end
end
```

In the j^{th} iteration of EXTEND, the routine NEAREST returns the j^{th} nearest neighbor of the state $(\mathbf{q}_{\text{rand}}, \dot{\mathbf{q}}_{\text{rand}})$ from the tree $\mathcal{T}$. Then, INTERPOLATE interpolates a path P between $\mathcal{U}$.state and $(\mathbf{q}_{\text{rand}}, \dot{\mathbf{q}}_{\text{rand}})$.

AVP_FORWARD computes the reachable velocity interval when the path P is traversed from any initial velocity in $[\mathcal{U}.\text{vmin}, \mathcal{U}.\text{vmax}]$ (cf. [5] for more detail). In contrast, AVP_BACKWARD computes possible initial velocity interval $\mathcal{V}$ such that when start traversing the path P from any velocity $v \in \mathcal{V}$, the final velocity will always be in the interval $[\mathcal{U}.\text{vmin}, \mathcal{U}.\text{vmax}]$.

Finally, when the two trees can be connected, COMPUTE_TRAJ will generate a new path by concatenating paths starting from $(\mathbf{q}_{\text{start}}, \dot{\mathbf{q}}_{\text{start}})$ to $(\mathbf{q}_{\text{goal}}, \dot{\mathbf{q}}_{\text{goal}})$ and reparameterize it using our library on time-optimal path parameterization. The library is open-source and available at http://www.ntu.edu.sg/home/cuong/software.html.

B. Implementation details

1) More constraints: Apart from the friction and ZMP constraints described above, we also included the joint acceleration bounds, which can be derived easily from (2), into our implementation. In fact, other constraints such as actuator torque bounds can be taken into account also. Note also that we can also consider more than one bottle at a time by taking into account another set of constraints from another bottle.

2) Path interpolation: In the INTERPOLATE subroutine, we choose to interpolate a path between the given states with a fifth degree polynomial. This makes the final trajectory output from AVP-biRRT having a continuous acceleration profile. Therefore, the robot's motion will be smoother. The real robot can then execute this trajectory more precisely than a trajectory only a piecewise continuous acceleration profile.

3) Dynamic shortcutting: After running AVP-biRRT, there will be some unnecessary motions generated by the algorithm. Therefore, it is preferable to shortcut the trajectory to reduce such motions. We use the minimum-time shortcut algorithm presented in [11] which also takes into account the dynamic constraints.

IV. Simulations and Experiments

A. Simulations

Simulations were performed using OpenRAVE [12] on a 3.2 GHz Intel® Core™with 3.8 GB RAM. The robot manipulator model used was DENSO VS-060 with a tray mounted as its end effector. The robot was to carry a bottle on the tray throughout a window in the wall.

The dimensions of the bottle were $d_x = d_y = 2.1$ cm. and $h = 10$ (cm.). The static friction coefficient μ was set to 0.27.

Using AVP-biRRT and dynamics shortcutting, we found a feasible trajectory of duration 4.0 seconds.

The maximum velocity curves and the velocity profiles for this trajectory are shown in Fig. 3 (for the definition of these profiles, see [5]). Profiles of the normal force, friction force, and ZMP positions for this trajectory are given in Fig. 4. For comparison, the profiles of the normal force, friction force, and ZMP positions for the same trajectory but executed quasi-statically are given in Fig. 5.

The position of the ZMP and COM (note that COM=ZMP at quasi-static regime) of the bottle relative to its support area are also shown in Fig. 6. As we can see from Fig. 5 and Fig. 6(b), both friction and ZMP profiles went beyond their bounds at time instants around $t = 1.0$ s. to $t = 1.5$ s. This suggests that the trajectory is quasi-statically infeasible.

B. Experiments on the DENSO VS-060 robot

We tested trajectories generated by AVP-biRRT on the actual DENSO VS-060 robot. A video of the robot executing this dynamic bottle transportation motion can be found at http://youtu.be/6KCv6EM9Mf8. In the video, one can see that the bottle can be maintained on the tray when the motion is executed at full speed, but that it slips from the tray when the motion is executed at 0.1x speed, hinting again that the quasi-static trajectory is *not* feasible.

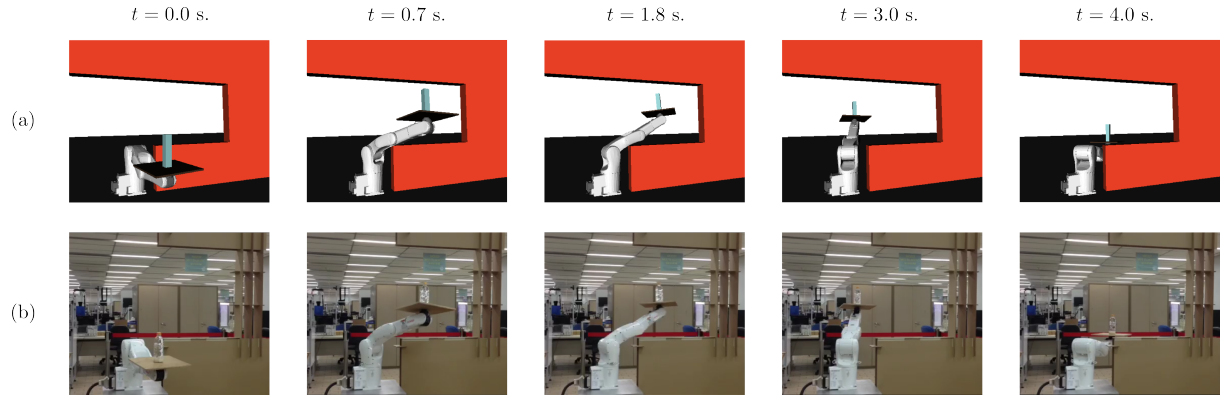

Fig. 2: Snapshots from a computer simulation (a) and an experiment (b). We can see from row (b) that the bottle did not slide on the tray nor fall down since no constraint was violated. The robot successfully transported the bottle to the goal configuration.

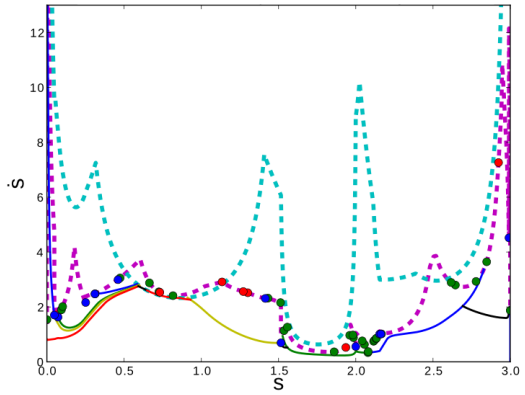

Fig. 3: Maximum velocity curves and profiles. The dotted blue line represents the saturated velocity bound of the robot. The dotted magenta line represents the Maximum Velocity Curves (MVC). These curves represent minimum velocity bounds resulting from problem constraints at each s. Velocity profiles (solid lines) must stay below these curves otherwise some constraints will be violated.

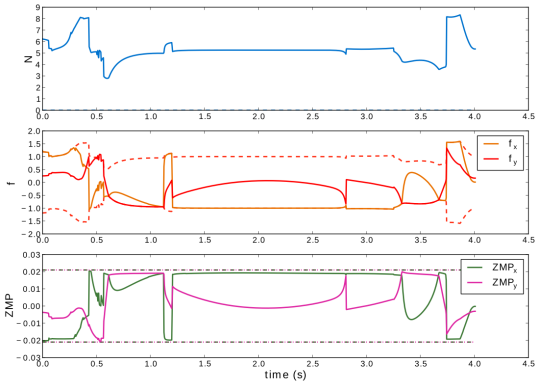

Fig. 4: Dynamic constraint profiles. The top subfigure shows the magnitude of the normal force N. The middle subfigure shows 2 components of friction force in the direction of $\mathbf{n}_x$ (orange) and $\mathbf{n}_y$ (red). The last subfigure shows ZMP trajectories in $x-$ and $y-$ directions. Note that at each time instance, at least one constraint was saturated. This follows from the Pontryagin Maximum Principle.

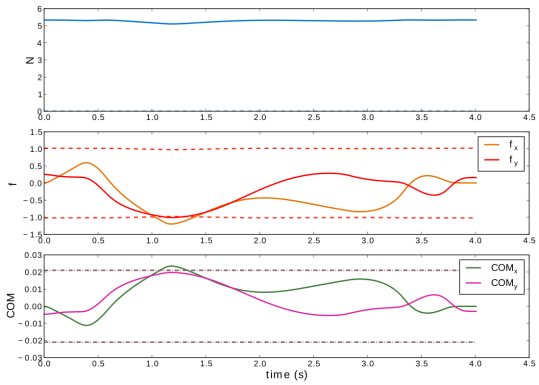

Fig. 5: Quasi-static constraint profiles resulting from executing the trajectory with very low speed such that there is no dynamic effect on the bottle. The top subfigure shows the magnitude of the normal force N. The middle subfigure shows 2 components of friction force in the direction of $\mathbf{n}_x$ (orange) and $\mathbf{n}_y$ (red). The x-component (orange) went below its lower bound just after $t = 1$s. The last subfigure shows ZMP trajectories, which coincide with COM, in $x-$ and $y-$ directions.

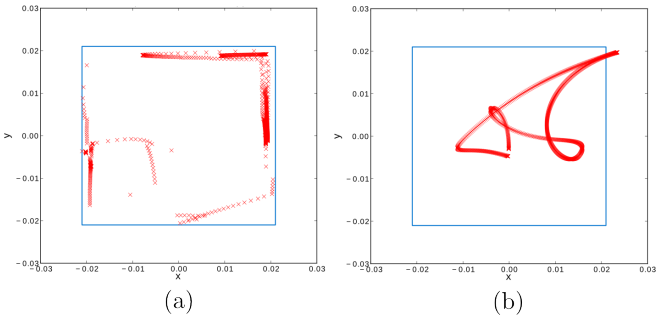

Fig. 6: (a) ZMP trajectory relative to the support area of the bottle. (b) COM trajectory relative to the support area of the bottle (note that at quasi-static regime, COM=ZMP). The COM went outside the support area shown in blue hinting at the infeasibility of the quasi-static trajectory.

V. Conclusion

We have presented a fast and general non-prehensile transportation scheme. Dynamic balance constraints for a 3D object was derived in the Bobrow form which is compatible with the AVP algorithm. Our developed algorithm called AVP-BiRRT, based on the AVP algorithm, is also described in detail. The results presented in this paper show that our algorithm is able to plan non-quasi-static trajectories subject to kinodynamic constraints while staying in the robot joint space. Therefore, this suggests the usefulness of algorithm especially in cases where a quasi-static trajectory is difficult to find or does not exist at all.

References

[1] B. Donald, P. Xavier, J. Canny, and J. Reif, "Kinodynamic motion planning," *Journal of the ACM (JACM)*, vol. 40, no. 5, pp. 1048–1066, 1993.

[2] S. LaValle and J. Kuffner, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[3] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, "Randomized kinodynamic motion planning with moving obstacles," *The International Journal of Robotics Research*, vol. 21, no. 3, pp. 233–255, 2002.

[4] J. Bobrow, S. Dubowsky, and J. Gibson, "Time-optimal control of robotic manipulators along specified paths," *The International Journal of Robotics Research*, vol. 4, no. 3, pp. 3–17, 1985.

[5] Q.-C. Pham, S. Caron, and Y. Nakamura, "Kinodynamic planning in the configuration space via velocity interval propagation," in *Robotics: Science and System*, 2013.

[6] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *IEEE International Conference on Robotics and Automation*, 2000.

[7] Q.-C. Pham, "A general, fast, and robust implementation of the time-optimal path parameterization algorithm," *CoRR*, vol. abs/1312.6533, 2013.

[8] Z. Shiller and H. Lu, "Computation of path constrained time optimal motions with dynamic singularities," *Journal of dynamic systems, measurement, and control*, vol. 114, p. 34, 1992.

[9] J. Kerr and B. Roth, "Analysis of multifingered hands," *The International Journal of Robotics Research*, vol. 4, no. 4, pp. 3–17, 1986.

[10] D. Hsu, J.-C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," in *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, vol. 3. IEEE, 1997, pp. 2719–2726.

[11] Q.-C. Pham, "Planning manipulator trajectories under dynamics constraints using minimum-time shortcuts," in *Second IFToMM ASIAN Conference on Mechanism and Machine Science*, 2012.

[12] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. dissertation, Carnegie Mellon University, Robotics Institute, August 2010. [Online]. Available: http://www.programmingvision.com/rosen_diankov_thesis.pdf