HAL
open science

Real-time footstep planning for humanoid robots among 3D obstacles using a hybrid bounding box

Nicolas Perrin, Olivier Stasse, Florent Lamiraux, Youn J. Kim, Dinesh Manocha

To cite this version:

Nicolas Perrin, Olivier Stasse, Florent Lamiraux, Youn J. Kim, Dinesh Manocha. Real-time footstep planning for humanoid robots among 3D obstacles using a hybrid bounding box. IEEE International Conference on Robotics and Automation, May 2012, Saint Paul, Minnesota, United States. pp.0-0. hal-00668794

HAL Id: hal-00668794

https://hal.science/hal-00668794v1

Submitted on 10 Feb 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Real-time footstep planning for humanoid robots among 3D obstacles using a hybrid bounding box

Nicolas Perrin and Olivier Stasse and Florent Lamiraux and Young J. Kim and Dinesh Manocha

Abstract— **In this paper we introduce a new bounding box method for footstep planning for humanoid robots. Similar to the classic bounding box method (which uses a single rectangular box to encompass the robot) it is computationally efficient, easy to implement and can be combined with any rigid body motion planning library. However, unlike the classic bounding box method, our method takes into account the stepping over capabilities of the robot, and generates precise leg trajectories to avoid obstacles on the ground. We demonstrate that this method is well suited for footstep planning in cluttered environments.**

I. Introduction

Humanoid robots tend to have a hybrid behavior: they combine continuous motions and discrete sequences of contacts. A key issue with respect to humanoid robots is footstep planning, whose main goal is to plan a discrete sequence of footsteps as well as the corresponding continuous motion of the robot. This can be regarded as a hybrid motion planning problem. The standard way to perform footstep planning is to plan a sequence of inputs, i.e. step parameters such as relative position and orientation of the feet, and send them to a walking pattern generator. The usual approach is to decide on a small finite set of potential steps, called the action set. Then, motion planning is performed using the A* search algorithm on this action set (see [12], [3], [4], [5], [8]). Many ad hoc methods have been proposed to extend this approach and perform better footstep planning, such as local footstep adjustments ([6]) or human-like strategies ([2]). However, due to the discrete aspect of footstep planning, the state-of-the-art algorithms for rigid body motion planning are hard to apply; for example, many challenges in terms of applying sampling-based algorithms to discrete search spaces are known [14]. The easiest way to apply these algorithms to footstep planning is based on the bounding box method (here we are talking not about a hierarchical, but a single oriented bounding box; see [21]). After planning a continuous motion for the bounding box, we can plan a sequence of steps that follow the continuous trajectory. Since the robot is required to stay inside the volume swept by the bounding box, a collision-free trajectory of the bounding box implies a collision-free path for the robot. The main problem with this method is that the bounding box must circumvent all the obstacles on the floor, even the smallest ones that could easily be stepped over by the humanoid robot. As a result, it fails to make use of the stepping over capabilities of humanoid robots, and is thus better suited for robots with no legs such as PR2 or Robonaut.

In this paper we introduce a new bounding box method which can be used with continuous rigid body motion planners while still taking advantage of the stepping over capabilities of humanoid robots. This method is based on our previous work which introduced an equivalence between "weakly collision-free" continuous trajectories and discrete sequences of collision-free footsteps (but only 2D obstacles were considered in that work, see [17]).

Here are the two main contributions of this paper:

1) We introduce a new bounding box representation that captures the hybrid behavior of humanoid robots. Our representation is composed of two parts and three rectangular boxes: the upper part (one rectangular box) moves continuously while the two rectangular boxes of the lower part move discretely, i.e. abruptly from one configuration to another. By combining this hybrid bounding box motion with a specific walking pattern generator that can perform online trajectory smoothing, we show that the initial use of simple boxes for collision avoidance does not prevent us from generating precise stepping over motions at execution time.

2) We use the equivalence theorem in [17] to design a practical algorithm and use it to compute the motion of our bounding box representation (i.e. continuous *and* discrete parts) with continuous paths that can be easily produced by state-of-the-art algorithms for rigid body motion planning. This enables us to apply classical sampling-based algorithms to footstep planning with a better theoretical foundation as compared to prior methods (e.g. [20]). Our novel contribution is in terms of combining our hybrid bounding box representation, the specific walking pattern generator and the equivalence theorem in [17] to quickly compute dynamic walking trajectories in cluttered environments where stepping over obstacles is required.

The rest of the paper is organized as follows: in section II, we first explain how the bounding box is structured and briefly present the different phases of our algorithm. Then, we show a preliminary example which demonstrates the relationship between continuous paths and discrete motions. We continue by describing the algorithm that converts continuous paths into sequences of steps, and finally talk about the smoothing phase. In section III, we describe our implemen-

N. Perrin is with Department of Advanced Robotics, Istituto Italiano di Tecnologia, via Morego, 30, 16163 Genova, Italy nicolas.perrin@iit.it

O. Stasse and F. Lamiraux are with CNRS/LAAS, Université de Toulouse UPS, INSA, INP, ISAE, Toulouse, France {ostasse, florent}@laas.fr

Y. J. Kim is with Dept of Comp. Science and Engineering, Ewha Womans University, Seoul, Korea kimy@ewha.ac.kr

D. Manocha is with Dept of Comp. Science, University of North Carolina, Chapel Hill, U.S.A. dm@cs.unc.edu

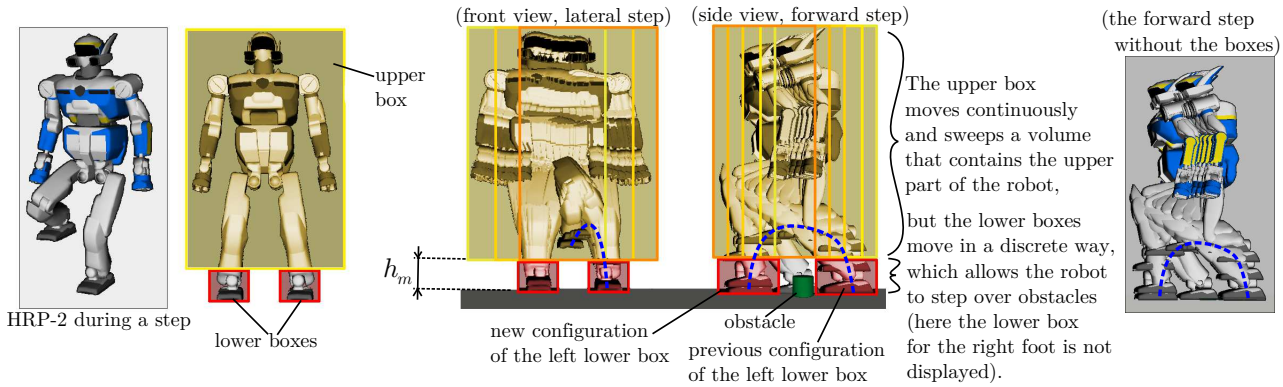

Fig. 1. HRP-2 and the improved bounding box, made of three rectangular boxes. The width and length of the lower boxes are 18.5cm and 28cm, that is to say both about 5cm greater than the corresponding dimension of the foot of HRP-2. The upper box width and length are respectively 95cm and 79cm.

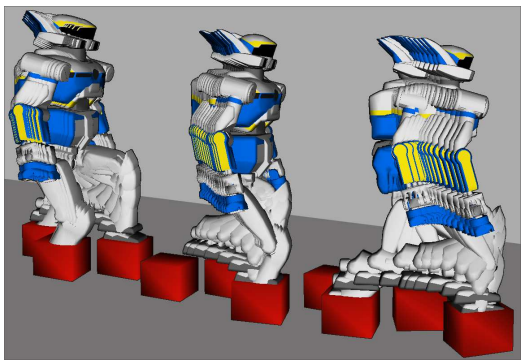

Fig. 2. A sequence of steps performed by HRP-2, with the corresponding configurations of the lower boxes. We can see that the feet of the robot always leave and enter the rectangular boxes from above, not from the sides. Thus, for all the obstacles whose height is less than h_m, the lower boxes can be used for conservative collision checks.

tation which uses the library OMPL for motion planning, the library PQP [13] for collision checks, and threads to perform both online conversion (from the bounding box trajectories to footsteps) and trajectory smoothing. Using OMPL, we compare several motion planning algorithms such as KPIECE [19], RRT-Connect [11], PRM [7], SBL [18], and show that real-time footstep planning among 3D obstacles can be obtained. Finally, in section IV, we discuss our conclusions and future work.

II. A TWO-LEVEL BOUNDING BOX

The bounding box used to conveniently plan the walking motions of a humanoid robot is usually a single rigid body. For instance, it can be a rectangular box that always contains the whole robot or at least sweeps volumes that entirely contain the robot motions (see [21], [10]). A cylinder is also sometimes used, for example when motion planning methods for differential drive robots are applied (see [9]). Bounding boxes are also sometimes made of several solid objects rigidly linked (see [8]).

Our improved bounding box behaves as a single box, but in fact, has three parts: two lower rectangular boxes, and one upper rectangular box. The upper box is similar to a classical bounding box in the sense that it moves continuously. The volume swept by the upper box contains every part of the robot above a fixed height h_m, which corresponds to the maximum height of the obstacles that can be stepped over. h_m is determined experimentally and depends on the capability of the robot and the walking pattern generator to raise the swing foot high enough above the ground during steps. For the robot HRP-2, this height was chosen to be 20cm, which means that the robot will circumvent all the obstacles of height greater than 20cm. Because of the similarity with the classical bounding box, the upper box is of secondary importance in this paper. Instead, we focus on the two lower boxes, which are used to capture the stepping over capabilities of the robot. Fig. 1 shows how the discrete motion of the lower boxes enables the robot to step over obstacles (without leaving the volume defined by the union of the boxes).

But in order to use classical motion planning techniques for our two-level bounding box, even the discrete motion of the lower boxes must be governed by continuous paths. In [17], we presented a reduction theorem that shows an equivalence between 2D footstep planning and more classical continuous motion planning problems. However, only 2D obstacles were considered, the feet of the robot were supposed circular, and no algorithm was given for an efficient reduction from the continuous path to a finite sequence of footsteps. In this section we present a method inspired from this reduction theorem to actually obtain a footstep planning algorithm for the robot HRP-2. To do so we will benefit from the features of an improved version of the specific walking pattern generator introduced in [16] (which has been extensively tested on the real HRP-2 robot) that provides online trajectory smoothing.

The problem is to make the humanoid robot walk from an initial location A to a goal B on a flat ground cluttered with 3D obstacles. The algorithm is divided into 3 phases:

1) Solve a classical rigid body motion planning problem to obtain some continuous path which is collision-free in some sense (as we will see below, “weakly collision-free”).

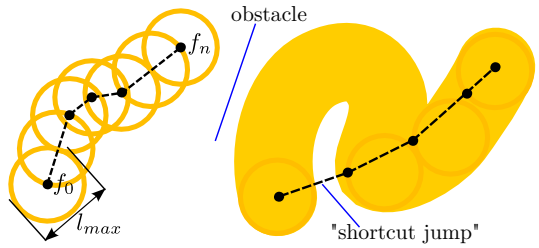

Fig. 3. The "flea motion planning problem". On the left: from a collision-free sequence of flea jumps to a continuous weakly collision-free path for the disk. On the right: converting a continuous weakly collision-free path of the disk into a sequence of flea jumps, using a greedy algorithm.

2) Convert the continuous path into a hybrid motion of the two-level bounding box. The bounding box motion directly corresponds to slow (but not statically stable) and large steps avoiding the obstacles, and we attempt to minimize the number of steps.
3) Keeping the same footprints, smooth and speed up the steps in order to reach the goal faster and avoid the obstacles in a more efficient way.

These 3 phases are described in the 3 next sections.

A. A continuous motion planning problem

Let us first consider a preliminary example that will show an equivalence between sequences of discrete displacements and continuous paths. This example is a problem of motion planning for a flea moving on a flat 2D surface with polygonal obstacles. This flea (represented by a point) can make jumps in any direction and of any length comprised between 0 and l_{max}. The goal is to find a sequence of jumps from a location $A(x_A, y_A)$ to a location $B(x_B, y_B)$ while avoiding the obstacles.

A sequence of jumps can be described by the sequence of configurations of the flea on the 2D surface (i.e. elements of $\mathbb{R}^2$). Let us assume that a sequence of jumps has been found, and that it corresponds to the sequence of configurations $f_0 = (x_0, y_0), f_1, f_2, \ldots, f_n = (x_n, y_n)$, with $(x_0, y_0) = (x_A, y_A)$ and $(x_n, y_n) = (x_B, y_B)$. We consider the continuous motion of a disk of diameter l_{max} as depicted on Fig. 3 (on the left): it starts in position (x_0, y_0), ends in position (x_n, y_n), and between two consecutive configurations (x_i, y_i) and (x_{i+1}, y_{i+1}), it moves straight from (x_i, y_i) to (x_{i+1}, y_{i+1}). A time parametrization of this motion gives us a continuous path $(s(t))_{t\in[0,1]} \in (\mathbb{R}^2)^{[0,1]}$. Let us define a new notion of collision-freeness for the disk:

Definition 1: We say that a disk configuration (x, y) is "weakly collision-free" if there exists at least one flea configuration (i.e. point) inside the disk which is collision-free. Conversely, if all the flea configurations inside the disk are in collision (i.e. the disk is entirely covered by the obstacles), we say that the disk is in "strong collision". We say that a continuous path is weakly collision-free if all the configurations along the path are weakly collision-free.

The continuous disk motion $(s(t))_{t\in[0,1]}$ previously defined has the following property (which is a direct consequence of the upper bound l_{max} on the length of jumps):

Property 1: For all $t \in [0, 1]$, the configuration $s(t)$ of the disk contains at least one of the flea configurations $f_0, f_1, \ldots, f_n$. Therefore the continuous disk motion $(s(t))_{t\in[0,1]}$ is weakly collision-free.

What's even more interesting than Property 1 is that the converse property is true:

Property 2: If there exists a weakly collision-free continuous path of the disk from A to B, then there exists a discrete sequence of jumps from A to B.

Together, Property 1 and Property 2 form an equivalence between the weakly collision-free paths of the disk and the collision-free sequences of jumps. Thus, in order to look for a discrete sequence of jumps, we can first look for a continuous path of the disk, which can be done with any classical rigid body motion planning algorithm (we just have to implement new collision checks using Definition 1).

For footstep planning, the principle is almost exactly the same, but instead of the discrete motion of a point (or flea), we need to consider the discrete motion of two feet, which is a bit more complex. This is the purpose of [17], which establishes a similar equivalence for a simple case of footstep planning. In the present paper we turn this equivalence result into a practical algorithm that first finds some (weakly collision-free) continuous path using a sampling-based planning algorithm, and then converts it into a) a continuous motion for the upper box, and b) a sequence of discrete (collision-free) displacements for the lower boxes.

Without going into details, let us exhibit some of the differences between the "flea example" and footstep planning. First of all, the configuration space changes, and we use $(x, y, \theta) \in \mathbb{R}^2 \times SO(2)$ to denote the position and orientation of the humanoid robot, or to be more precise the position and orientation of the upper bounding box. We denote by $(c_L, c_R) \in (\mathbb{R}^2 \times SO(2))^2$ a double support configuration of the robot, i.e. the configurations of the left and right foot (actually the left and right lower box) on the ground. The object for which we must find a weakly collision-free path is a set of such double support configurations that depends on the position and orientation of the robot. If we denote this set by $\Phi(x, y, \theta)$ we have $\Phi(x, y, \theta) \subset (\mathbb{R}^2 \times SO(2))^2$ (of course all the double support configurations in Φ must be feasible, and Φ is a more complex set than a disk), and $\Phi(x, y, \theta) = \vec{T}_{(x,y)} \circ Rot_\theta(\Phi(0, 0, 0))$ where $\vec{T}_{(x,y)}$ and Rot_θ respectively denote the translation of vector (x, y) and rotation of angle θ (of subsets of $(\mathbb{R}^2 \times SO(2))^2$) about the origin. To reduce the complexity of weak collision checks, we choose to assume that $\Phi(x, y, \theta)$ is the cartesian product of two sets, one being possible configurations of the left foot ($\Phi_L(x, y, \theta)$), and the other one being possible configurations of the right foot ($\Phi_R(x, y, \theta)$): $\Phi(x, y, \theta) = \Phi_L(x, y, \theta) \times \Phi_R(x, y, \theta)$. This property implies the independence between Φ_L and Φ_R in the following definition:

Definition 2: $\Phi(x, y, \theta)$ is said weakly collision-free if and only if there exists a collision-free configuration of the left lower box in $\Phi_L(x, y, \theta)$, *and* there exists a collision-free configuration of the right lower box in $\Phi_R(x, y, \theta)$, *and* the

additional condition is verified: the configuration (x, y, θ) of the upper box is collision-free.

Because of the restriction on Φ which must be a cartesian product, it is impossible to accurately approximate the stepping capabilities of the robot with only one fixed Φ object. Thus we add to our configuration space a new dimension used to continuously deform Φ_L and Φ_R. We now denote the configurations of Φ by $\Phi(x, y, \theta, \delta)$, and the new configuration space is $\mathbb{R}^2 \times SO(2) \times (-1, 1)$ (see [17] for more details) which is still perfectly suitable for sampling-based planning algorithms. Like in [17], Φ_L and Φ_R can be represented by portions of disks of varying size: see Fig. 5 and 6.

B. Reduction to a finite sequence of steps

Now we assume that a continuous weakly collision-free path has been found for Φ, and we show how to convert it into a hybrid motion of the rectangular boxes. First, let us consider again the "flea example". We denote by $(s(t))_{t\in[0,1]} \in (\mathbb{R}^2)^{[0,1]}$ the solution path for the disk. When we convert it into a sequence of jumps, a natural objective is to try to minimize the number of jumps. To do so, we use the following greedy algorithm: assuming that the current configuration of the flea (x_0, y_0) is inside the disk of configuration $s(t_0)$, we search for the largest $t_{new} > t_0$ such that it is possible to jump from (x_0, y_0) into the disk of configuration $s(t_{new})$. Fig. 3 (on the right) depicts the result of this greedy algorithm, which can be straightforwardly adapted to our "hybrid bounding box planning problem" and implemented with a dichotomy (see Algorithm 1 which sets the next *left* step), but attention must be paid to one important detail related to the upper box motion. Basically, during the planning phase we continuously move the upper box along the path $s(t)$, and we use classical collision checks to determine the validity of the path (the weak collision checks are for Φ, i.e. the lower boxes). However, in order to enable the kind of shortcut shown on Fig. 3, some non-trivial properties of the upper box must be verified. Roughly, the union of the first and last configurations of the upper box during a step must be enough to encompass the whole motion of the upper part of the robot (i.e. everything above height h_m) during this step. In our current implementation we chose an upper box big enough to empirically verify this property, but in future work we will consider automated procedures to tune the upper box geometry in order to verify the required properties while not being overly conservative.

Another detail must be carefully addressed: if the continuous path $(s(t))_{t\in[0,1]}$ is indeed made of weakly collision-free configurations, then the next step can always be found. However planners usually decide the validity of a whole continuous path based upon the validity of a discretization of this path. Therefore the solution path may not be entirely weakly collision-free, and in that case Algorithm 1 might not find a next step. Although there are several techniques to avoid that, one convenient solution is to use a margin for the collision checks when the continuous weakly collision-free solution path is searched for, but no margin (or a smaller one) when the finite sequence of lower box configurations is constructed.

Algorithm 1 SETNEXTLEFTSTEP($(t,\ (c_L, c_R))$)

Require: A solution path $(s(t))_{t\in[0,1]}$
Require: An upper bound on $t_{new} - t$: Δ_{max}
Require: A precision parameter $\epsilon > 0$

$t_{min} \leftarrow t$
$t_{max} \leftarrow \min(1,\ t + \Delta_{max})$
while `true` **do**
  $t_{new} \leftarrow \frac{t_{min}+t_{max}}{2}$
  if $c_R \in \Phi_R(s(t_{new}))$ **then**
    Try to find (through sampling) a *feasible* left step such that the next double support configuration (c_{new}, c_R) is in $\Phi(s(t_{new}))$, and such that the lower box in configuration c_{new} is collision-free.
  end if
  if A new feasible step has been found **then**
    if $t_{max} - t_{new} < \epsilon$ **then**
      BREAK.
    end if
    $t_{min} \leftarrow t_{new}$
  else
    $t_{max} \leftarrow t_{new}$
  end if
end while
RETURN $(t_{new},\ (c_{new}, c_R))$

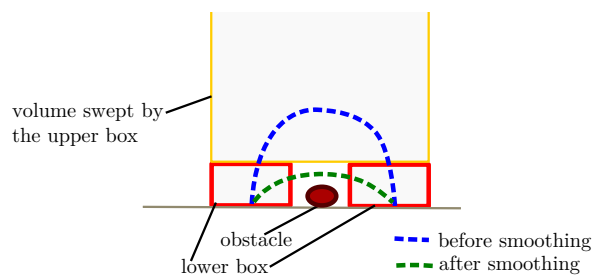

Fig. 4. Swing foot trajectory before and after smoothing. We can see that after the smoothing the robot trajectory is not necessarily contained in the boxes anymore.

C. Smoothing

As shown on Fig. 1 and 2, in order for the robot to remain inside the union of the bounding boxes, the steps constructed have initially $\cap$-shaped swing foot trajectories. Although $\cap$-shaped swing foot trajectories can slightly reduce the stepping capabilities of the robot due to potential joint limits violations, small variations in the height of the center of mass during the steps can compensate for this effect. But because of the distance traveled by the swing foot, such steps are rather slow, and not energy-efficient. For this reason, we use a smoothing process to continuously modify the swing foot trajectories at execution time, so that the swing foot will gently avoid the obstacle, stepping over it without making an unnecessarily large motion. This smoothing process is depicted on Fig. 4 and its specificities are introduced in [16].

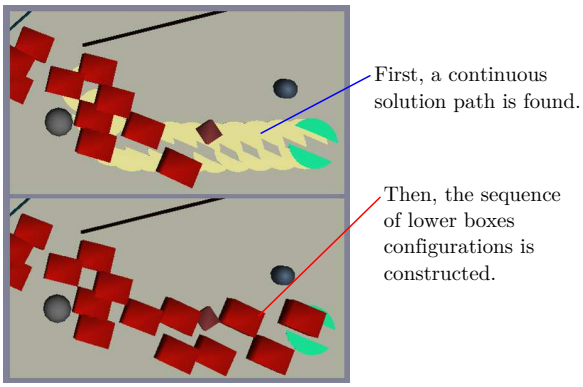

Fig. 5. An example of the phases 1) and 2) of the algorithm. The initial walking trajectory (before smoothing) is such that below height h_m no point outside the lower boxes is touched by the robot (see Fig. 2).

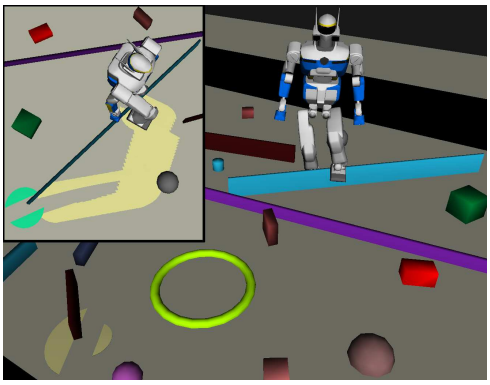

Fig. 6. A simulation of HRP-2 performing real-time footstep planning in an environment cluttered with 3D obstacles. On the bottom-left corner of the image a beige 2D shape made of two portions of disks can be seen. It is a representation of a weakly collision-free configuration $s(t)$ from the continuous path that the robot follows (one side is Φ_L while the other is Φ_R). Note that the configuration is weakly collision-free even though the shape intersects an obstacle. The area swept by this 2D shape along another continuous path is shown on the smaller image in the upper-left corner.

The smoothing can speed up the motion by up to a factor 3, and similarly to the steps construction (section II-B), it is done progressively (i.e. we smooth one step at a time).

III. Implementation and application

Let us first recall the 3 phases of our algorithm: 1) find a continuous path $s(t)$, 2) convert it into a sequence of steps, and 3) smooth the steps.

Fig. 5 illustrates phases 1) and 2).

Phases 2) and 3) require a non-negligible amount of time, especially if the sequence of steps is long. However, they have the advantage of being built progressively. Furthermore, in practice we verify that the time needed for the construction and smoothing of one step (when the continuous path is known) is much less than the duration of the step. Therefore we use two threads (that are launched in parallel just after the continuous path has been found): one that actually executes the walking motion on the robot, and the other that progressively constructs and smooths the steps. The first thread makes the robot start to move as soon as the first step has been constructed and smoothed by the second thread (it takes less than half a second). Then, while the robot is performing the first step, the second thread constructs (based on the continuous path) and smooths the second step which can then be executed without interruption. This parallel process continues further, and since the steps are always constructed and smoothed faster than they are executed, no problem occurs. As a consequence of this use of threads, the relevant time-complexity of the algorithm is only the one of phase 1), unless the robot has to perform other online tasks with the same CPU(s).

As explained in the previous section, finding a continuous path $s(t)$ is a classical problem of rigid body motion planning, except for the configuration space and the configuration validity test (collision checks) which are specific. The library OMPL [1] provides exactly the API we need: we can easily redefine the manifold representing the configuration space as well as the configuration validity test (called "state validity test" in OMPL). Then, any rigid body motion planner can be directly used. We tried the algorithms KPIECE [19], SBL [18], PRM [7], and RRT-Connect [11], all readily available in OMPL, and compared their performances.

We chose the following experimental setup: in a fixed environment (the one of Fig. 6, which is a 4m×4m room cluttered with 3D obstacles lying on the floor), the robot is given 8 consecutive random goals (the same random goals for every motion planner). For each goal and each motion planner we execute our algorithm 10 times and record the average time required to find a continuous solution path, as well as the average number of steps of the sequences constructed. The computations are made on two Intel(R) Core(TM) i7 1.60GHz CPUs. The results are shown in Fig. 7 and 8. What can be seen is that RRT-Connect and KPIECE are the fastest algorithms (averaging both less than half a second on the eight goals), but also that KPIECE produces sensibly longer solution paths that lead to larger sequences of steps.

As collision checking is the bottleneck of most motion planning algorithms, the time costs not only depend on the algorithm chosen but also on the quickness of our configuration validity tests, i.e. our weak collision checks.

In these tests, we must verify whether a collision-free configuration of the lower box exists in Φ_L and in Φ_R. We do this by randomly sampling configurations in Φ_L and Φ_R (which is a conservative approach). If the current Φ_L (resp. Φ_R) has a large overlap with the previously checked Φ'_L (resp. Φ'_R), which happens often, for instance when testing consecutive configurations $s(t_k)$ along a discretized segment line, then it might be a waste of time to check new samples: indeed we might already know a collision-free sample which belongs to both Φ'_L and Φ_L (resp. Φ'_R and Φ_R). For this reason we keep track of the last few collision-free samples found and first test their membership to Φ_L (resp. Φ_R). This method speeds up our validity tests a bit. Note also that because many collision checks are grouped in each validity test, our algorithm is particularly well suited for parallel computations.

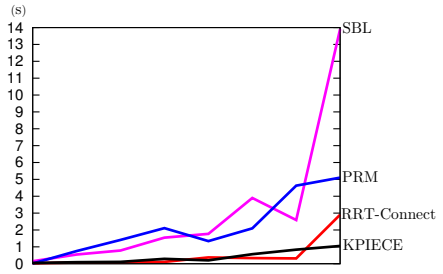

Fig. 7. Planning times for 8 different random goals.

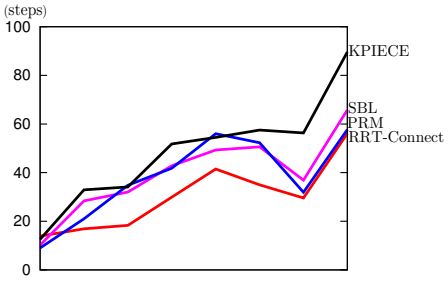

Fig. 8. Number of steps for the same goals as in Fig. 7

IV. Conclusion and Future Work

In this paper we have introduced a new bounding box method for footstep planning for humanoid robots. The two-level bounding box, whose upper part moves continuously and lower part discretely, captures the hybrid behaviour of humanoid robots (and could easily be extended to robots with more legs). Based on this bounding box, our planning algorithm enables the robot to plan dynamic trajectories and accurately step over obstacles at a low computational cost. In fact, we have shown through simulation that real-time footstep planning can be achieved in an environment of limited global size but cluttered with 3D obstacles (here "real-time" means that after receiving a new goal the robot almost always starts to move without any noticeable delay, thus potentially enabling reactive behaviors).

Because the bounding box motion is based on continuous paths, classical motion planning algorithms can be used effortlessly. And since the problem of footstep planning is very specific and much less studied than rigid body motion planning, this generalization is clearly beneficial. We could also further reduce the computation times by using more advanced motion planning libraries such as g-Planner [15] which takes advantage of GPU architectures to achieve extremely fast motion planning. Using such a tool for footstep planning has never been done before, and it should enable the construction of optimized paths in constrained environments in the blink of an eye, and thus pave the way for practical and robust real-time footstep planners.

Acknowledgment

Part of this work was supported by a grant from the RBLINK Project, Contract ANR-08-JCJC-0075-01. Young J. Kim was supported in part by IT R&D program of MKE/MCST/KOCCA (2008-F-033-02) and NRF and Korea (NRF-2011-013-D00112). Dinesh Manocha was supported in part by ARO Contract W911NF-10-1-0506, NSF awards 1000579 and 1117127.

References

[1] The Open Motion Planning Library (OMPL). `http://ompl.kavrakilab.org`, 2010.

[2] Y. Ayaz, K. Munawar, M. Bilal Malik, A. Konno, and M. Uchiyama. Human-like approach to footstep planning among obstacles for humanoid robots. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'06)*, 2006.

[3] J.-M. Bourgeot, N. Cislo, and B. Espiau. Path-planning and tracking in a 3d complex environment for an anthropomorphic biped robot. In *IEEE Intl. Conf. on Intelligent Robots and Systems*, volume 3, pages 2509–2514, 2002.

[4] J. Chestnutt, J.J. Kuffner, K. Nishiwaki, and S. Kagami. Planning biped navigation strategies in complex environments. In *IEEE Int. Conf. on Humanoid Robotics (Humanoids'03)*, 2003.

[5] J. Chestnutt, M. Lau, G. Cheung, J.J. Kuffner, J. Hodgins, and T. Kanade. Footstep planning for the honda asimo humanoid. In *IEEE Int. Conf. on Robotics and Automation (ICRA'05)*, pages 631–636.

[6] J. Chestnutt, K. Nishiwaki, J.J. Kuffner, and S. Kagami. An adaptive action model for legged navigation planning. In *IEEE/RAS Int. Conf. on Humanoid Robotics (Humanoids'07)*, pages 196–202, 2007.

[7] R. Geraerts and M.H. Overmars. A comparative study of probabilistic roadmap planners. In *5th Workshop on the Algorithmic Foundations of Robotics (WAFR'02)*, 2002.

[8] J.-S. Gutmann, M. Fukuchi, and M. Fujita. Real-time path planning for humanoid robot navigation. In *Int. Joint Conf. on Artificial Intelligence (IJCAI05)*, pages 1232–1237, 2005.

[9] J.-B. Hayet, C. Esteves, G. Arechavaleta, and E. Yoshida. Motion planning for a vigilant humanoid robot. In *9th IEEE-RAS International Conference on Humanoid Robots (Humanoids'09)*, pages 196–201, 2009.

[10] A. El Khoury, M. Taix, and F. Lamiraux. Path Optimization for Humanoid Walk Planning: an Efficient Approach. In *Int. Conf. on Informatics in Control, Automation and Robotics (ICINCO'11)*, 2011.

[11] J.J. Kuffner and S.M. Lavalle. RRT-Connect: An efficient approach to single-query path planning. In *IEEE Int. Conf. on Robotics and Automation (ICRA'00)*, pages 995–1001, 2000.

[12] J.J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue. Footstep planning among obstacles for biped robots. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'01)*, pages 500–505, 2001.

[13] E. Larsen, S. Gottschalk, M.C. Lin, and D. Manocha. Fast proximity queries with swept sphere volumes. Technical report, Univ. N. Carolina Chapel Hill, 1999.

[14] J.M. O'Kane and S.M. LaValle. Sampling-based methods for discrete planning. In *Doctoral Consortium of the International Conference on Automated Planning and Scheduling*, 2004.

[15] J. Pan, C. Lauterbach, and D. Manocha. g-Planner: Real-time motion planning and global navigation using GPUs. In *AAAI Conference on Artificial Intelligence (AAAI'10)*, 2010.

[16] N. Perrin, O. Stasse, F. Lamiraux, and E. Yoshida. A biped walking pattern generator based on "half-steps" for dimensionality reduction. In *IEEE Int. Conf. on Robotics and Automation (ICRA'11)*, 2011.

[17] N. Perrin, O. Stasse, F. Lamiraux, and E. Yoshida. Weakly collision-free paths for continuous humanoid footstep planning. In *IEEE Int. Conf. on Intelligent Robots and Systems (IROS'11)*, 2011.

[18] G. Sanchez and J.-C. Latombe. A single-query bi-directional probabilistic roadmap planner with lazy collision checking. In *Int. Symp. Robotics Research*, pages 403–417, 2001.

[19] I.A. Sucan and L.E. Kavraki. Kinodynamic motion planning by interior-exterior cell exploration. In *8th Workshop on the Algorithmic Foundations of Robotics (WAFR'08)*, 2008.

[20] Z. Xia, G. Chen, J. Xiong, Q. Zhao, and K. Chen. A random sampling-based approach to goal-directed footstep planning for humanoid robots. In *IEEE/ASME Int. Conf. on Advanced Intelligent Mechatronics (AIM'09)*, pages 168–173, 2009.

[21] E. Yoshida, C. Esteves, I. Belousov, J.-P. Laumond, T. Sakaguchi, and K. Yokoi. Planning 3D collision-free dynamic robotic motion through iterative reshaping. *IEEE Trans. on Robotics*, 24(5):1186–1198, 2008.