Biped Footstep Planning

Nicolas Perrin

▶ **To cite this version:**

Nicolas Perrin. Biped Footstep Planning. Ambarish Goswami; Prahlad Vadakkepat. Humanoid Robotics: A Reference, Springer Netherlands, pp.1-21, 2017, 978-9400760455. 10.1007/978-94-007-7194-9_29-1 . hal-03089498

HAL Id: hal-03089498
https://hal.science/hal-03089498v1

Submitted on 15 Feb 2023

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.

50 Biped Footstep Planning

Nicolas Perrin

Institut des Systèmes Intelligents et de Robotique (ISIR), Université Pierre et Marie Curie-Paris 6, CNRS UMR 7222, 4, place Jussieu, 75005 Paris. perrin@isir.upmc.fr

Summary. Planning footsteps before the actual walking motion is a way to ignore the dynamic complexity of bipedal walking in the motion planning process without losing the possibility to exploit the essential characteristic of walking: the fact that it relies on isolated contacts with the ground, and therefore provides the ability to traverse irregular or discontinuous terrain. The duality between a continuous environment and discrete sequences of contacts confers to footstep planning a hybrid nature and an interesting theoretical aspect, which adds up to its obvious practical interest. Well incorporated and efficient footstep planning abilities would give humanoid robots a navigation autonomy allowing them to perform a great number of versatile tasks in unstructured environments. Many approaches have been considered to obtain efficient footstep planning, for example using only a finite number of predefined steps, or bounding boxes for fast collision checking. And for really reactive footstep planning, dynamics have to be put back into the equation, either by merging footstep planning with gait control, or by adaptively switching between different layers of planning. In this chapter, we give an overview of the main techniques that have been proposed and tested over the past 15 years.

Key words: footstep planning, motion planning, bipedal locomotion

50.1 Introduction

Bipedal walking is a difficult problem in itself, but as a mode of transport, it should be usable as a low-level tool whose complexity is abstracted away from higher-level tasks such as path planning. Although the particular dynamics of walking have to be addressed carefully, reasoning on footstep plans is a natural choice for this abstraction. Indeed, it hides most of the complexity of walking without losing its essential characteristic: the fact that it relies on isolated contacts with the ground, and therefore provides the ability to traverse irregular or discontinuous terrain. The duality between a continuous environment and discrete sequences of contacts confers to footstep planning a hybrid nature and an interesting theoretical aspect, which adds up to its obvious practical interest. Well incorporated and efficient footstep planning abilities would give humanoid robots a navigation autonomy allowing them to perform a great number of versatile tasks in unstructured environments.

Over the past two decades, the progressive development of reliable humanoids made extensive research on this topic possible, with the goal of reaching abilities comparable to those of humans. But this is not an easy challenge, as humans plan and execute walking motions with absolutely remarkable ease. Through a long process of learning, humans develop, in parallel to their motor skills, various heuristics that help them evaluate the terrain and obstacles (whether the floor is going to be slippery or not, whether an object can be stepped on

or not, etc.), plan efficiently adequate paths and guess quickly and accurately whether they will be hard to follow or not, without having to think precisely about the motions that will be required, except in complex scenarios. Precise footsteps are planned only when needed, and they can be modified very reactively when unpredicted events occur, while dealing well with balance issues. Humanoid robots are still far from this level of virtuosity, but a lot of promising approaches and algorithms have been introduced and tested.

The biggest challenge is that footstep planning is hard to dissociate from the realization of the walking motion, which involves underactuation and complex dynamics in high dimensional spaces. A lot of research has aimed at introducing good simplifying hypotheses to make the problem computationally affordable while still being able to exploit well the advantages of legged locomotion, and produce efficient and fast walking. But a good trade-off is very hard to obtain. For example, it is easier to plan and control statically stable walking motions (motions that involve only balanced postures, see [28]), but they are usually slow, not energy-efficient, and better obstacle clearance can be obtained with more dynamic motions. Another common method for footstep planning is called the bounding box method (see [49]): it first plans the continuous motion of a large box that contains the whole robot, and then a sequence of steps that follows the box trajectory. If the bounding box trajectory is collision-free, then the robot trajectory is a fortiori collision-free, but the drawback of this method is obvious: the bounding box must circumvent all the obstacles on the floor, even the smallest ones that could easily be stepped over. Thus, it becomes quite unnecessary for the robot to have legs, making this method better suited for wheeled robots such as the PR2.

Probably the most successful approaches for footstep planning are based on the use of the A* search algorithm (or variants) with a finite transition model, i.e. a finite set of possible steps decided in advance (see for example [28], [5], [8], [9], [16]). However, using finite action sets is not ideal, one reason being that the complexity of the A* search quickly increases with the size of the transition model, therefore this size is limited and so are the stepping capabilities. It can lead to walking motions with little flexibility, or unnecessarily large number of steps for simple actions. A lot of ad hoc methods have been proposed to extend this approach and perform better footstep planning, such as for instance local footstep adjustments [10], human-like strategies [3], or tiered planning combining different low-level and high-level planners [7]. There are also continuous methods that rely on the capacity of the robot to make infinitesimally small steps: sequences of steps are produced based on trajectories of the robot "sliding" on the ground [25, 12]. However, with these methods stepping over obstacles is not possible. Other methods are based on solving optimization problems [18, 13]. In fact, a full spectrum of strategies has emerged, ranging from "discrete approaches" on one end to "continuous approaches" on the other, with very different associated planning techniques. In this chapter, without being exhaustive, we propose a review of some of the approaches that have been proposed for biped footstep planning.

Fig. 50.1 gives an overview of the motivation for footstep planning and its main challenges.

50.2 Footstep planning problem formulation in 2D

The simplest case of footstep planning consists in planning sequences of footsteps on a flat horizontal ground with 2D obstacles. Potential collisions between obstacles above the ground and the legs or the upper body of the robot are ignored.

A state of the robot corresponds to a double-support stance, and is described by the configurations of both feet (in position and orientation). A foot configuration is a 4-tuple

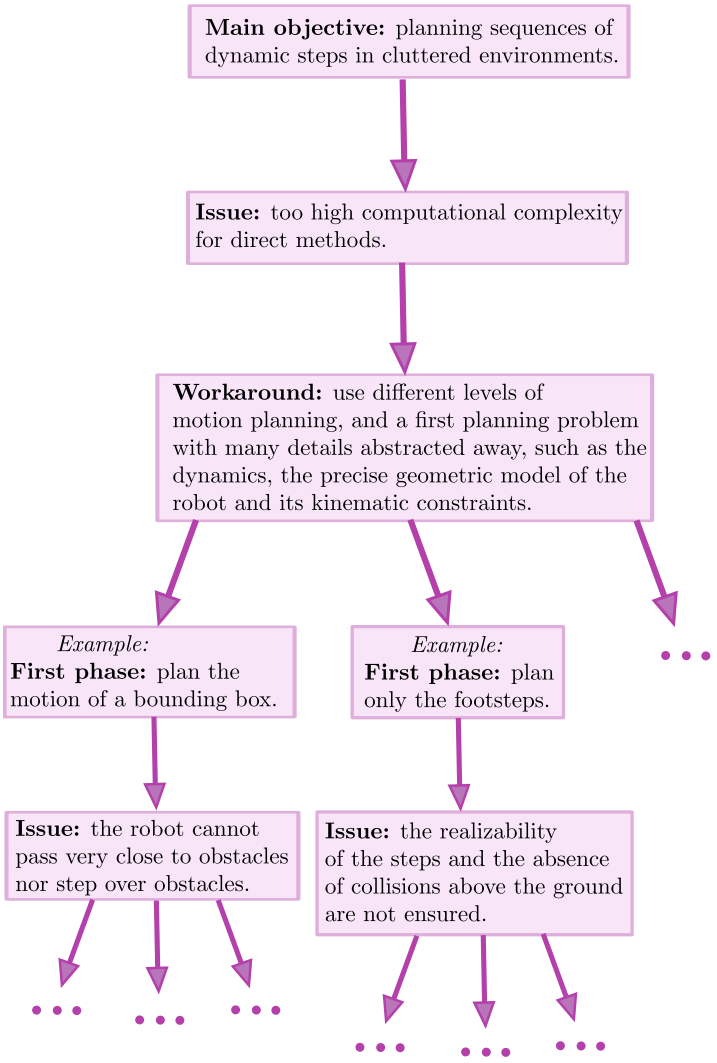

Fig. 50.1. Footstep planning is an example of a natural and widespread technique that consists in solving difficult problems via several phases of computation, using at first abstractions that hide part of the problem complexity but lead to a rough estimate of a potential solution. It faces the same issues as all approaches based on this principle: by not dealing with the full complexity of the problem right away, it computes estimates that can lead to either largely suboptimal solutions or no solution at all. In this chapter we present various methods and heuristics that have been designed to obtain a good compromise between the quality of the footstep plan and the algorithmic efficiency.

$(P,x,y,\theta) \in \{L,R\} \times SE(2) = \mathscr{C}_{FS}$ (here the subscript FS stands for "footstep"), where SE(2) denotes the 3-dimensional special Euclidean group of rigid-body motions in the plane. The first element of the tuple, P, can be either L or R; it differentiates left footsteps from right footsteps. The other elements, (x,y,θ), uniquely define the position and orientation of the footstep.

The obstacles define a collision-free space $\mathscr{F}_{FS} \subset \mathscr{C}_{FS}$. The objective of footstep planning is to find a sequence of footsteps in $\mathscr{F}_{FS}$ from an initial state to a goal region, with some constraints on the transitions between states. Indeed, the kinematics of the robot and its actuation restrict the set of steps that can be performed. There are many options to define the transition restrictions. Ideally, the set of possible steps in a given state should depend on the steps previously performed, and on how these steps were performed. If, for example, the robot is walking forward at a fast pace, it is likely that it will be impossible for it to suddenly step backwards. However, this dependency on previous steps results in an explosion of the dimensionality, and in general footstep planning is simplified by considering that the set of possible next footsteps depends only on the configuration of the support foot. From a state $(l,r) \in \mathscr{C}_{FS}^2$, the set of new states (l',r) reachable with a left step is defined by the following constraint:

$$l' \in S(r). \tag{50.1}$$

Similarly, the set of states (l,r') reachable with a right step is defined by the constraint:

$$r' \in S(l). \tag{50.2}$$

$S : \mathscr{C}_{FS} \to \mathscr{P}(\mathscr{C}_{FS})$ is such that for any right footstep r, $S(r)$ contains only left footsteps, and reciprocally, for any left footstep l, $S(l)$ contains only right footsteps. This makes it compulsory for the robot to alternate right and left steps, which is usually not an issue since allowing consecutive left (or right) steps does not change the robot navigation abilities. The possibility of in-place stepping is ensured by the following symmetry property, often verified in practice:

$$\forall l = (L,x,y,\theta), \forall r = (R,x',y',\theta'),\ l \in S(r) \Leftrightarrow r \in S(l) \tag{50.3}$$

Let us call S the transition function.

There are different ways to define the goal region, and we choose to define it as a subset of $\mathscr{F}_{FS}$, reached as soon as one of the footsteps belongs to it. We can now formally specify the input and output of footstep planning in 2D:

Data:
- $S : \mathscr{C}_{FS} \to \mathscr{P}(\mathscr{C}_{FS})$, the transition function
- $\mathscr{F}_{FS}$, the free space (usually known implicitly via the description of the obstacles)
- $(s_0,s_1) \in \mathscr{F}_{FS}^2$: the initial state, verifying $s_1 \in S(s_0)$
- $G \subset \mathscr{F}_{FS}$: the goal region

Result: A sequence of footsteps $s_2, s_3, \ldots, s_n \in \mathscr{F}_{FS}$ such that $\forall i \in \{1,\ldots,n-1\}$, $s_{i+1} \in S(s_i)$, and $s_n \in G$.

Fig. 50.2 illustrates footstep planning in 2D, and Fig. 50.3 shows a classical example of transition function model for a humanoid robot. An ideal transition function should have a relatively simple structure, and yet model accurately the stepping capabilities of the robot. For a given robot with a given walking algorithm, a good choice for the function S can be progressively obtained after thorough tests and experiments. Conservativeness of the transition function is always preferred over allowing infeasible steps.

The main particularity of footstep planning is that it is not an entirely discrete planning problem, because of the continuity of the configuration space, and at the same time it is not

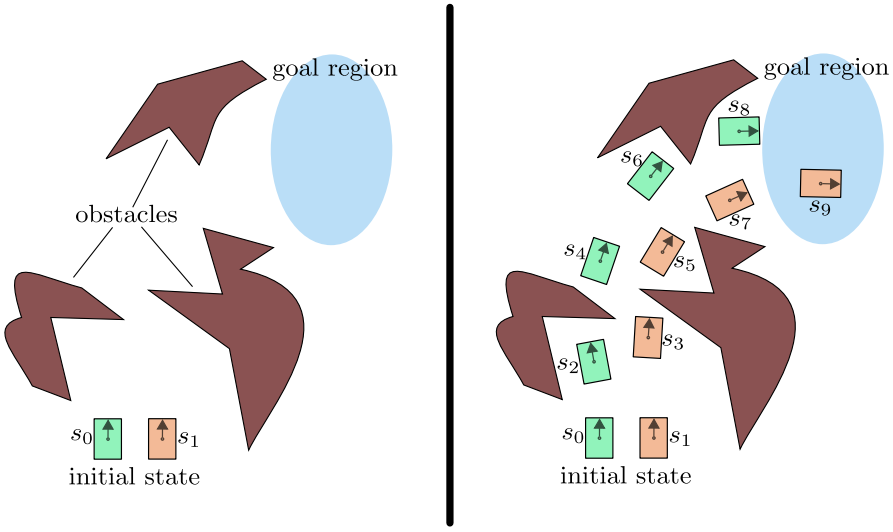

Fig. 50.2. On the left: an instance of the footstep planning problem in 2D; on the right: a solution.

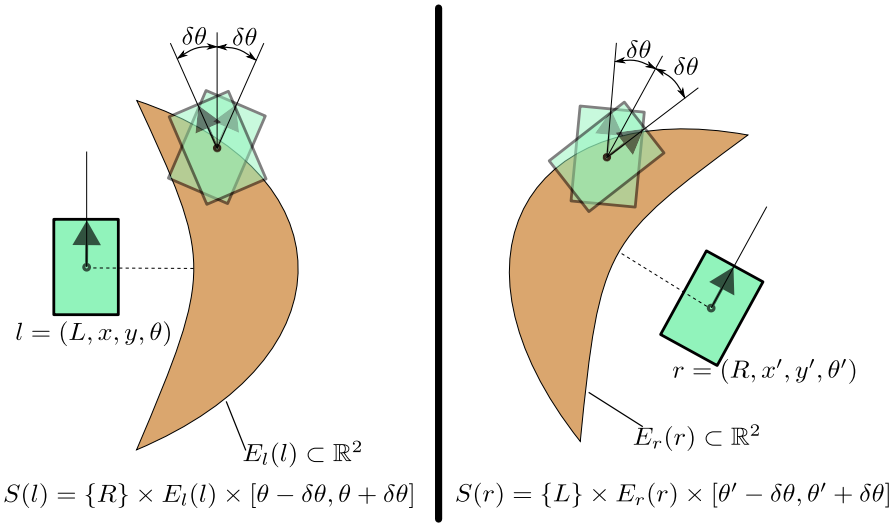

Fig. 50.3. An example of classical "shape" for the transition function, usually empirically defined. The maximum distance between the feet is limited by the length of the legs and kinematic constraints. The transition function should also prevent self-collisions from occurring, so E_l and E_r are defined in a way that keeps the feet at a safe distance from each other. A constraint also bounds the difference of orientation of the feet, as very different orientations can typically result in collisions between the knees or violations of kinematic constraints at the hips.

a continuous motion planning problem either. It is a motion planning problem of hybrid nature, and, as a result, neither classical algorithms for continuous motion planning (the piano mover's problem) nor discrete planning techniques can be directly applied. That is why developing the footstep planning abilities of humanoid robots has required the creation of specific approaches. And although footstep planning is a very applied problem, its particular hybrid nature gives it an additional theoretical interest, and can be studied in more abstract problems, for example by considering point feet, ignoring orientations, and choosing a simple transition function. See [4] and [42] for theoretical studies of such problems. Fig. 50.4 illustrates the "flea motion planning problem", introduced in [44], which can be seen as a very simplified footstep planning problem.

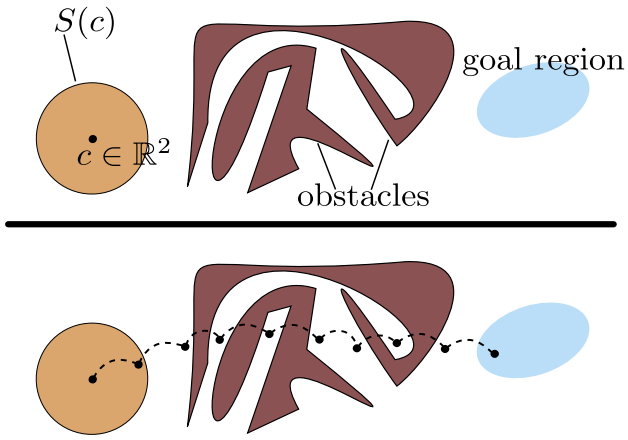

Fig. 50.4. Flea motion planning: a variant of 2D footstep planning with state-space $\mathbb{R}^2$.

50.3 2D footstep planning problem solution via finite transition sets

The most widespread method to obtain easily implementable and efficient algorithms for footstep planning is to use finite sets for the transitions. With $S(s)$ finite for any state s (using the notation of the previous section), the problem can be solved with search algorithms on discrete graphs. As in [9], a classical choice is the A* algorithm, described by the pseudo-code of Algorithm 1 (where Q is a priority queue).

In this algorithm, Q.ExtractMin() extracts and removes from the queue an element with minimum $reach\_cost + expected\_cost$ value. StepCost() attributes a cost to steps. For example, a large step can be judged more costly than a standard one, and therefore be associated to a greater cost. Alternatively, if one is interested only in minimizing the number of steps, StepCost() can be defined to return a constant value. LocationCost() attributes to footstep locations a cost that depends on terrain information. For instance, if the terrain representation is stored as a heightmap, some regions are considered as obstacles, and for acceptable footsteps

Data: S, $\mathscr{F}_{FS}$, $(s_0, s_1) \in \mathscr{F}_{FS}^2$, $G \subset \mathscr{F}_{FS}$
Q.Insert(s_1, 0, 0, s_0)
while *running_time* $< t_{max}$ **do**
  $(s_{best}, reach\_cost, expected\_cost, s_{parent}) \leftarrow$ Q.ExtractMin()
  if $s_{best} \in G$ **then**
    return s_{best}
  end
  foreach $s_{next} \in S(s_{best})$ **do**
    $c_s \leftarrow$ StepCost(s_{parent}, s_{best}, s_{next})
    $c_l \leftarrow$ LocationCost(s_{next})
    $c_e \leftarrow$ ExpectedCost(s_{next}, G)
    Q.Insert(s_{next}, $reach\_cost + c_s + c_l$, c_e, s_{best})
  end
end

Algorithm 1: A* for footstep planning. Remark: when the algorithm is successful, a sequence of steps from (s_0, s_1) to the goal region G is implicitly created; for the sake of clarity, the pseudo-code contains no data structure that would enable a reconstruction of this path after termination of the algorithm, but it is necessary in any real implementation. See the second paragraph of section 50.3 for a description of Q.ExtractMin(), StepCost(), LocationCost() and ExpectedCost().

outside these regions, the location cost is high if the terrain is perceived as rough, inclined, or if it is partially unknown. The lowest location costs are usually in flat and horizontal regions with a lot of data. In [8], Chestnutt et al. discuss various criteria for location costs. Finally, ExpectedCost() is the heuristic used by the A* algorithm, which estimates the remaining total cost to go to the goal. Assuming the existence of a solution, A* is guaranteed to find an optimal one if this heuristic never overestimates the remaining cost. However, A* with an inflated heuristic or overestimations is sub-optimal but proves to find good solutions faster in many cases. In footstep planning, there are several options for the heuristic. The most naive one is to use the Euclidean distance between the current footstep and the goal region. With appropriate choices for the location and step costs, the Euclidean distance is an "admissible" heuristic, i.e. a heuristic that never overestimates the cost-to-go. Another common approach is to plan backwards from the goal region with a standard mobile robot planner as a precomputation step. The planner explores outwards from the goal region, and records the cost to reach each location. As explained in [9], this cost provides an informed estimate of the actual cost, and steers the robot away from local minima (unlike the Euclidean heuristic). Since mobile robot planners are typically fast (compared to the A* search for sequences of footsteps), this precomputation step is not overly computationally expensive. However, by ignoring the ability of the robot to step over obstacles, the resulting heuristic can overestimate the cost-to-go, which means that this heuristic sacrifices optimality guarantees for execution speed.

One of the main issues with the A* approach is the size of the transition sets $S(s)$, or action sets. Usually, it is constant or has a few possible different values. For instance, in [9], two different transition sets are used, depending on whether the robot is walking at full speed or standing still. Fig. 50.5 shows an example of transition set. In [8], experiments show that decreasing the number of transitions, i.e. the branching factor, decreases in general the computation time, but also that on the other hand, a smaller number of options reduces the set of solvable maps. It is rather difficult to find a good compromise, and the size of the transition

set should ideally depend on the complexity of the environment. Based on the literature, good results can be obtained with sets of about 15 transitions in average.

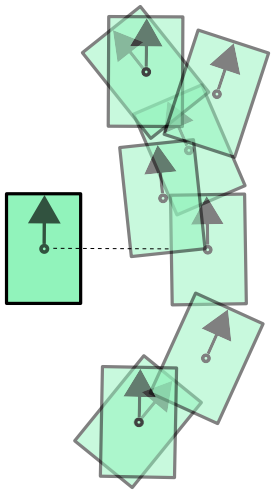

Fig. 50.5. A finite transition set for right steps. The current step is in dark green (the left foot), and the 9 potential next right foot placements are in light green. Typically, such a set of steps is defined manually with the aim to give good stepping capabilities to the robot, allowing it to perform forward, backward and sideway steps, and to turn left and right.

Variants and extensions of the A* approach have been studied. In [21], ARA* [31] and R* [32] are used instead of A*. ARA* stands for Anytime Repairing A*. Its main property is that it quickly finds a feasible solution and then continually works on improving it until time runs out. Therefore, it is better suited for the time constraints often imposed in footstep planning. R* is another heuristic search algorithm that depends much less than A* on the quality of the heuristic function, which is often difficult to chose appropriately. Both ARA* and R* run series of A* searches with inflated heuristics (in the case of R* towards randomly chosen subgoals), and both have bounds (probabilistic bounds for R*) on the sub-optimality of the solutions they find. For footstep planning in changing environments, Garimort et al. [15] used the incremental D* Lite algorithm [26] which extends A* by continually searching for shortest paths in the environment while the current starting state progresses along the path and the edge costs may change arbitrarily.

Ayaz et al. [3] proposed to classify the transitions into several categories based upon the function they realize: stepping over obstacles, walking straight, turning, etc. A planning algorithm can then perform more human-like footstep planning by first deciding which category of motion is required, and then looking for the right step to perform.

To go beyond finite action sets and get better flexibility, Chestnutt et al. proposed an extension of the A* search algorithm with possible local adjustments of the footsteps [10].In [21], transitions are dynamically added in special situations, for instance to reach the goal region.

Hauser introduced a general two-stage approach for legged locomotion [17]. Random sampling-based methods generate key intermediate stances, or milestones, and then a finite set of motion primitives is used to help the planner generate motions between the milestones (the motion primitives are transformed to match the terrain and the intermediate stances).

Other attempts to combine sampling-based algorithms (such as RRT [29]) and finite action sets have been proposed [48, 43, 33]. In [45] and [42], a geometric property of the flea motion planning problem is generalized and used to transform footstep planning into a continuous problem, thus avoiding finite transition sets and enabling a direct application of traditional sampling-based algorithms.

50.4 Bounding boxes

In practice, it is not enough to reason only about the footsteps since the whole body of the robot must be taken into account. This implies checking for collisions between all the body parts and the environment. But the geometrical complexity of humanoid robots, the dynamicity of walking motions, and the need to explore many paths in cluttered environments make direct approaches almost surely intractable. A common solution is to consider bounding boxes to simplify collision tests. The most basic approach consists in using a unique large box that contains the whole robot. A preliminary phase finds a collision free path for the box towards a goal region, sliding it on the ground with translations and rotations. It is particularly convenient that this can be done using classical sampling-based planning algorithms. To obtain natural-looking motions, it might be adequate to perform nonholonomic motion planning, since evidence suggests that this is what humans tend to do [1]. After the motion of the box has been set, a walking motion following the same path is computed. If along this motion the robot remains inside the volume swept by the bounding box, no additional collision test is required.

The main problem of this method is that it is over-conservative in several ways. For instance, in narrow passages, the box might not pass while the robot could, and, more importantly, the use of such a bounding box removes the ability to step over obstacles, which is one of the main advantages of legged locomotion.

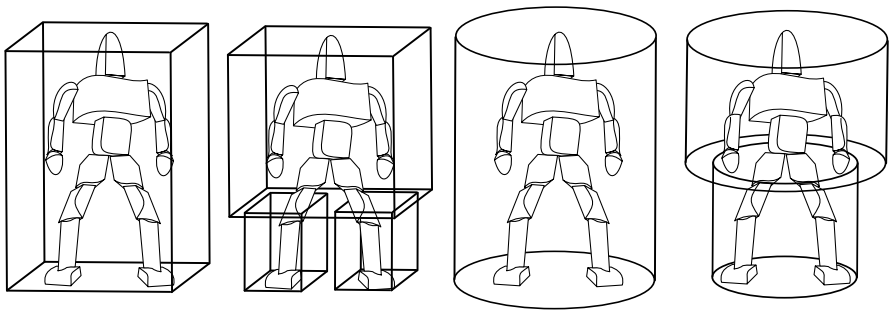

Fig. 50.6. A few examples of bounding boxes.

Planning first a bounding box motion makes completeness impossible in the sense that some solutions will necessarily be missed. Nevertheless, there exist various methods to limit this drawback. In some cases, partial boxes suffice [49]. Different shapes can also be considered to better fit the robot's geometry. Furthermore, the bounding volume can actually be a union of several boxes (e.g. [16]; see also Fig. 50.6), either rigidly linked together in one particular configuration, or with degrees of freedom allowing relative movements of the different boxes. The bounding box can also be adaptive, for example with a structure of bounding volume hierarchy, but using a tight approximation is difficult, for as the bounding volume becomes more complex, the motion planning gets increasingly closer to directly planning the whole-body motion. Another challenge with relatively tight bounding boxes is to make sure that the robot remains inside the box during its motion. For that, bounding volumes can directly be computed from robot motions, as in [6], rather than from fixed configurations. In particular, with a finite set of transitions, a bounding volume can be computed for each action, provided that concatenating these actions does not require a modification of the motion. This issue is addressed in [43], where an approximated bounding volume is computed for a set of half-steps. In [35], for each action of the finite set of transitions, an inverse heightmap (IHM) is computed. These inverse heightmaps are compact representations of bounding volumes that are especially useful to check quickly and precisely which obstacles the robot can circumvent or step over. Other works have addressed the problem of stepping over obstacles by using special bounding boxes for the legs ([44, 13]). Here again, one of the difficulties is to keep the motion planning for the box easier than a direct motion planning of the steps of the robot.

In summary, with approaches based on bounding boxes, the main issue is to find a good compromise that avoids over-conservativeness and still provides a significant speed-up. Because of the complex motions and the nonlinear dynamics involved in walking, combining seamlessly bounding box-based and more precise motion planning in a hierarchical way remains an open challenge.

Bounding boxes also permit to accelerate precomputations, as shown in [13] where a bounding box is used to find large convex regions of the configuration space that are free from obstacles.

50.5 Tiered strategies

Bounding box approaches are usually tiered strategies in the sense that they divide the planning into two phases: a high-level planning phase that computes a path for the bounding box, and a low-level planning phase that defines the footsteps and the actual motion of the robot. There exist various types of tiered strategies, which are a very natural way to tackle the problem, by using simple models and heuristics to find a global path, and extract information from this path to guide precise, low-level motion planning algorithms.

In [7], the environment (e.g. a building floor) is represented as a finite graph (where the vertices are rooms or specific areas, and edges correspond to doors, corridors or adjacency between rooms or areas) and the high-level planner performs a graph search to find a path from the initial location to the goal. A mid-level planner based on mobile robot path planning provides a heuristic that enables the low-level planner to find sequences of footsteps to intermediate goals (subgoals) along the global path. Fig. 50.7 illustrates a similar tiered strategy. Strict subgoals should in general be avoided since they might constrain the robot's path through undesirable locations. In [7], the authors propose two options: either switching to

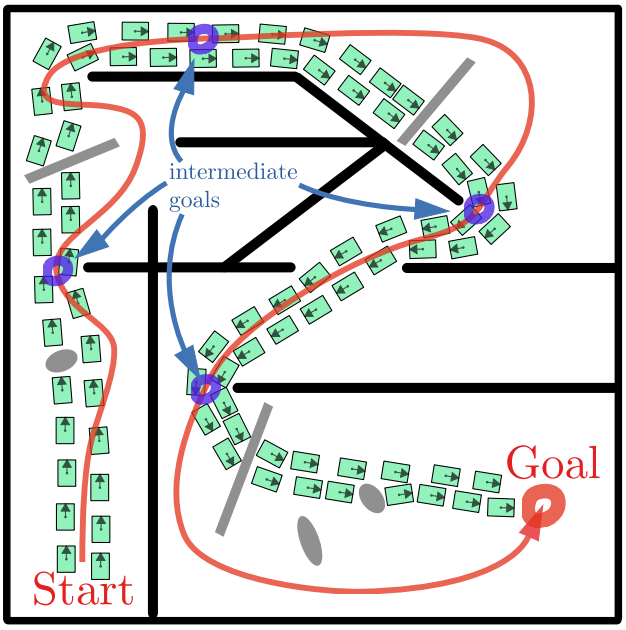

Fig. 50.7. An example of tiered strategy: first, a high-level path is planned, for instance with a bounding box or a mobile robot motion planning algorithm. All the obstacles are avoided, even the gray ones which can potentially be stepped over. In a second phase, the sequence of footsteps is planned, using intermediate goals taken from the high-level path. At this stage "shortcuts" can be found, using the ability of the robot to step over small obstacles. Additional levels of planning can be considered: for example, a higher level using a finite topological graph describing a building's floor plan, or a lower level motion planner that computes collision-free whole-body motions given short sequences of footsteps and obstacles in proximity.

the next subgoal before actually reaching the current one, or constantly updating the subgoal to some distance ahead of the robot's current position, thus introducing a receding planning horizon.

The tiered strategy proposed in [16] also relies on a three-level architecture: a perception layer creates a 2.5D terrain model, following the method introduced in [24], and classifies the areas into one of six different types. A control layer takes in input the environment classification and selects a sequence of actions based on behavior modules, such as normal walking, crawling, walking sideways, climbing stairs, etc. Finally, a planning layer takes the sequence of actions in input and searches for a collision-free path in the environment map.

The hierarchical planner proposed in [6] is also based on an initial workspace decomposition: from a 3D polygonal mesh of the environment, the height of the floor, height of the ceiling and orientation of the floor are extracted into a set of 2D maps, which constitute the 2.5D representation. Then a global planner identifies a set of regions through which the robot can move, and represents the ability of the robot to move between these regions in

a graph structure. This connectivity graph is used to find subgoal regions towards a target location, and a local planner performs modified A* searches to compute sequences of motion primitives (such as forward step, side step, turning in place, step up or down, etc.) that make the robot go from a subgoal region to the next while avoiding the obstacles. The local planner may be unable to find a collision-free trajectory. In that case, it reports the failure to the global planner that processes the information and tries to propose another path to the goal. The possibility of failure of the local planner shows that hierarchical approaches are not always "sound" at every level of planning. This notion of soundness, dual to the issue of completeness already raised by bounding box approaches, is crucial in tiered strategies: indeed, it is difficult to guarantee that the paths found by a high-level planner always lead to feasible queries for the low-level planner. For example, the method described in [30] suggests to use a rough evaluation of the obstacles to plan a trajectory for the body trunk, and then a local planner to place the footsteps and compute the motion of the legs. Planning first the trajectory of the upper body is computationally efficient, but for a given trajectory there is no guarantee that valid steps can be found. Giving more degrees of freedom to the low-level planners can decrease the likelihood of failure, but it remains an open problem to get very precise notions of completeness and soundness in tiered strategies for humanoid robot footstep planning, although handling these issues with flexibility and uncertainty might be a better solution. In [20], an evaluation of the terrain is used to decide switches between direct footstep planning in cluttered regions of the environment, and a fast grid-based 2D planning in open spaces. When grid-based planning is used, soundness is ensured by "inflating" the obstacles, which raises again the question of completeness since this conservative choice might cause the robot to miss some solutions. But this does not happen since the algorithm switches back to direct footstep planning when the obstacles are difficult to avoid, showing that adaptive level-of-detail planning leads to a good compromise between efficiency, soundness and completeness.

50.6 Optimization-based footstep planning

Instead of defining hierarchical layers of motion planning, a completely different way to solve footstep planning is to merge it with low-level tasks by incorporating it to the whole-body controller. As mentioned in [46], the motivation comes from the fact that separating planning from control makes it difficult to account for the whole-body motion objectives and constraints in the plan. In [46], footstep planning and control are merged together by introducing long-term balance constraints in the whole-body controller. Steps are triggered automatically whenever achieving a task (for instance a reaching task) would require a balance constraint to be violated. Potential future steps are taken into account via model predictive control. Using discrete time, finding the steps and control inputs amounts to solving an optimization problem. Similarly, [18] proposed a model predictive control scheme that solves convex quadratic programming problems to plan footsteps and the center of mass trajectory, with the objective to track a reference velocity as well as possible while ensuring the feasibility of the steps and their dynamic balance (under some classical assumptions involving simplified dynamics).

In the original method presented in [25], footstep planning is done via inverse kinematics. Given a task (e.g. grasping an object), an upper bound N on the number of steps needed is estimated. Then, the N potential footsteps are linked with a virtual kinematic chain, and the configuration of the chain is iteratively improved through inverse kinematics. The main advantage of this approach is that the virtual kinematic chain can be considered as a part of the

robot. This implies that a unique framework of iterative inverse kinematics can compute the positioning of the footsteps and the continuous motion of the robot limbs in a similar way. Adding constraints on joint limits, collision avoidance, relative feet placement, position of the center of mass, etc., each configuration update can be solved as a nonlinear optimization problem that tries to minimize a cost related to one or several task objectives, while always satisfying the constraints.

These examples show that approaches merging planning and control have many advantages, but they also tend to have two major drawbacks: their relatively high computational complexity, and the fact that it is usually difficult to obtain a global optimum, or a "good" local optimum, in reasonable time.

Optimization-based approaches have also been used in strategies that separate footstep planning from the whole-body control. For example with evolutionary algorithms dealing with multiple objectives [19], or in [14] where bounding box paths are optimized using knowledge on the robot walking abilities (for example, that it walks faster straight than sideways).

In [13], an optimization of the footsteps is done in two phases. In a precomputation phase, large convex obstacle-free regions of the configuration space are computed using a multi-level bounding box. This leads to a finite set of convex regions where the feet can safely be placed. Footsteps are assigned to regions via a matrix of binary variables. Given the initial location and the goal region, an upper bound on the total number of footsteps is fixed, and new binary variables are added to allow an optimization of the number of steps actually used. The complete formulation takes into account the precise placement of the footsteps (position and orientation), linear constraints on relative configurations between consecutive footsteps (computed based on piecewise linear approximations of sine and cosine), and various costs on the steps. It results in an optimization problem where the typically nonlinear, non-convex constraints of footstep planning, are replaced by mixed-integer convex constraints. The main advantage is that the corresponding mixed-integer quadratically-constrained quadratic program (MIQCQP) can be efficiently solved to its global optimum.

In [23], another algorithm based on mixed-integer programming realizes footstep planning within a model predictive control scheme based on simplified dynamics of the robot, and shows the following important point: merging footstep planning with control does not necessarily mean losing the ability to search for globally optimal trajectories.

50.7 Dynamics and reactiveness

Footsteps are not only used for navigation, they also play an important role in maintaining balance. In slow walking, when all the velocities involved in the robot motion can be approximated to zero (this is called "quasi-static walking"), a configuration is balanced if and only if the vertical projection of the center of mass on the ground is inside the convex hull defined by all the supporting contact points between the robot and the environment. For example, during single support, this corresponds to the center of mass being vertically aligned with the support foot. Quasi-static walking is characterized by successive shifts of the center of mass from one foot to the other. These shifts result in unnecessary sway motions, and excessive energy consumption. Dynamic walking is much more efficient and interesting, but it implies that the robot is almost constantly in a falling motion, the fall only being prevented by the next footstep(s). As a result, the required balance criteria become more complicated, and it becomes more difficult to concatenate steps. For example, during moderately fast walking, it is usually impossible to perform a backward step just after a forward step.

In [27], Kuffner et al. proposed a method for dynamic locomotion planning that uses a sampling-based algorithm to find dynamic trajectories between consecutive stances. Although the walking motions produced by this method are not quasi-static, they are not fully dynamic either since the concatenation of steps still requires them to start and end with zero speed. In [43], zero-speed connections are removed using a homotopy that continuously modifies the trajectory, making it faster and smoother. Dalibard et al. proved a controllability property of dynamic walking and used it in an original algorithm for walking and whole-body motion planning [12]. In a nutshell, the idea is that any statically balanced trajectory of the robot sliding on the ground can be approximated arbitrarily closely by a dynamic walking trajectory (but not by a quasi-static walking trajectory). This enables a two-phase planning that first considers only statically balanced trajectories, and then generates dynamic walking motions.

Another major issue is reactiveness, which is truly of key importance in changing environments. As seen in section 50.3, some approaches focus on reactively planning sequences of footsteps [15], which raises the following difficult question: until when can a planned footstep be modified, and how? For instance, while the swing foot is moving, it might be impossible to suddenly decide to change a forward step into a side step, or a backward step. In fact, due to inertial effects, it may even be difficult to modify significantly a planned step during the execution of the previous step. Therefore, when an obstacle appears or moves, reacting accordingly can only be done with a gradual modification of the planned walking motion. This is similar to mobile robots with bounded acceleration or bounded curvature during turns, except that the inertial effects involved in bipedal walking are much more complex, and it is difficult to know precisely what amount of trajectory change a walking robot could withstand. Simple models have been studied for this purpose, and, as far as reactiveness is concerned, the dynamics of running are surprisingly simpler than the dynamics of walking, which is one of the reasons why reactive motion strategies on spring-mass hoppers have been studied quite extensively (see for example [2]). In [47], a robust control policy uses knowledge about the dynamics of the 3D spring-mass model to constantly modify, in a feedforward manner, the position of the next footstep as the swing foot goes down, thereby removing the necessity to accurately predict the ground level. This controller can steer the system along reference trajectories with a high robustness to unknown changes in the ground height.

A particularly relevant notion in the field of reactive stepping is capturability, and it is presented in details in chapter 46, which discusses stepping strategies for balance recovery, a problem related to reactive footstep planning.

Dynamic footstep planning has also been studied by the computer graphics and animation community. Notable results include the work of Mordatch et al. [38] who introduced an algorithm for torque control that optimizes at each instant of the simulation the trajectory of a low-dimensional preview model based on the Spring-Load Inverted Pendulum (see chapter 23). It can plan and control walking and running motions on uneven terrain, is robust to external disturbance and can dodge projectiles. Although the transfer of animation techniques to physical platforms requires a lots of additional considerations (e.g. limitations in sensing and actuation), they are an important source of ideas and inspiration for future research in robotics.

Overall, planning walking motions while taking well and efficiently into account the robot dynamics remains a largely open problem, at least when the objective is to do it in real time. The main issue is that some dynamical effects important for smooth and robust walking (for instance related to the motion of the arms) are difficult to account for with simplified models (like the inverted pendulum). And with more precise models of the robot

dynamics, the problem of whole-body trajectory generation usually requires costly nonlinear optimization procedures, which makes the more complex problem of motion planning very hard to tackle in a computationally efficient way.

50.8 Vision-based footstep planning

To further improve reactiveness and adaptiveness, footstep planning should be tightly coupled with the ability to map the environment in real-time. Okada et al. [40, 41] used binocular stereo-vision and extracted planar surfaces from the 3D vision input to define safe regions for the robot feet. With an accuracy of about 10 mm on the position of these surfaces, they managed to combine their vision system to a 3D footstep planner.

Michel et al. [37] proposed a footstep planning algorithm using a floor map built from an external camera information, but most recent approaches aim at on-board sensing. The same authors and Kagami and Nishiwaki proposed in [36] a GPU implementation of a 3D tracking algorithm combined with heightmap-based footstep planning. Thanks to the computational power of the Graphics Processing Unit (GPU), the tracker can reliably recover the 6D pose of viewable objects relative to the robot. This enables to accurately and rapidly localize stairs and obstacles. A finite transition set and local adjustments are used to find a footstep path, and swing leg trajectories are defined based on computations of the convex hull of the terrain between successive locations for each foot.

Chestnutt et al. [11, 39] introduced a method for navigation in rough environments using a pivoting laser scanner creating point clouds from which 3D obstacles are extracted. To improve stability and the accuracy of measurements, the laser scanner is mounted on the torso of the robot with a swinging mechanism. A mixed-reality interface allows navigation control by a user who can easily define a path along which the robot will plan footsteps.

Self-localization is as important as mapping, and due to the frequent changes of direction and sway motions in bipedal walking, odometry is usually quite inefficient. Improvements in this field can be expected to follow advances in sensor fusion. In [22], a Monte-Carlo localization of the 6D pose of the torso is obtained by combining information from odometry, attitude and joint angle sensors, and a head-mounted 2D laser rangefinder. Maier et al. [34] extended this framework to use a head-mounted depth camera instead of the laser rangefinder, and used a static map as well as probabilistic updates of a local map to perform real-time navigation. In [35], a more advanced framework integrates perception, mapping, and footstep planning. As mentioned in section 50.4, for whole-body collision checking, a representation called inverse heightmap (IHM) is computed for each action of the robot (among a finite set of actions). It consists of a local grid storing at each cell the minimum height of any body part going above this cell while executing the action. IHMs are particularly well suited for efficient whole-body collision checking when the terrain is represented as a heightmap.

For more details on SLAM and vision-based humanoid navigation, the reader can refer to chapter 52.

50.9 Future directions and open problems

Over the past 15 years, significant advances have been made in footstep planning, and steady progress in autonomy, reactiveness and adaptiveness has been possible thanks to the application of techniques from various fields (planning, control, optimization, vision, etc.) and

to well organized combinations of different approaches, such as adaptive level-of-detail algorithms for high-level and low-level planning, finite transition sets with local adjustments, mixed-integer programming and model predictive control to perform footstep planning within control schemes, sensor fusion for real-time mapping and localization, etc.

Nevertheless, there remain many challenges that need to be overcome. The hybrid dynamics involved in bipedal locomotion (see chapter 29) should be better understood to allow a seamless insertion of footstep planning in reactive tasks such as balance recovery. The articulation between lower body and upper body motion planning should also be improved, to permit a better use of the upper body and arms, even when they are constrained by tasks such as carrying, pushing or pulling an object. The transitions from footstep planning to more general methods of multi-contact planning (see chapter 56) should also be handled with better efficiency. Like humans, humanoid robots use only their feet for standard locomotion, but in many exceptional cases (rough terrain, sudden balance perturbations, etc.), the robot needs to be able to extremely quickly plan contacts with potentially any part of its body. And the constant access to such complex behaviors should not be a computational burden for the robot.

In fact, as computational power continues to increase and becomes cheaper, footstep planning will probably be completely replaced by more general multi-contact planning methods. But in the long term, it will become more efficient to use entirely flexible hierarchies of motion planning. Depending on the environment and on the type of motion considered, algorithms will use different kinds of representations, developed over time, to first guess imprecise motion plans and then incrementally refine them. So the next biggest challenge may be the use of lifelong learning to continually improve the ability of the robot to plan walking motions. Ideally, the robot should be able to analyze its own experience, understand and even extend its stepping capabilities over time, and learn from its environment (with or without supervision) to guess what surfaces are possibly unstable or slippery, what obstacles can be pushed away, what objects can be used for support, how different doors should be operated, etc. In [50], optimization and learning are used to perform footstep planning on rough terrain with a quadruped. Similar approaches could pave the way towards learning-based planning of walking motions, but balance issues are more drastic with bipeds than quadrupeds, which makes failures more critical and exploration of new behaviors more difficult. On the other hand, the numerous problems related to bipedal walking make it a very interesting testbed for machine learning, and both fields could benefit from an increasing effort in this direction.

References

1. G. Arechavaleta, J.-P. Laumond, H. Hicheur, and A. Berthoz. The nonholonomic nature of human locomotion: a modeling study. In *IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics*, pages 158–163, 2006.
2. O. Arslan, U. Saranli, and O. Morgul. Reactive footstep planning for a planar spring mass hopper. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 160–166, 2009.
3. Y. Ayaz, K. Munawar, M.B. Malik, A. Konno, and M. Uchiyama. Human-like approach to footstep planning among obstacles for humanoid robots. *International Journal of Humanoid Robotics*, 4(01):125–149, 2007.
4. J.-D. Boissonnat, O. Devillers, and S. Lazard. Motion planning of legged robots. *SIAM Journal on Computing*, 30(1):218–246, 2000.

5. J. M. Bourgeot, N. Cislo, and B. Espiau. Path-planning and tracking in a 3D complex environment for an anthropomorphic biped robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2509–2514, 2002.
6. S. Candido, Y.-T. Kim, and S. Hutchinson. An improved hierarchical motion planner for humanoid robots. In *IEEE-RAS International Conference on Humanoid Robots*, pages 654–661, 2008.
7. J. Chestnutt and J. J. Kuffner. A tiered planning strategy for biped navigation. In *IEEE/RAS International Conference on Humanoid Robots*, pages 422–436, 2004.
8. J. Chestnutt, J. J. Kuffner, K. Nishiwaki, and S. Kagami. Planning biped navigation strategies in complex environments. In *IEEE/RAS International Conference on Humanoid Robots*, 2003.
9. J. Chestnutt, M. Lau, G. Cheung, J. J. Kuffner, J. Hodgins, and T. Kanade. Footstep planning for the Honda ASIMO humanoid. In *IEEE International Conference on Robotics and Automation*, pages 629–634, 2005.
10. J. Chestnutt, K. Nishiwaki, J. J. Kuffner, and S. Kagami. An adaptive action model for legged navigation planning. In *IEEE-RAS International Conference on Humanoid Robots*, pages 196–202, 2007.
11. J. Chestnutt, Y. Takaoka, K. Suga, K. Nishiwaki, J. J. Kuffner, and S. Kagami. Biped navigation in rough environments using on-board sensing. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3543–3548, 2009.
12. S. Dalibard, A. El Khoury, F. Lamiraux, A. Nakhaei, M. Taïx, and J.-P. Laumond. Dynamic walking and whole-body motion planning for humanoid robots: an integrated approach. *The International Journal of Robotics Research*, 32(9-10):1089–1103, 2013.
13. R. Deits and R. Tedrake. Footstep planning on uneven terrain with mixed-integer convex optimization. In *IEEE-RAS International Conference on Humanoid Robots*, pages 279–286, 2014.
14. A. El Khoury, M. Taïx, and F. Lamiraux. Path optimization for humanoid walk planning: an efficient approach. In *International Conference on Informatics in Control, Automation and Robotics*, pages 179–184, 2011.
15. J. Garimort and A. Hornung. Humanoid navigation with dynamic footstep plans. In *IEEE International Conference on Robotics and Automation*, pages 3982–3987, 2011.
16. J. S. Gutmann, M. Fukuchi, and M. Fujita. A modular architecture for humanoid robot navigation. In *IEEE-RAS International Conference on Humanoid Robots*, pages 26–31, 2005.
17. K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, and B. Wilcox. Motion planning for legged robots on varied terrain. *The International Journal of Robotics Research*, 27(11-12):1325–1349, 2008.
18. A. Herdt, H. Diedam, P.-B. Wieber, D. Dimitrov, K. Mombaur, and M. Diehl. Online walking motion generation with automatic footstep placement. *Advanced Robotics*, 24(5-6):719–737, 2010.
19. Y.-D. Hong, Y.-H. Kim, J.-H. Han, J.-K. Yoo, and J.-H. Kim. Evolutionary multiobjective footstep planning for humanoid robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 41(4):520–532, 2011.
20. A. Hornung and M. Bennewitz. Adaptive level-of-detail planning for efficient humanoid navigation. In *IEEE International Conference on Robotics and Automation*, pages 997–1002, 2012.
21. A. Hornung, A. Dornbush, M. Likhachev, and M. Bennewitz. Anytime search-based footstep planning with suboptimality bounds. In *IEEE-RAS International Conference on Humanoid Robots*, pages 674–679, 2012.

22. A. Hornung, K. M. Wurm, and M. Bennewitz. Humanoid robot localization in complex indoor environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1690–1695, 2010.
23. A. Ibanez, P. Bidaud, and V. Padois. Emergence of humanoid walking behaviors from mixed-integer model predictive control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4014–4021, 2014.
24. S. Kagami, K. Nishiwaki, J. J. Kuffner, K. Okada, M. Inaba, and H. Inoue. Vision-based 2.5D terrain modeling for humanoid locomotion. In *IEEE International Conference on Robotics and Automation*, pages 2141–2146, 2003.
25. O. Kanoun, J.-P. Laumond, and E. Yoshida. Planning foot placements for a humanoid robot: A problem of inverse kinematics. *The International Journal of Robotics Research*, 30(4):476–485, 2011.
26. S. Koenig and M. Likhachev. D* lite. In *National Conference on Artificial Intelligence and Conference on Innovative Applications of Artificial Intelligence*, pages 476–483, 2002.
27. J. J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue. Dynamically-stable motion planning for humanoid robots. *Autonomous Robots*, 12(1):105–118, 2002.
28. J. J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue. Footstep planning among obstacles for biped robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 500–505, 2001.
29. S. LaValle and J. J. Kuffner. Randomized kinodynamic planning. *The International Journal of Robotics Research*, 20(5):378–400, 2001.
30. T.-Y. Li, P.-F. Chen, and P.-Z. Huang. Motion planning for humanoid walking in a layered environment. In *IEEE International Conference on Robotics and Automation*, pages 3421–3427, 2003.
31. M. Likhachev, G. J. Gordon, and S. Thrun. ARA*: Anytime A* with provable bounds on sub-optimality. In *Advances in Neural Information Processing Systems, MIT Press*, 2003.
32. M. Likhachev and A. Stentz. R* search. In *AAAI Conference on Artificial Intelligence*, pages 344–350, 2008.
33. H. Liu, Q. Sun, and T. Zhang. Hierarchical RRT for humanoid robot footstep planning with multiple constraints in complex environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3187–3194, 2012.
34. D. Maier, A. Hornung, and M. Bennewitz. Real-time navigation in 3D environments based on depth camera data. In *IEEE-RAS International Conference on Humanoid Robots*, pages 692–697, 2012.
35. D. Maier, C. Lutz, and M. Bennewitz. Integrated perception, mapping, and footstep planning for humanoid navigation among 3D obstacles. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2658–2664, 2013.
36. P. Michel, J. Chestnutt, S. Kagami, K. Nishiwaki, J. J. Kuffner, and T. Kanade. GPU-accelerated real-time 3D tracking for humanoid locomotion and stair climbing. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 463–469, 2007.
37. P. Michel, J. Chestnutt, J. J. Kuffner, and T. Kanade. Vision-guided humanoid footstep planning for dynamic environments. In *IEEE-RAS International Conference on Humanoid Robots*, pages 13–18, 2005.
38. I. Mordatch, M. De Lasa, and A. Hertzmann. Robust physics-based locomotion using low-dimensional planning. *ACM Transactions on Graphics*, 29(4):71, 2010.

39. K. Nishiwaki, J. Chestnutt, and S. Kagami. Autonomous navigation of a humanoid robot over unknown rough terrain using a laser range sensor. *The International Journal of Robotics Research*, 31(11):1251–1262, 2012.
40. K. Okada, M. Inaba, and H. Inoue. Integration of real-time binocular stereo vision and whole body information for dynamic walking navigation of humanoid robot. In *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 131–136, 2003.
41. K. Okada, T. Ogura, A. Haneda, and M. Inaba. Autonomous 3D walking system for a humanoid robot based on visual step recognition and 3D foot step planner. In *IEEE International Conference on Robotics and Automation*, pages 623–628, 2005.
42. N. Perrin. From discrete to continuous motion planning. In *Algorithmic Foundations of Robotics X, Springer Tracts in Advanced Robotics*, volume 86, pages 89–104. 2013.
43. N. Perrin, O. Stasse, L. Baudouin, F. Lamiraux, and E. Yoshida. Fast humanoid robot collision-free footstep planning using swept volume approximations. *IEEE Transactions on Robotics*, 28(2):427–439, 2012.
44. N. Perrin, O. Stasse, F. Lamiraux, Y. J. Kim, and D. Manocha. Real-time footstep planning for humanoid robots among 3d obstacles using a hybrid bounding box. In *IEEE International Conference on Robotics and Automation*, pages 977–982, 2012.
45. N. Perrin, O. Stasse, F. Lamiraux, and E. Yoshida. Weakly collision-free paths for continuous humanoid footstep planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4408–4413, 2011.
46. A. Sherikov, D. Dimitrov, and P.-B. Wieber. Whole body motion controller with long-term balance constraints. In *IEEE-RAS International Conference on Humanoid Robots*, pages 444–450, 2014.
47. A. Wu and H. Geyer. The 3-D spring–mass model reveals a time-based deadbeat control for highly robust running and steering in uncertain environments. *IEEE Transactions on Robotics*, 29(5):1114–1124, 2013.
48. 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 International Conference on Advanced Intelligent Mechatronics*, pages 168–173, 2009.
49. E. Yoshida, I. Belousov, C. Esteves, and J.-P. Laumond. Humanoid motion planning for dynamic tasks. In *IEEE-RAS International Conference on Humanoid Robots*, pages 1–6, 2005.
50. M. Zucker, N. Ratliff, M. Stolle, J. Chestnutt, J. A. Bagnell, C. G. Atkeson, and J. J. Kuffner. Optimization and learning for rough terrain legged locomotion. *The International Journal of Robotics Research*, 30(2):175–191, 2011.