Search-Based Footstep Planning

Armin Hornung Daniel Maier Maren Bennewitz

***Abstract*— Efficient footstep planning for humanoid navigation through cluttered environments is still a challenging problem. Often, obstacles create local minima in the search space, forcing heuristic planners such as A* to expand large areas. Furthermore, planning longer footstep paths often takes a long time to compute. In this work, we introduce and discuss several solutions to these problems. For navigation, finding the optimal path initially is often not needed as it can be improved while walking. Thus, anytime search-based planning based on the anytime repairing A* or randomized A* search provides promising functionality. It allows to obtain efficient paths with provable suboptimality within short planning times. Opposed to completely randomized methods, anytime search-based planners generate paths that are goal-directed and guaranteed to be no more than a certain factor longer than the optimal solution. By adding new stepping capabilities and accounting for the whole body of the robot in the collision check, we extend the footstep planning approach to 3D. This enables a humanoid to step over clutter and climb onto obstacles. We thoroughly evaluated the performance of search-based planning in cluttered environments and for longer paths. We furthermore provide solutions to efficiently plan long trajectories using an adaptive level-of-detail planning approach.**

I. Introduction

Compared to wheeled robots, the greater flexibility of humanoid robots allows for unique capabilities such as climbing stairs and stepping over or onto objects instead of bypassing them. However, exploiting these capabilities during path planning is far more complex because many degrees of freedom have to be controlled. Planning whole body motions for navigation in the real world is computationally not feasible yet [1], [2]. A common solution is to use predefined motions to walk on a sequence of footstep locations, which reduces the problem to planning a sequence of collision-free footsteps. The search in the space of footsteps is hereby carried out either using A*-based methods including dynamic variants (e.g., [3], [4], [5]) or randomized methods [6]. While A* will find the optimal path for a given footstep parameterization and state discretization, its planning performance highly depends on the quality of the heuristic function that guides the search. Randomized methods find results fast but don't have any guarantees on the solution quality of the final path, which may be far from optimal. Furthermore, they often depend on smoothing the final path in a post-processing step [1].

In this paper, we introduce and compare several search-based footstep planning approaches. In case obstacles pose local minima in the search space, anytime planners can efficiently create sub-optimal solutions with guarantees on the solution quality. While it is desired to fully plan the path from start to goal to decide on its feasibility, it is often not necessary to initially find the optimal path since it can be further improved while the robot navigates. We first describe Anytime Repairing A* (ARA*, [7], [8]) for footstep planning and evaluate various heuristics for it. ARA* runs a series of weighted A* (wA*) searches, thereby efficiently re-using previous information. wA* inflates the heuristic by a factor w. As a second planner, which depends less on the heuristic function, we propose to use R* [8], [9], an A* search variant that combines wA* in a local deterministic search with a randomized component, thereby trying to speed up the search. In contrast to purely randomized approaches such as rapidly-exploring random trees, R* provides probabilistic guarantees on the sub-optimality of the solution.

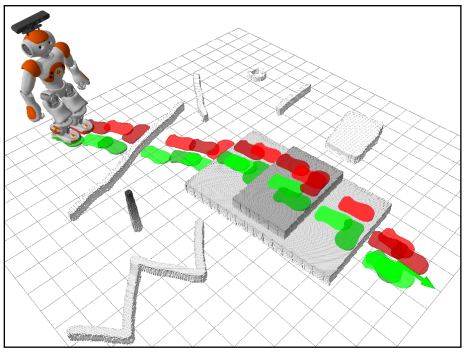


Fig. 1. Footstep planning enables a biped robot to traverse even difficult and three-dimensional terrain. Search-based planning hereby results in efficient and goal-directed plans.

By adding new stepping capabilities and accounting for the actual robot shape in the collision check, footsteps can be planned not only in planar 2D environments but also in 3D. This enables a humanoid to step over clutter and climb onto obstacles.

When planning long distances for navigation, the robot may encounter large open spaces where no obstacles are present. In these areas, a 2D plan is often sufficient and more efficient to plan than footsteps. To this end, we propose adaptive-level-of-detail planning [10] to efficiently combine coarse global path planning with detailed motion planning in areas requiring complex navigation capabilities. The advantages of this approach are two-fold. First, efficient paths can be found since the difficult regions can be passed instead of choosing detours around them, as a strictly two-dimensional planner would do. Second, the computational burden of globally planning detailed motions is seriously reduced.

All authors are with the Humanoid Robots Lab, University of Freiburg, Germany. This work has been supported by the German Research Foundation (DFG) under contract number SFB/TR-8 and within the Research Training Group 1103.

We present our extensions to search-based footstep planning and provide a comprehensive comparison of the planners to existing approaches. We thoroughly discuss the performance for different situations and heuristic functions. As the experiments demonstrate, we are able to plan even long footstep paths in short planning times. Due to search-based planning, the paths are directed towards the goal and have provable suboptimality bounds.

II. Related Work

In the last few years, several approaches to planning paths for humanoids have been presented. First approaches computed a local footstep path to follow a globally planned 2D path [11], [12]. The drawback here is that the globally computed 2D path may be rather inefficient since it does not consider actions to step over obstacles.

Gutmann *et al.* [13] and Candido *et al.* [14] use different motion primitives according to the type of terrain/passage and locally plan paths for regions based on the corresponding motion primitive. These approaches consider various motion behaviors but also do not adapt footstep locations to traverse objects, which may lead to sub-optimal behavior.

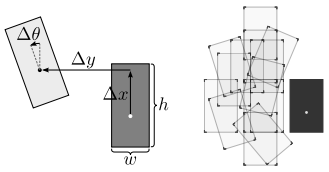
Chestnutt *et al.* [4], [5] were the first who proposed to plan footstep paths amongst planar obstacles using A* search. With a set of possible footstep actions, their humanoid was able to traverse obstacles by stepping over them. Garimort *et al.* [3] presented an extension of this approach and presented dynamic replanning of footstep paths to be able to efficiently replan paths, thereby reusing information from previous searches.

Perrin *et al.* [6] and Baudouin *et al.* [15] also investigated footstep planning and evolved it further to account for the 3D shape of the humanoid and the obstacles. They perform collision checks for the legs of the robot by precomputing swept volume approximations of the leg trajectories. These approaches use rapidly-exploring random trees (RRTs) to plan motions for humanoid.

Hauser *et al.* [1] apply probabilistic roadmaps (PRMs) to generate movements for humanoids, including stepping over actions or highly complex and computationally expensive whole-body motions. The drawback of all these randomized methods is that no bounds on the quality of the found solution can be given as it is the case for A*-based planning methods. To this end, RRT* provides probabilistic convergence to an optimal solution and was recently extended for anytime motion planning [16], albeit only for a wheeled vehicle so far.

Vernaza *et al.* [17] presented search-based planning for a quadruped robot. The authors compute global plans in a stance graph with R*. Compared to humanoid footstep planning, their quadruped only has point feet, and planning is performed in a relatively small area of rough terrain.

To the best of our knowledge, we present the first thorough evaluation of anytime search-based footstep planning with the aim of near-realtime plan results over long distances.



Parameter	Value / Range
w	10 cm
h	16 cm
Δx	$[-10, 22]$ cm
Δy	$[12, 28]$ cm
$\Delta\theta$	$[-0.23, 40.0]^\circ$

Fig. 2. Footstep transition model (left) and parameterization for a large humanoid such as HRP-2 or ASIMO, consisting of 14 different steps (middle and right).

III. Planning Framework

A. States, Actions, Transition Model, and Lattice Graph

In a planar world model, the robot's state is described by the global position and orientation of its stance foot $s = (x, y, \theta)$, alternating between the right and left foot.

A single footstep action is parameterized by the displacement of the moving foot (left or right) relative to the stance foot, as illustrated in Fig. 2. Under the constraint that humanoids move the left and the right leg alternating and that the possible footsteps for both feet are symmetric, a footstep action a can be parameterized by $a = (\Delta x, \Delta y, \Delta\theta)$ relative to the position of the stance foot.

When the footstep planner expands a state s, it determines all successor states s' resulting from applying the transitions t of each available action $a \in A$: $s' = t(s, a)$. The resulting states s' are checked for collisions with obstacles and invalid states are discarded.

The costs of a state transition from s to s' are given by the transition costs $c(s, s')$ that correspond to the traversed distance according to

$$c(s, s') = \|(x, y)^T - (x', y')^T\| + k \tag{1}$$

for $s = (x, y, \theta)$ and $s' = (x', y', \theta')$. Here, k are constant costs associated to executing one footstep, leading to a penalization of paths with a higher number of steps. We hereby assume that the effort of changing the orientation of a footstep can be neglected compared to the Euclidean distance.

The iterative construction of states (x, y, θ) connected by footsteps builds a sparsely-connected lattice graph, where the single footstep actions correspond to the motion primitives used in search-based planning [18].

The footstep set for a large humanoid robot used throughout this work is shown in Fig. 2 (middle). Since this represents a discrete set of stepping motions derived from the kinematic range of the humanoid, the goal location (given as a pair of footsteps or a single state) may not be exactly reached, or require too many intermediate steps to reach. Thus, we dynamically extend the footstep set with a transition to the goal if it is reachable from the current state.

B. Environment Model and Collision Checking

We use a 2D grid map consisting of equally-sized grid cells that are marked as either free or occupied. As in [10], occupied cells also contain traversability information, enabling the robot to differentiate between shallow obstacles that can be stepped over (such as uneven floor or clutter),

and obstacles that have to be avoided with a larger clearance (such as walls or furniture).

In order to validate a possible footstep, a planner needs to check if the footstep collides with an obstacle in the environment. Because this validation needs to be performed for all successors of an expanded state, it should be as efficient as possible. Here, we apply an efficient recursive collision check in the distance map [3] and assume that the humanoid's footprint is rectangular, or can be approximated by a rectangular bounding box. Extensions to 3D will be discussed in Sec. V.

C. Search-based Planning

On the lattice graph, search methods such as A* can be applied. The costs to transition between states are given from the transition model (1) and only valid states remain after collision checking. The search is guided towards the goal by a heuristic that can include different information.

Common heuristics for footstep planning are the straight-line Euclidean distance to the goal, or the estimated costs along a 2D path planned with A* or Dijkstra in a grid. The latter is potentially inadmissible because it does not reflect the humanoid's capability of stepping over obstacles [3], [4].

IV. Anytime Search-based Footstep Planning

A. Weighted A* Search

As a basis for both ARA* and R*, we will first recapitulate weighted A* (wA*) search. A* search expands states according to the evaluation function

$$f(s) = g(s) + h(s), \tag{2}$$

where $g(s)$ are the actual costs of the best path from the start to the current state s and the heuristic $h(s)$ provides the estimated costs to the goal from state s. $h(s, s')$ denotes the heuristic costs from a state s to s'. For the optimality of A* to hold, $h(s)$ must be admissible, i.e.

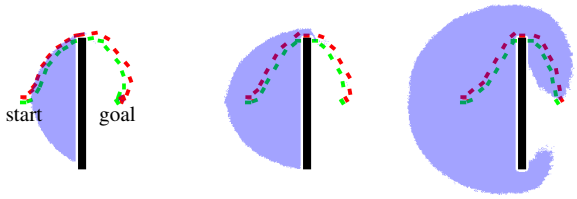
$$\forall s : h(s) \leq c(s, \text{goal}), \tag{3}$$

where $c(s, \text{goal})$ denotes the true costs of traversing from state s to the goal.

wA* inflates h with a factor $w \geq 1$ which results in suboptimal paths that can be found faster, thereby expanding fewer states. With this inflated heuristic, the quality of the solution can be traded off for efficiency, while the resulting paths are guaranteed to cost no more than w times the cost of an optimal path [7]. For $w = 1$, this corresponds to a regular A* search. As illustration, Fig. 3 shows the resulting footstep plans around an obstacle and the expanded states for different weights w.

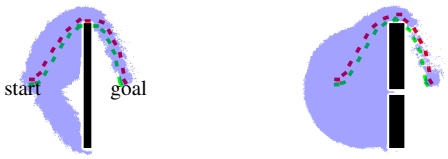
B. Footstep Planning with ARA*

Anytime Repairing A* (ARA*) search runs a series of wA* searches while efficiently reusing previous information [7], [8]. An initially large w causes the search to find a non-optimal initial solution quickly. Then, as time allows, the search reruns with incrementally lower values for w while reusing much of the information from previous iterations.



(a) $w = 10$ (b) $w = 2$ (c) $w = 1$ (A*)

Fig. 3. wA* and ARA* footstep planning (the paths and expansions are identical) around an obstacle for different heuristic inflation weights w. Here, the Euclidean distance to the goal was used as heuristic. The expanded state area is shaded in blue. An inflated heuristic results in fewer expanded states and faster planning times at the cost of suboptimal paths.



(a) initial $w = 10$, final $w = 1$ (b) initial $w = 10$, final $w = 1$

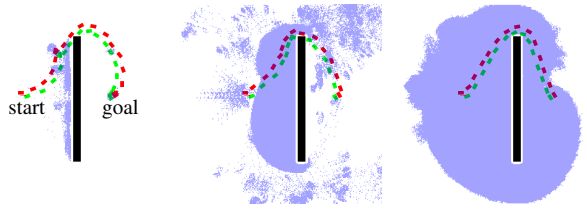
Fig. 4. ARA* planning around a local minimum with a 2D Dijkstra path as informed heuristic. In (a), planning succeeded after 1.8 s with 267 288 expanded states. However, the heuristic wrongly leads into a non-traversable narrow passage in (b) where planning succeeded after 15.0 s with 1 892 608 expanded states.

Given enough time, ARA* finally searches with $w = 1$, producing an optimal path. If the planner runs out of time before, the cost of the best solution found is guaranteed to be no worse than w times the optimal solution cost. This allows ARA* to find some solution faster than regular A*, and to approach optimality as time allows.

As any heuristic search, the efficiency of ARA* depends on the quality of the used heuristic. The more informed it is about the environment, the fewer states need to be expanded. The standard heuristic of the Euclidean distance to the goal will create local minima around obstacles that block the straight path to the goal, and forces the planner to expand large areas of the state space (Fig. 3(c)).

Compared to that, using a 2D Dijkstra path to the goal as heuristic is better informed of obstacles. This results in expanding states on paths around the obstacles instead of local minima (Fig. 4(a)), and it has been shown to result in more efficient planning times [3], [4]. However, this heuristic is potentially inadmissible and may even result in inefficient or no solutions at all in the case of footstep planning. It may also be susceptible to other types of local minima, again leading to inefficient planning as demonstrated in Fig. 4(b). The 2D Dijkstra heuristic will lead the planner to expand states through the narrow passage even though it is not traversable with stepping motions. With a larger obstacle inflation for the 2D heuristic path this specific problem could be avoided. However, this degrades the planning process to planning footsteps around a 2D path, resulting in non-optimal paths since the robot can no longer step through clutter [10].

All in all, ARA* is highly dependent on a well-designed heuristic function for efficient results.



(a) $w = 10$ (b) $w = 2$ (c) $w = 1$

Fig. 5. R* iterations for different heuristic inflation weights w. As heuristic the Euclidean distance to the goal was used.

C. Footstep Planning with R*

The randomized A* search (R*) aims to solve that problem by depending less on the quality of the heuristic function [8], [9]. The search avoids local minima by running a series of short-range, fast wA* searches towards randomly chosen sub-goals. The exploration of the search space with random sub-goals relates to randomized planners such as RRTs. But contrary to them, R* aims to minimize the solution cost and provides probabilistic guarantees of the solution suboptimality [9].

R* iteratively constructs a graph Γ of sparsely placed states in the same lattice graph as the dense search. At each iteration, a state in Γ is expanded by generating k random successor states at a distance Δ. Any goal state (as left or right foot) within Δ is added to the successors of s as well.

Each edge in Γ between two random states corresponds to a path in the original, dense search graph, which is determined by a local search with wA*. R* hereby first tries to find all "easy" local paths. If a local path requires too many expansions, planning it is delayed for a later stage when it is required to meet the suboptimality bounds. During the whole expansion from start to goal, R* provides probabilistic guarantees on the suboptimality of the solution by heuristic inflation w of the local wA* search. As in ARA*, R* iteratively lowers w and re-runs the search if a given planning time limit permits.

Generating the random nodes in Γ ensures the exploration of the search space. For footstep planning with humanoid robots, we randomly sample a direction and place a state s' at distance Δ from the current state. The leg of the state (left or right) is also chosen randomly, whereas its orientation is given by the initial random direction to favor forward walking. If s' is collision-free, it is added to Γ along with the edge $s \rightarrow s'$.

Examples for different weights in the scenario with a local minimum are shown in Fig. 5, using the Euclidean distance heuristic. At the beginning, R* sparsely samples the state space towards the goal and around the obstacle. As the heuristic inflation w approaches 1, the state space expansion grows more densely in order to find the optimal path. Only then the expansion resembles wA* and ARA* expansions.

V. Extensions to 3D

Up to now, the footstep transition model and collision checks were implemented in 2D for comparing the different planners. With 3D sensing, a robot can build an elevation map of its surroundings (Fig. 6, left). When planning in this representation, each footstep is associated a height based on the underlying terrain in addition to the 2D position and orientation. The set of footsteps needs to be extended with a valid height range to climb onto obstacles, and the complete body of the robot needs to be collision-checked against the environment over the robot's whole-body trajectory of executing one step. This ensures that the robot won't collide with an obstacle while stepping over it. For efficiency, this can be done with swept volumes of the robot's collision mesh model [6] or with a so-called inverse height map [19]. This representation contains the minimum clearance of the robot over the ground while executing a stepping motion. The inverse heightmaps are precomputed for each stepping motion, resulting in simple comparisons when applying them on the elevation map to check for collisions.

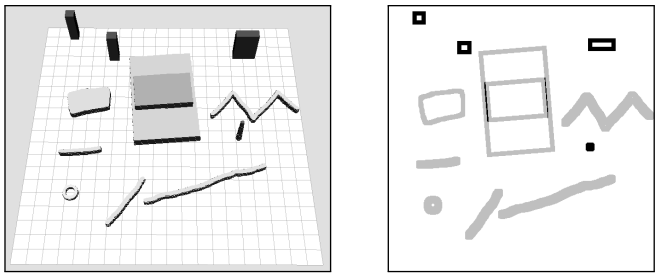


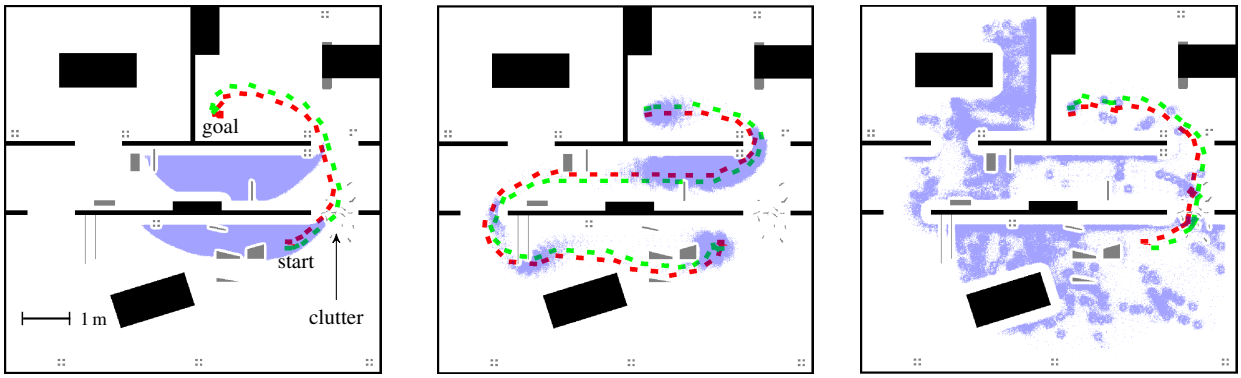
Fig. 6. Example 3D map (left) and the corresponding traversability map for the extended Dijkstra heuristic (right).

To account for stepping over obstacles of varying height, we introduce an extended Dijkstra heuristic that is informed of the environment, while still being admissible [19]. Obviously, small obstacles cannot be simply treated non-traversable since they could be stepped onto. In our traversability map for the extended Dijkstra heuristic (Fig. 6, right), we mark a cell as non-traversable (black) if the obstacle exceeds the robot's maximum step height. Small height changes result in increased costs for the cell (gray) to account for the additional time the robot needs to traverse the elevation. All other cells, i.e., planar surfaces, are considered free (white).

VI. Adaptive Level-of-Detail Planning

In large open spaces without obstacles, it is often not necessary to employ footstep planning. A 2D path on a grid representation can be easily computed, and followed by a humanoid with a corresponding walking controller. Close to obstacles, however, footstep planning offers a greater flexibility and results in more efficient plans since the robot can step close to or over obstacles. Thus, we propose to combine fast 2D planning in open spaces with footstep planning in areas containing obstacles [10].

We first classify the environment into regions of different complexity in an optional preprocessing step by segmenting the environment map, labeled as traversable and non-traversable obstacles, based on the humanoid's walking circumcircle for clearance (Fig. 8). This results in a set of open areas where 2D planning can be employed and obstacle



(a) ARA* with Euclidean heuristic after 92 s (b) ARA* with Dijkstra heuristic after 5 s (c) R* with Euclidean heuristic after 5 s

Fig. 7. Footstep planning with a time limit of 5 seconds through a cluttered passage. ARA* with the Euclidean heuristic fails to find a path within the time limit and requires 92 s to find a first solution with $w = 10$. The inadmissible Dijkstra heuristic results in a detour due to the clutter blocking the heuristic path. R* finds a path even with the Euclidean heuristic (final $w = 8$).

regions that contain planar obstacles as well as areas close to other obstacles. On the contour of the obstacle regions, we then sample transition points and apply a footstep planner between all transition points of one obstacle region. This estimates the traversal costs of the obstacle region for the planning stage.

Planning now corresponds to a search in the new state space, consisting of all 2D cells in the free areas and the sampled points on the obstacle contours. Transitions between all neighboring cells in the free areas are allowed, as well as between the sampled transition points. The costs are hereby given by the traversal estimates from the preplanning stage, or by a heuristic. As soon as the search reaches the goal, the segments crossing an obstacle region are converted into footsteps by using a footstep planner from the entry to the exit point of the obstacle region, illustrated in Fig. 8 (right). During execution, the robot then uses a walking controller to execute the planned footsteps and a velocity controller to follow the 2D path.

This approach results in efficient plans using the level of detail required for the type of region in the environment. By classifying regions as free, planar, or three-dimensional objects we can also avoid using expensive 3D footstep planning and collision checking when not needed.

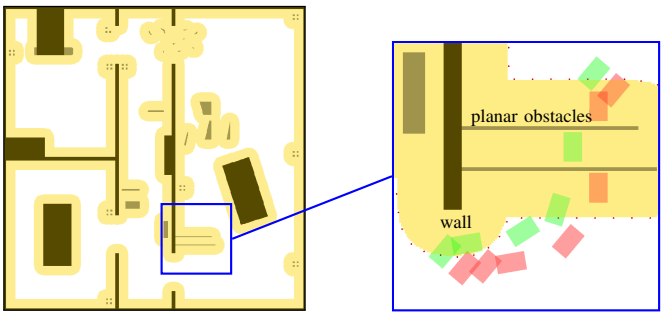


Fig. 8. Environment classification for adaptive level-of-detail planning: General obstacles are black, planar obstacles gray. The white area was classified to be suitable for 2D planning. In the yellow areas, a footstep planner can be used to plan efficient, collision-free paths. The close-up on the right shows sampled contour points (red dots) and an exemplary footstep path between two points.

VII. Experiments

A. Anytime Planning Results

In a first set of experiments, we evaluate and compare the performance of the anytime planners ARA* and R* for footstep planning. We chose a lattice graph resolution (discretization) of 1 cm for x/y and 5° for θ with the footstep parameterization from Fig. 2. This discretization is used to check equality when expanding states on the lattice. The occupancy maps for collision checks have a resolution of 1 cm and angles are preserved to be continuous in the collision check. The robot needs to maintain a clearance of 15 cm to obstacles marked as walls. The inflation radius for the 2D Dijkstra heuristic is the foot incircle radius of 5 cm. We experimentally determined Δ=1.5 m, k=20, and the expansion limit for easy-to-find paths as 500 for R*. All planning times are given for a single core of a desktop CPU (Intel Core i7, 3.4 GHz).

1) Indoor Environment with Limited Time: In a first experiment, we evaluated the algorithms qualitatively in an indoor environment containing several rooms with shallow obstacles and obstacles that have to be avoided (walls). To obtain near-realtime plans suitable for navigation, we set the time limit for the planners to 5 seconds and the initial heuristic inflation weight w=10. While the 2D Dijkstra heuristic works well in some cases, e.g., passing through the hallway, it is problematic for the start and goal configuration shown in Fig. 7. Here, the optimal path leads through a passage blocked by clutter that the robot can step over. A similar situation arises when there is a door sill that the robot should avoid stepping onto, which poses an obstacle completely blocking the doorway in 2D. In this scenario, ARA* with the 2D Dijkstra heuristic wrongly expands a non-optimal path despite w reaching 1.4 after 5 s. This is because the suboptimality guarantees no longer hold due to the heuristic inadmissibility, and demonstrates that the Dijkstra heuristic is a poor choice for footstep planning in cluttered environments. With the Euclidean distance heuristic, ARA* fails to find a plan within the time limit due to expansions being misled into local minima. In comparison, even with the Euclidean

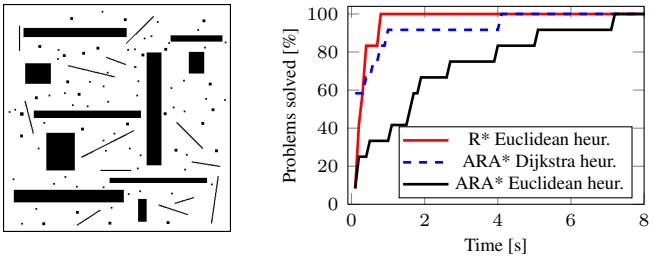


Fig. 9. A densely cluttered area (left) and the success rate for 12 random start and goal configurations in it (right). The percentage of solved problems within a certain time is shown for the first solution with w=5.

TABLE I

FOOTSTEP PLANNING PERFORMANCE FOR FIRST SOLUTIONS IN A DENSELY CLUTTERED ENVIRONMENT

Planner	Heuristic	Planning time [s]	Path costs
R* (w=5)	Euclidean	0.32 ± 0.23	16.45 ± 3.16
ARA* (w=5)	Euclidean	2.15 ± 2.21	13.57 ± 1.15
ARA* (w=5)	2D Dijkstra	0.56 ± 1.13	20.41 ± 5.08
Optimal: A* (w=1)	Euclidean	33.31 ± 15.00	11.06 ± 1.20

heuristic R* finds good initial solutions quickly.

2) Initial Plan Results in Dense Clutter: The next set of experiments is designed to evaluate how the different planners can cope with densely cluttered scenes such as the $4 \times 4\,\mathrm{m}^2$ area shown in Fig. 9 (left). To this end, we plan footstep paths between 12 random start and goal locations approximately 3.5 m apart and analyze the planning time and path quality. A* with the Euclidean distance heuristic requires on average 33 s to find the optimal paths containing 37 single steps on average. Fig. 9 (right) shows the success rate for the anytime planners to find the first solution with w=5 and Table I shows the aggregated statistics as mean and standard deviation.

As before, R* succeeds in finding fast results in general. ARA* with the admissible Euclidean heuristic requires more planning time even for initial suboptimal results, while it needs less time with the inadmissible Dijkstra heuristic, although it still requires more time than R*. However, the Dijkstra heuristic leads to longer paths, since it overestimates in some instances. All anytime planners find paths significantly faster than A*, at the cost of longer paths for the initial solution. Given enough planning time, both R* and ARA* with the Euclidean heuristic will converge to the optimal result from A* planning.

B. 3D Planning Results

1) Planning and Execution with a Nao Humanoid: When using the 3D footstep planning approach introduced in Sec. V for a Nao humanoid, we could extend the capabilities of the robot in cluttered environments. As demonstrated in Fig. 10, the robot can now step over small obstacles and onto stairs using footstep planning. To this end, we added a special stepping motion to the humanoid's footstep set so that it can step over an obstacle and a parameterized motion to climb up and down objects of different heights.

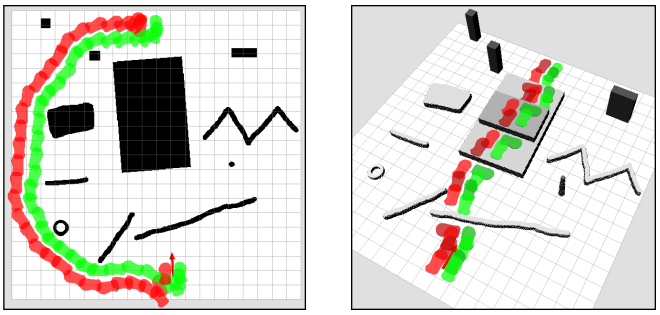


Fig. 10. 3D footstep planning results in more efficient navigation plans for a small Nao humanoid. Instead of walking around all obstacles in 2D (left), it can step over or even onto some of them (right).

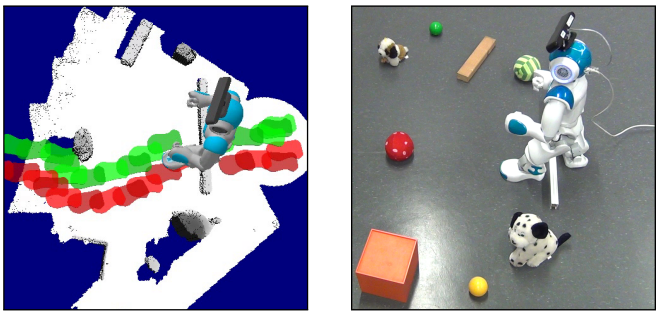


Fig. 11. 3D perception and footstep execution with a Nao humanoid.

2) Quantitative Evaluation of 3D Planning: To evaluate the performance of the footstep planner with 3D extensions, we randomly generated ten different maps of size 2.5 m × 2.5 m containing obstacles such as bars, platforms, and blocks of varying width, length, and height at a resolution of 4 mm. We randomly sampled ten different start and goal locations at similar distances, and evaluated the ARA* planner with time limits of 5 and 10 seconds and the two heuristics: Euclidean straight-line distance to the goal and the extended Dijkstra heuristic for elevation maps. All 100 planning problems could be solved within the given time limits of 5 s and 10 s, respectively. On average, it took 97 s to compute the optimal plan, whereas our anytime algorithm generates a first solutions within short time and afterwards improves the found motion plan. Table II shows mean and standard deviation of the path cost suboptimality (difference from the optimal solution) and the time for planning the initial solution with w=8. The extended Dijkstra heuristic leads to more efficient solutions that are closer to the optimal path compared to the Euclidean distance heuristic.

TABLE II

EVALUATION OF THE 3D FOOTSTEP PLANNER

Heuristic t-Limit	Dijkstra 10 s	Euclid 10 s	Dijkstra 5 s	Euclid 5 s
Suboptimality	1.03±0.05	1.12±0.14	1.07±0.10	1.19±0.19
$t_{\text{init\_sol}}$ [s]	0.74±0.86	0.57±0.51	0.72±0.83	0.55±0.5

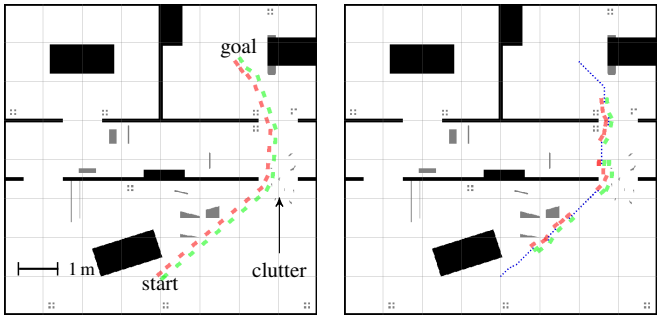


Fig. 12. Comparison between full footstep planning (left) and an adaptive combination of footstep and 2D planning (right) in an indoor office environment. In the latter, footstep planning is only employed in complex, cluttered regions and close to obstacles. Fast conventional 2D planning (blue dots) is used in the open spaces.

TABLE III

PERFORMANCE COMPARISON BETWEEN ADAPTIVE LEVEL-OF-DETAIL AND REGULAR PLANNING METHODS.

Approach	Planning time [s]	Path costs
2D grid planning	0.51 ± 0.11	19.18 ± 5.28
Footstep planning	43.70 ± 25.66	11.78 ± 1.16
Adaptive with precomputation	0.41 ± 0.32	11.78 ± 1.08
Adaptive, no precomputation	0.99 ± 0.54	12.41 ± 1.43

C. Adaptive Level-of-Detail Planning Results

In the large open areas of the indoor office environment, adaptive level-of-detail planning can provide a significant planning speedup. For this evaluation, we compare the results and planning time of the optimal solution with 2D path planning and footstep planning. A 2D-only path was fastest to compute in less than a second but resulted in a significant detour through the hallway since it could not pass the clutter. The global footstep plan (Fig. 12, left) took substantially longer to compute (29 s), but resulted in a more efficient path as planar obstacles are stepped over instead of walking around them. Compared to that, adaptive level-of-detail planning combines the advantages of both in that it is as fast as 2D planning and resulted in an efficient path (Fig. 12, right). Footstep planning was only invoked where needed. The path costs in this scenario were only 2% higher compared to the optimal footstep plan, while being 51% less compared to 2D planning.

For a statistical evaluation, we planned a 2D path, a footstep path, and a path using our adaptive method between ten different start and goal configurations in the environment. Each required a path length of approximately 8 m in the optimal case. As evaluation criteria in Table III, we used the planning time and execution costs for each plan (mean and standard deviation). Footstep planning yielded the best paths, but took up to 94 s to return the optimal solution in the most complicated scenario. Conventional 2D planning was faster, but the resulting paths were significantly longer as they cannot pass close to obstacles or step over objects. In contrast to that, our adaptive approach yielded fast results and lead to paths that were as efficient as full footstep plans.

Ideally, the environment is known and the costs to traverse obstacle regions can be estimated in a pre-processing step. When precomputation is not possible, e.g., because the environment is not completely static, a heuristic can be used to estimate the obstacle traversal costs. As can be seen in Table III, this resulted in 5% longer plans with a small increase in planning time since more non-relevant footstep segments were planned.

VIII. CONCLUSIONS AND FUTURE WORK

In this paper, we discussed search-based footstep planning for biped navigation. The anytime algorithms ARA* and R* enable planning efficient paths with provable suboptimality within short planning times. In the presence of local minima due to obstacles, we found ARA* with the 2D Dijkstra path heuristic in danger of yielding non-optimal paths. On the other hand, ARA* with the admissible Euclidean heuristic needs much more time since it expands too many states. R*, in contrast, yields fast initial solutions even with the Euclidean heuristic and avoids local minima. With 3D sensing and collision checking, footstep planning can be extended to 3D, thus enabling humanoids to reliably step over or onto objects. To this end, we introduced a novel admissible Dijkstra heuristic. For efficiently planning longer paths, we presented adaptive level-of-detail planning, which combines fast but coarse 2D planning in the open areas with detailed footstep planning through obstacle regions. Open problems for future work include building consistent, accurate and global 3D world models, as well as integrating a reactive walking controller for uneven terrain and external disturbances.

REFERENCES

[1] K. Hauser, T. Bretl, and J.-C. Latombe, "Non-gaited humanoid locomotion planning," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2005.

[2] K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, and B. Wilcox, "Motion planning for legged robots on varied terrain," *Int. Journal of Robotics Research (IJRR)*, 2007.

[3] J. Garimort, A. Hornung, and M. Bennewitz, "Humanoid navigation with dynamic footstep plans," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2011.

[4] J. Chestnutt, M. Lau, K. M. Cheung, J. Kuffner, J. K. Hodgins, and T. Kanade, "Footstep planning for the Honda ASIMO humanoid," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2005.

[5] J. Chestnutt, K. Nishiwaki, J. Kuffner, and S. Kagami, "An adaptive action model for legged navigation planning," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2007.

[6] 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*, vol. 28, pp. 427–439, 2012.

[7] M. Likhachev, G. Gordon, and S. Thrun, "ARA*: Anytime A* with provable bounds on sub-optimality," in *Proc. of the Conf. on Neural Information Processing Systems (NIPS)*, 2004.

[8] A. Hornung, A. Dornbush, M. Likhachev, and M.Bennewitz, "Anytime search-based footstep planning with suboptimality bounds," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2012.

[9] M. Likhachev and A. Stentz, "R* search," in *Proc. of the National Conf. on Artificial Intelligence (AAAI)*, 2008.

[10] A. Hornung and M. Bennewitz, "Adaptive level-of-detail planning for efficient humanoid navigation," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2012.

[11] 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 *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2005.

[12] T.-Y. Li, P.-F. Chen, and P.-Z. Huang, "Motion planning for humanoid walking in a layered environment," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2003.

[13] J.-S. Gutmann, M. Fukuchi, and M. Fujita, "A modular architecture for humanoid robot navigation," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2005.

[14] S. Candido, Y.-T. Kim, and S. Hutchinson, "An improved hierarchical motion planner for humanoid robots," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2008.

[15] L. Baudouin, N. Perrin, T. Moulard, O. Stasse, F. Lamiraux, and E. Yoshida, "Real-time replanning using 3d environment for humanoid robot," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2011.

[16] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the RRT*," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2011.

[17] P. Vernaza, M. Likhachev, S. Bhattacharya, S. Chitta, A. Kushleyev, and D. D. Lee, "Search-based planning for a legged robot over rough terrain," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2009.

[18] M. Pivtoraiko and A. Kelly, "Generating near minimal spanning control sets for constrained motion planning in discrete state spaces," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2005.

[19] D. Maier, C. Lutz, and M. Bennewitz, "Autonomous biped navigation in unknown 3D environments," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2013, submitted for publication.