Real-Time Footstep Planning in 3D Environments

Philipp Karkowski   Stefan Oßwald   Maren Bennewitz

***Abstract*— A variety of approaches exist that tackle the problem of humanoid locomotion. The spectrum ranges from dynamic walking controllers that allow fast walking to systems that plan longer footstep paths through complicated scenes. Simple walking controllers do not guarantee collision-free steps, whereas most existing footstep planners are not capable of providing results in real time. Thus, these methods cannot be used, not even in combination, to react to sudden changes in the environment. In this paper, we propose a new fast search method that combines A* with an adaptive 3D action set. When expanding a node, we systematically search for suitable footsteps by taking into account height information. As we show in various experiments, our approach outperforms standard A*-based footstep planning in both run time and path cost and, combined with an efficient map segmentation, finds valid footstep plans in 3D environments in under 50 *ms*.**

I. INTRODUCTION

The field of humanoid robotics has extended to a wide range of scenarios. Applications comprise logistics, help on construction sites, or aid during disaster scenarios. Humanoids also possess the potential to help in everyday life, e.g., by supporting elderly and disabled people. Their versatile movement capabilities allow for locomotion through complicated and cluttered areas, which might be impossible for wheeled robots. These abilities, however, come with the cost of challenging balance keeping and high-dimensional motion planning.

A difficult task that needs to be overcome is the development of a reliable locomotion controller. In a general scenario, using solely a motion controller is not sufficient, since it does not guarantee the validity of the robot's steps, which may cause the robot to lose balance or lead to undesired collisions between the robot and objects in the environment. On the other hand, many methods exist that are capable of planning a valid sequence of footsteps. Strategies include searches for global plans in 2D, which are generally more suitable for simpler obstacle avoidance on even terrain [1], [2]. Other approaches consider the more general problem of planning in 3D environments, which includes actions to step onto and over objects [3]–[6]. Computing a whole footstep plan to a distant goal location, however, entails the disadvantages that inaccurate execution of steps decreases the probability for validity of the plan with every step. Additionally, it does not consider sudden changes in the environment that may in general happen at any time and to which the robot needs to react instantly.

All authors are with the Humanoid Robots Lab, University of Bonn, 53113 Bonn, Germany. This work has been supported by the European Commission under contract numbers FP7-ICT-600890-ROVINA and FP7-610532-SQUIRREL and by the German Academic Exchange Service (DAAD) under contract number 57178163.

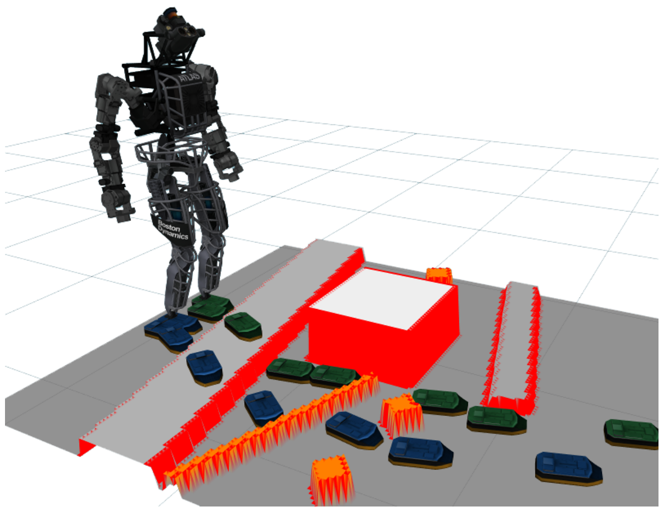
Fig. 1: Result of an adaptive footstep search for the stepping capabilities of the Atlas robot, developed by Boston Dynamics, Massachusetts, USA.

In our previous work, we have already successfully demonstrated a method for fast local mapping combined with a 2D geometric planning approach on a segmented height map that both work in real time [7]. In this paper we propose a new footstep planning method that couples the advantages of an A* search with an adaptive node expansion for 3D environments. Instead of relying on a fixed set during each node expansion, we instead systematically search for valid footsteps depending on the local environment. The benefit over our previous planning approach is the extension to 3D while using the same map segmentation approach. We perform a systematic search to find valid and efficient paths also through challenging scenes and achieve a planning frequency of 20 *Hz* in complex local environments.

II. RELATED WORK

Since the initial development of humanoid robots, many methods have been proposed that address the problem of how to choose the next stepping action to reach a certain navigation goal. In the following, we present the most prominent ones.

Michel *et al.* developed a method using A* to find full, global footstep paths [2]. While the A* search ensures optimal plans for a given set of actions, it may lead to computation times in the range of seconds for more lengthy paths, which does not allow reacting to sudden changes. Hornung *et al.* presented approaches that can more easily be adapted to finding a plan in a limited time frame using a combination of grid-based planning with A* and also by using ARA* and R* [1], [8]. While with these methods, footstep plans can generally be found more quickly, they are still not capable of planning in real time in all scenarios. In addition, all the mentioned techniques are restricted to be used in 2D which only includes planar obstacles.

Maier *et al.* applied ARA* in combination with precomputed inverse height maps for a given set of footsteps to allow for fast collision checking and stepping over and onto objects [5]. In this approach, the robot was required to frequently halt for a brief moment, to perform scan matching with ICP and replan the footstep plan if necessary.

Baudouin *et al.* proposed to use an approximate swept volume collision check [6] for stepping over objects and performed a footstep search based on rapidly exploring random trees (RRTs). Despite the ability for replanning without stopping the walk, RRTs are not goal directed and the method still takes seconds such that the replanning only starts a couple of steps ahead.

Chestnutt *et al.* published a technique that uses A* with a large set of footsteps in 3D environments [9], which results in low-cost paths. However, a large set of steps also leads to very long planning times in more complicated scenarios. The authors developed another method that uses an informed adaptive set of actions instead [10], which leads to low-cost paths that are found in a shorter time span. However, the planning times are still not real time capable and unlike our full systematic search they only adapt steps of a fixed footstep set if an instability is detected. Nishiwaki *et al.* integrated this adaptive A* search into a full navigation framework [11]. While the locomotion controller runs at a high frequency, the footstep planner itself is not included in these times.

Orthey *et al.* proposed a method for precomputing a large amount of the contact points between the robot and known objects with a given shape [12]. The approach combines the precomputed contact point information with a standard A* search and accomplishes a near real-time performance, however, introducing general obstacle shapes and combining it with stepping in 3D may lead to infeasible precomputation times. Hildebrandt *et al.* introduced swept-sphere-volumes for fast collision checks [13]. They created a framework that reacts to unknown obstacles in real time using an onboard RGB-D sensor where the walking direction is given by user input. The method does currently not allow for avoiding large obstacles or stepping onto new surfaces.

Others have applied more approaches that do not limit the planning problem to a fixed set of steps. For example, Stumpf *et al.* proposed to extend ARA* [5] to fit the 6 DoF pose of the footsteps to the environment using ground contact points while keeping the search space at 3 DoF [14]. While providing valid footstep paths in 3D environments, this technique directly operates on point clouds and leads to very lengthy computation times. Deits *et al.* approached the footstep planning problem in 3D by first finding convex planar regions [4]. Planning is then limited to the convex regions using quadratic programming and paths in simpler scenes can be found in the time frame of seconds. Fallon *et al.* presented a method for continuous planning in 3D [3]. Computing the

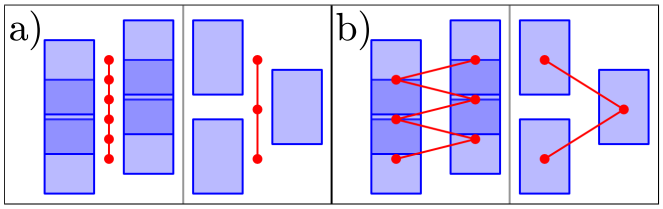

Fig. 2: If only the robot trajectory length (red) is minimized, but not the number of footsteps, for some situations very different plans lead to the same overall cost, i.e., both plans in a) have the same robot trajectory, however, the number of steps differs. If we define the cost as the distance between the subsequent footstep positions as in b), we simultaneously minimize the path length and the number of footsteps.

path takes a few seconds but they take advantage of the time the robot needs to perform a step. While continuous walk is possible without prior background knowledge of the environment, they rely on strong hardware including a GPU.

Our goal is to realize a real-time search by minimizing the number of expanded nodes during A* and adapting all footsteps to the environment to achieve low-cost paths.

III. FOOTSTEP-BASED A*

In A*, the quality of a plan depends on a suitable cost function for step execution. While it might seem that minimizing the trajectory length of the robot's center of mass, automatically minimizes the number of footsteps, this is not true in general. The humanoid can perform many short steps in the forward direction, which leads to a similar robot trajectory as a few larger steps (see Fig. 2 a)). It is possible to minimize both, the number of steps and the length of the trajectory, simultaneously by instead using the distance between subsequent steps as shown in Fig. 2 b).

Since the visual analysis of the terrain by a robot generally takes place along the forward direction, we prefer straight footstep plans that follow the visible region. Furthermore, we prefer longer footsteps in order to decrease the total amount of necessary step actions. Thus, we define the cost k for expanding a parent node at position $\mathbf{x}'$ and height map elevation e' by a child node at position $\mathbf{x}$ and height e as follows

$$k = ||\mathbf{x}' - \mathbf{x}|| + \nu \cdot |\theta| + \mu \cdot |e' - e|, \qquad (1)$$

where θ is the relative orientation between the steps. The coefficients ν and μ scale the cost of the relative rotation between subsequent steps and their height difference, respectively. The geometry is illustrated in Fig. 3 a).

In order to ensure optimality of the search, we need to define an admissible heuristic h. The lowest possible costs to reach the goal g based on the costs k as defined above are given by a path without relative rotations between steps, no height difference, and taking the largest possible steps. We define the approximate robot position of a footstep to its side at a distance that is half the feet separation s during straight walk as shown by the red dots in Fig. 3 b). The heuristic is then defined by

$$h = r \cdot \frac{\sqrt{f_0^2 + s^2}}{f_0}, \qquad (2)$$

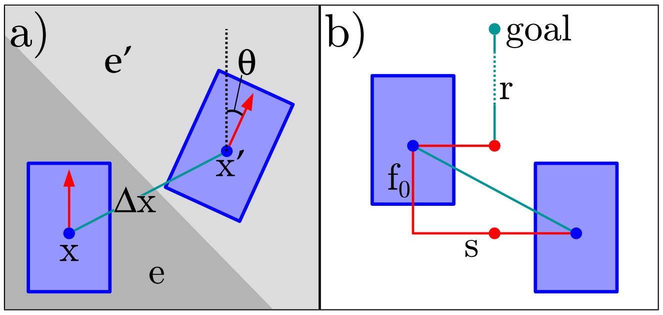

Fig. 3: Geometry of node expansion. a) $\mathbf{x}$ is the footstep position, e is the height of the corresponding planar region, and θ the relative orientation between the parent and child node. b) The heuristic given by Eq. 2 is based on the maximum forward stepping capability f_0, the feet separation during straight walk s, and the estimated distance from the robot to the goal r.

where r is the Euclidean distance from the approximated robot position of the footstep to the goal and f_0 is the maximum forward step distance. Fig. 3 b) shows that a maximal forward step moves the robot by a distance f_0 and leads to a cost increase given by $\sqrt{f_0^2 + s^2}$. Thus, we multiply the Euclidean distance to the goal by a factor that scales the robot movement distance to the actual cost increase for forward walk.

IV. A* WITH AN ADAPTIVE FOOTSTEP SET

For footstep planning, the expansion set consists of a selected number of footsteps which are all reachable from the previous parent step. Usually, a fixed set of steps is used [5], [8], [9], however, this can lead to cases in which no solution path can be found, e.g., when moving in a highly cluttered environment. An obvious solution for this issue is to extend the set to include steps with more variety. This, however, rapidly increases the search space which, in turn, increases memory usage and the search time.

Therefore, we propose a method that systematically searches for a small set of valid steps at every node expansion to keep the search space small while also ensuring a high success rate.

A. Reachability Map for Footstep Placements

In order to have quick access to viable footsteps relative to the previous step, we use a finely discretized reachability map that can be precomputed using inverse kinematics. The structure of this map is organized such as to suit our search strategy. Considering Fig. 4 the structure is as follows:

- Starting at the zero position Z located at the distance s to the side of the previous step, the reachability map provides the maximum displacement f_α for an angle α relative to the forward direction of the previous step.
- For a certain displacement on the line between Z and f_α the map provides the maximum displacements along the upward and downward directions e_{up} and e_{down}, respectively.
- Finally, depending on the footstep height in the range between e_{up} and e_{down} the reachability map provides the maximum possible rotation θ_{max} of the footstep relative to the forward direction of the previous step.

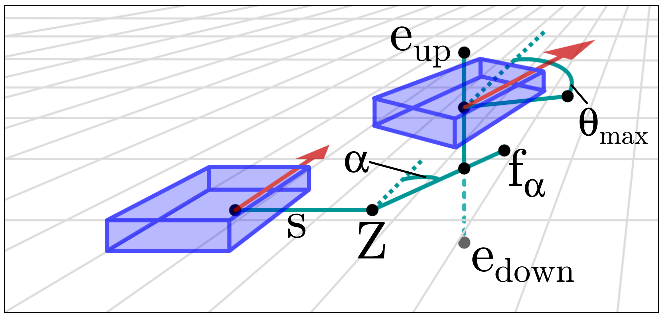

Fig. 4: The reachability map for footstep placements is based on rotations and displacements relative to the previous step, see text for details (Sec. IV-A).

Furthermore, we denote the rotation of the new step relative to the previous one θ.

B. Systematic Footstep Search

In each iteration, A* expands the node with the lowest value for $g + h$ by searching for a small set of valid steps. During this search the search state of a footstep is defined by three variables $\{\alpha, d, \theta\}$, where α was already defined above and d is the displacement distance from Z. We set the value of θ to

$$\theta(\alpha, d, e) = \min(\alpha, \theta_{max}(\alpha, d, e)). \tag{3}$$

As we are interested in a path with a preferably small number of steps and few relative foot rotations, this definition of θ helps in finding nodes that are as far as possible rotated in the direction of the node displacement while still considering the stepping capabilities of the humanoid. In the remainder of this paper, θ is given by (3) without stating its dependence on α, d, and e. The value for e is at any time automatically determined by the absolute position of the step within the height map such that no search along e is performed.

During the search, we carry out the following rough validity checks for the successor nodes:

- Check if the height difference between the current and the previous step is within the range of the reachability map.
- Check if the center falls onto a planar region, defined by a normal vector in the vertical direction.
- Check if the four corners of the footstep all lie within the same planar region. If this condition is not satisfied, an edge must be located somewhere below the footstep, making it invalid.

These checks do not fully guarantee the validity of the step, e.g., a small object could be located somewhere beneath it, but in most cases it provides a good first guess. Note that before expanding any node by additional steps, we fully analyze the footstep for validity first by checking for underlying cells in the height map. We then estimate the cells which are swept over from the previous step position to detect the maximum height along the footstep trajectory. If it exceeds a predefined height, i.e., the maximum height the robot can lift the foot over an obstacle, the step is also declared invalid.

The search for successor nodes starts at the state $\{0, f_0, 0\}$, i.e., the maximum forward step and proceeds as follows:

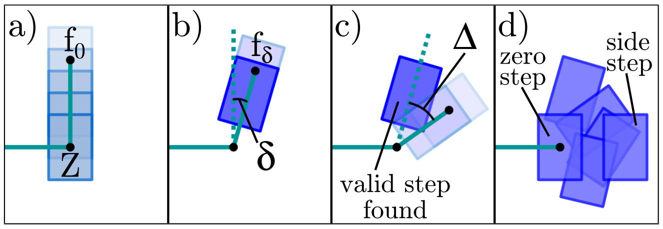

Fig. 5: Search during node expansion. We perform a search around the zero position Z by systematic rotation and displacement of the new step. A detailed explanation on how the search proceeds is given in Sec. IV-B.

1) If any of the above checks for the current search state fails, the displacement distance is decreased by an amount c, which is equal to the size of a single height map cell, e.g., for $\alpha = 0$ and $d = f_0$ the state changes from $\{0, f_0, 0\}$ to $\{0, f_0 - c, 0\}$. This procedure continues until we either find a valid step, or $d = 0$ as can be seen in Fig. 5 a).
2) Should no valid step be found during 1), i.e., for a given α, the search state changes according to $\{\alpha, d, \theta\} \rightarrow \{\alpha + \delta, f_{\alpha+\delta}, \theta\}$ (see Fig. 5 b)), where
$$\delta(\alpha) = \frac{c}{f_\alpha}. \tag{4}$$
The definition for δ originates from the *small angle approximation* and ensures that the position of any state $\{\alpha + \delta, f_{\alpha+\delta}, \theta\}$ is approximately one cell distance away from the position of the previous maximally displaced step $\{\alpha, f_\alpha, \theta\}$. The search continues with 1), but now with the new α.
3) Once a valid step has been found during 1), it is used for the expansion of the current parent node while the search state now changes by a larger value as given by $\{\alpha, d, \theta\} \rightarrow \{\alpha + \Delta, f_{\alpha+\Delta}, \theta\}$ which can be seen in Fig. 5 c), and we again continue with 1). The exact value for Δ is discussed further in Sec. V.
4) The search continues until we reach an angle α_{max} at which point the node search is finished. Finally, we add two additional fixed steps. First, a side step, i.e., $\{\pi/2, f_{\pi/2}, 0\}$, and secondly a zero step, if it has not already been added during the search for $\alpha = 0$. We found that a zero and side step are necessary in order to find paths around closely lying objects where forward walk is not always possible. An example of possible successors generated during node expansion with our method are illustrated in Fig. 5 d).

V. EXPERIMENTS

We evaluated our approach in simple and also complicated environments and compared the planning performance with A* based on fixed sets. For the A* search with fixed action sets, we used two different sets with a varying number of steps relative to the support foot, which we denote A*-Small (10 steps) and A*-Large (20 steps). Just as for our approach, we perform the same brief validity checks, introduced in Sec. IV-B, before adding footsteps to the priority queue. The footstep dimensions and stepping capabilities were chosen comparable to those of the Atlas robot and are

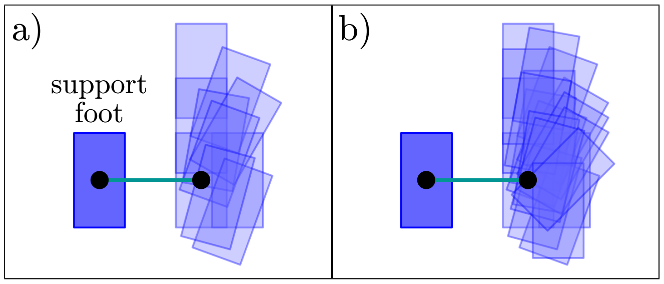

Fig. 6: Footstep sets for standard A*. a) A small set with ten steps used for a search with A*-Small. b) A larger set with twenty steps used for a search with A*-Large. Mirrored sets are used for the left foot.

shown in Fig. 6. We used a resolution of $c = 1.5\,cm$ for the height map.

By comparing planning results in free space and with single steppable objects, we experimentally determined suitable values for ν and μ to be 0.1 and 3, respectively (see Eq. 1). We chose these values such that the path was preferably straight and only leads over steppable obstacles if a detour around them requires multiple additional steps.

Since we plan locally in the space that lies ahead of the robot, we set $\alpha_{max} = 135°$ in order to exclude steps that go too far in the backward direction. We have also found that expanding more than ten steps at each node has a strong impact on the performance, especially for longer paths, which needs to be reflected in our choice for Δ. Additionally, since we prefer forward paths, it is advantageous to expand more discrete steps in the forward direction than sideways, hence, we set $\Delta = 10° + 0.5 \cdot \alpha$. In the case that α never needs to be increased by δ, i.e., the highest possible number of steps is expanded during the search, this choice of Δ leads to a maximum amount of eight expanded steps, including the zero and side step, which is smaller than the ten steps used for the smaller fixed set. Finally, our framework plans to a local goal point which can be determined by, e.g., a precomputed global path that does not take any obstacles into account. All computations were performed on a single Intel Core i7 3770 CPU.

A. World Representation

In order to achieve real-time footstep planning, we first need a suitable representation of the local environment around the robot. As already presented in our previous work [7], we use height maps that are segmented into planar regions with edges defined between adjacent regions. Fig. 7 illustrates one example of a segmented height map and the relative position to the humanoid within the local map.

B. Planning in Free-Space Environments

At first, we compared our approach to A* in environments with planar ground and no obstacles in order to show that our method does not present any disadvantages in simple scenarios. We planned a path to goal points on a circle with radius 3 m at different directions around the robot. The results are summarized in Table I. The results show that using our approach strongly outperforms A* on even terrain in terms of computation time for both the small and large

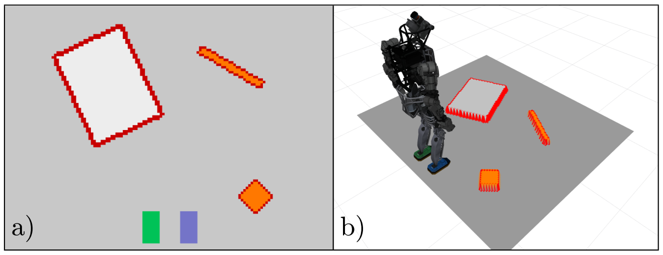

Fig. 7: Two visualizations of a segmented height map. a) Top view, b) 3D view. Planar regions are displayed in gray while regions that are either non-planar or too small to step on are orange. Edges (red) are defined between adjacent regions.

Rot.	**A*-Small**			**A*-Large**			**Our approach**		
	Time	*Cost*	*Exp. N.*	*Time*	*Cost*	*Exp. N.*	*Time*	*Cost*	*Exp. N.*
0°	0.09	4.10	10	0.10	4.10	10	0.09	4.10	10
15°	0.17	4.14	29	0.40	4.16	67	0.18	4.16	35
30°	0.38	4.17	84	0.42	4.17	58	0.12	4.18	17
45°	0.49	4.49	103	3.69	4.48	650	0.16	4.19	26
60°	1.75	4.62	399	1.01	4.50	176	0.32	4.62	53
75°	1.85	4.91	476	5.32	4.86	948	1.74	4.93	405
90°	18.92	5.43	5812	8.44	4.88	1510	2.73	5.15	682

TABLE I: Planning in free-space. Planning was done to a goal located on a circle with radius 3 *m* around the robot and at different relative orientations. *Exp. N.* denotes the number of expanded nodes and times are given in *ms*.

set while keeping very similar absolute path costs. This is mainly due to the fact that we expand fewer steps per node and no large search has to be performed, as all steps are free.

C. Planning through Cluttered Scenes

We performed further comparisons in various cluttered environments. In all cases, the planning region had dimensions $2.4\,m \times 2.4\,m$ and the local goal was located at the opposite side on the map relative to the robot. The choice of the map size corresponds to the approximate region that is visible by the depth camera used in the experiments with real data described in Sec. V-D. We compared our resulting footstep plan with both A*-Small and A*-Large for absolute planning time and the final path costs and summarized the results in Table. II. The resulting footstep plans are shown in Fig. 8. A* with a smaller footstep set has the disadvantage that highly cluttered scenes can lead to unnecessary footstep combinations; this is the case for planning with A*-Small in *map 1*. Both our approach and A*-Large led to a direct path through the clutter with similar costs, however, our approach resulted in a reduced planning time by over 75%.

While a humanoid has the potential to step upon obstacles, finding a low-cost plan across them by using a limited action set can lead to many additional steps, which are necessary to find suitable footstep combinations, especially if the steppable objects are narrow (see A*-Small in *map 2*).

Map	**A*-Small**			**A*-Large**			**Our approach**		
	Time	*Cost*	*Exp. N.*	*Time*	*Cost*	*Exp. N.*	*Time*	*Cost*	*Exp. N.*
1	85.68	7.61	28260	60.26	3.99	13835	14.34	3.70	1516
2	32.02	9.93	14118	32.22	4.98	7596	19.69	4.33	1520
3	67.23	6.91	22374	75.01	5.00	16805	48.74	4.34	5852

TABLE II: Planning in a local region with clutter. Planning was performed on maps with dimensions 2.4 *m* × 2.4 *m* to a goal point on the opposite side of the robot and are shown along with the final footstep plans in Fig. 8. *Exp. N.* denotes the number of expanded nodes and times are given in *ms*.

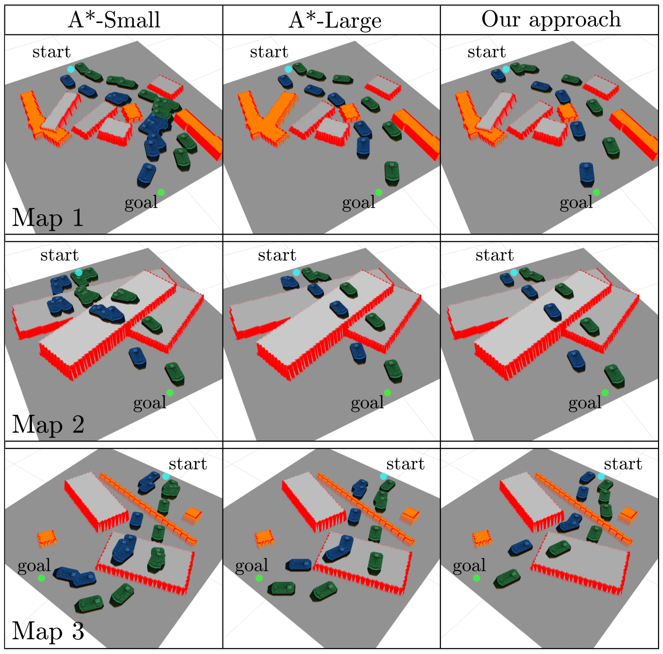

Fig. 8: Our approach compared to A* with fixed sets. A*-Small contains 10 and A*-large 20 fixed steps. As can be seen, our approach provides footstep paths with fewer steps and fewer relative rotations.

Our approach, however, was faster on both, *map 2* and *map 3*, and in both cases reduced the overall costs of the path.

D. Planning on Real Data

It is of high importance that a real-time footstep planner is capable of providing a high quality and valid footstep plan within a given time frame. During the path search, our algorithm keeps track of the node that is closest to the local goal point. Thus, we can start the search with a given maximum planning time and in case no plan fully reaches the goal in this time frame, our planner instead returns the path leading to the node that is closest to the goal. Our framework does a full planning cycle after every height map update to check whether a better plan can be found or to replace a footstep plan that becomes invalid after changes in the map. Thus, in general only the next single step of the plan will actually be executed by the robot after which further planning may replace subsequent steps.

In order to demonstrate that our approach works both on real data and in real time, we have combined it with our map segmentation framework and set the maximum planning time to 50 *ms*. For this experiment we used an *ASUS Xtion Pro Live* running with a resolution of 640 × 480 for perception of the environment and learning the height map [7].

The real scenario is depicted on the top-right of Fig. 9. For the experiment, we moved the wooden pallet rapidly in the way of the previously planned footstep plan. The remaining images show snapshots of how our footstep planner immediately finds new valid plans around and across the pallet. Even though we used a maximum planning time of $50\,ms$ the planner actually operated at a frequency of $110\,Hz$ on average during this experiment.

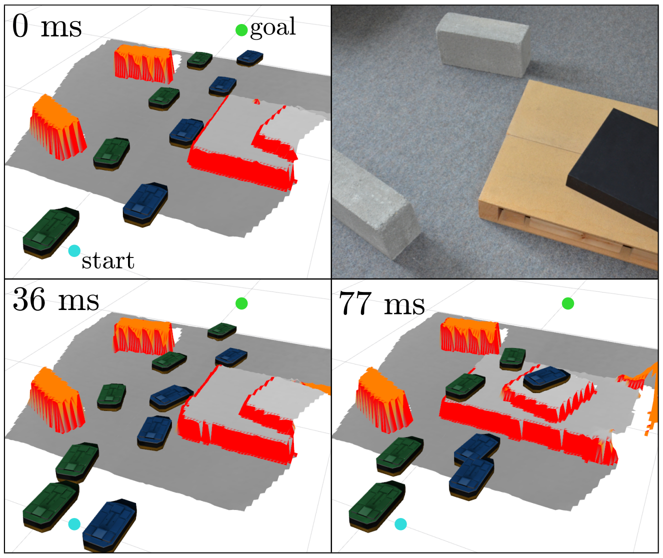

Fig. 9: Planning on real data. We combined our map segmentation [7] with the new footstep planner with a maximum planning time set to 50 *ms*. When suddenly moving a steppable obstacle in the way of the path, our planning immediately adapts the footstep plan to the newly detected changes. While the maximum planning time was set to 50 *ms*, the planner operated at 110 *Hz* on average in this scenario.

This experiment may be compared to a possible disaster scenario, e.g., moving through a collapsed house. Should any unstable debris suddenly fall in the path of the robot, our framework could immediately provide a new footstep plan helping the robot to maintain a continuous walk.

VI. Discussion

While we present a framework that has the capability of finding valid footstep paths in real time that considers the stepping capabilities of the humanoid robot, a locomotion controller is necessary to execute the steps. In the future, we will work on combining our planner with a suitable controller that takes into account the changes in the footstep plan in order to achieve smooth walking capabilities in dynamic environments. Such a framework can be combined with a coarse global path to a target location, which does not need to take into account non-static obstacles, to provide local sub goal points in the local map region.

As described in Sec. V-D, we keep track of the path that leads closest to the local goal point for those cases where the maximum planning time is exceeded. While this approach provides the next step to be performed, a possibility remains that an incomplete path might lead into local minima. The risk for this to happen is reduced by the fact that a newly planned path on an updated local region after performing the next step could already find a path around the local minimum.

Since our map segmentation framework provides normals for all planar regions, we can also easily extend our footstep planning to include inclined planes to allow for a 6D search.

VII. Conclusion

In this paper we presented a method for combining an adaptive footstep search with A* to efficiently find valid footstep paths through complex 3D environments. Our key idea is to perform a systematic and fast local search for valid steps during node expansion. In this way, we increase the success rate at finding nodes in complex scenes while keeping the overall expansion count small. We compared our method to other approaches that use a fixed set of footsteps. As we demonstrate in a variety of experiments, our approach outperforms A* with small or large fixed sets in both overall path costs and computation time. We combined our new search algorithm with our real-time map segmentation based on depth data [7] and showed that our framework can quickly react to sudden large changes in the environment and adapt the footstep plan accordingly.

Acknowledgements

We would like to thank Prof. Dr. Oskar von Stryk and Alexander Stumpf for providing us data on the approximate stepping capabilities of the Atlas robot.

References

[1] 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.
[2] P. Michel, J. Chestnutt, J. Kuffner, and T. Kanade, "Vision-guided humanoid footstep planning for dynamic environments," in *Proc. of the IEEE/RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2005.
[3] M. F. Fallon, P. Marion, R. Deits, T. Whelan, M. Antone, J. McDonald, and R. Tedrake, "Continuous humanoid locomotion over uneven terrain using stereo fusion," in *Proc. of the IEEE/RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2015.
[4] R. Deits and R. Tedrake, "Footstep planning on uneven terrain with mixed-integer convex optimization," in *Proc. of the IEEE/RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2014.
[5] D. Maier, C. Lutz, and M. Bennewitz, "Integrated perception, mapping, and footstep planning for humanoid navigation among 3D obstacles," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, 2013.
[6] L. Baudouin, N. Perrin, T. Moulard, F. Lamiraux, O. Stasse, 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.
[7] P. Karkowski and M. Bennewitz, "Real-time footstep planning using a geometric approach," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2016.
[8] 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.
[9] J. Chestnutt, J. Kuffner, K. Nishiwaki, and S. Kagami, "Planning biped navigation strategies in complex environments," in *Proc. of the IEEE/RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2003.
[10] 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.
[11] 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*, 2012.
[12] A. Orthey and O. Stasse, "Towards reactive whole-body motion planning in cluttered environments by precomputing feasible motion spaces," in *Proc. of the IEEE/RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2013.
[13] A. C. Hildebrandt, D. Wahrmann, R. Wittmann, D. Rixen, and T. Buschmann, "Real-time pattern generation among obstacles for biped robots," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, 2015.
[14] A. Stumpf, S. Kohlbrecher, D. Conner, and O. von Stryk, "Supervised footstep planning for humanoid robots in rough terrain tasks using a black box walking controller," in *Proc. of the IEEE/RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2014.