

# Planning Hybrid Driving-Stepping Locomotion on Multiple Levels of Abstraction

Tobias Klamt and Sven Behnke

**Abstract**—Navigating in search and rescue environments is challenging, since a variety of terrains has to be considered. Hybrid driving-stepping locomotion, as provided by our robot Momaro, is a promising approach. Similar to other locomotion methods, it incorporates many degrees of freedom—offering high flexibility but making planning computationally expensive for larger environments.

We propose a navigation planning method, which unifies different levels of representation in a single planner. In the vicinity of the robot, it provides plans with a fine resolution and a high robot state dimensionality. With increasing distance from the robot, plans become coarser and the robot state dimensionality decreases. We compensate this loss of information by enriching coarser representations with additional semantics. Experiments show that the proposed planner provides plans for large, challenging scenarios in feasible time.

## I. INTRODUCTION

Hybrid driving-stepping locomotion is a flexible approach to traverse many types of terrain since it combines the advantages of both, wheeled and legged, locomotion types. However, due to its high robot state dimensionality, planning respective paths is challenging.

In our previous work [1] we presented an approach to plan hybrid driving-stepping locomotion paths for our robot Momaro [2] even for very challenging terrain such as staircases with additional obstacles on it. The planner prefers omnidirectional driving whenever possible and considers individual steps in situations where driving is not possible. The individual configuration of ground contact points (robot footprint) is considered at any time. During planning, steps are represented as abstract manoeuvres which are expanded to detailed motion sequences before executing them. For small scenarios, this method generates high quality paths in feasible time with bounded suboptimality. Due to the high dimensionality of the robot configuration, the explored state space increases rapidly for larger scenarios and makes planning expensive. This effect is not unique for hybrid driving-stepping locomotion but affects high-dimensional planning in many applications such as locomotion planning for robots with tracked flippers or manipulation planning.

The search space can be reduced by choosing a coarser resolution or describing the robot and its manoeuvres in a more abstract way with less degrees of freedom (DoF). However, a fine resolution is key to navigate the robot

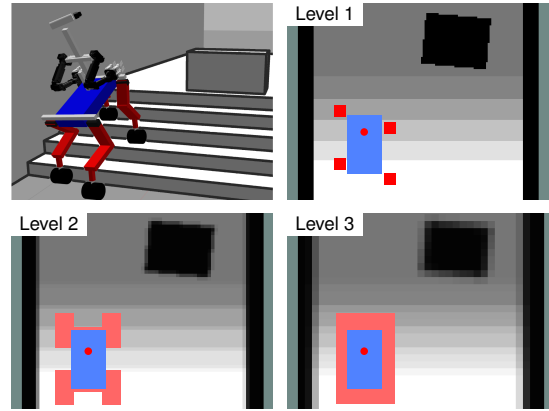


Fig. 1. Momaro on a staircase, visualized in three representation levels. Maps show terrain heights (olive = unknown).

precisely through challenging terrain. Moreover, only using a more abstract robot description is difficult, since the planning result shall be a path which can be executed by the robot with its given number of DoF.

Coarse-to-fine planning approaches [3], [4] address this problem by generating a rough plan first and refine the resulting path to the desired resolution and number of DoF in a second step. Especially in challenging, cluttered terrain, this procedure bears the risk of only finding expensive paths due to the lack of detail in the initial search.

We present a method which plans hybrid driving-stepping locomotion on three different levels of representation (see Fig. 1). In the vicinity of the robot, a representation with a high resolution and a high number of DoF is used to find paths which can be executed by the robot. With increasing distance from the robot, the resolution gets coarser and the robot is described with less DoF. These path segments are situated further in the future which comes along with a higher degree of uncertainty and less accurate sensor information. We compensate this loss of information for higher levels of representation by enriching the representation with additional semantics. All levels of representation are unified in a single planner. We further present methods to refine path segments into more detailed levels of representation. This decreases the number of necessary replanning steps. Replanning is only initiated if costs indicate that a situation is wrongly assessed in the coarser representation. In addition, we introduce a heuristic, based on the most abstract level of representation.

Experiments show that, compared to our previous work, this approach can handle much larger scenarios in feasible planning time while the path quality stays comparable.

## II. RELATED WORK

Multiple works addressed path planning for challenging environments, either by driving [5]–[7] or walking with quadruped robots [8], [9]. To our knowledge, there exist no approaches for hybrid driving-stepping path planning in challenging terrain, except our previous work [1].

A common idea to accelerate planning for larger scenarios is the usage of multiresolutional approaches. Behnke [10] proposed a general concept for A\*-based multiresolution planning with a decreasing resolution with increasing distance from the robot. González-Sieira et al. [11] apply high resolution in areas of high environment complexity. Resolution decreases with increasing distance from these areas. Similarly, Pivtoraiko et al. [12] apply different sets of state transitions to different areas of the environment. Bohlin [3] generates an initial plan in a coarse resolution first and refines this plan into a finer resolution. Since high resolution planning is only applied to parts of the map, the search space decreases and planning performance increases, compared to pure high resolution planning. One of the main challenges in multiresolutional approaches is the definition of feasible transitions between the different resolutions. All of the presented approaches face the problem that a coarse resolution representation neglects information and thus is not capable of representing challenging terrain features in sufficient detail—which might lead to wrong or bad plans.

Planning for systems with high-dimensional motion flexibility quickly reaches its limits for larger environments since the search space grows exponentially. Similar to multiresolution planning, several approaches utilize multiple representations with different planning dimensionalities to decrease planning complexity. Kohrt et al. [4] generate an initial plan in a low-dimensional search space and replan in the high-dimensional search space by only considering those states that are part of the low-dimensional plan. Gochev et al. [13] plan a path in a low-dimensional search space and only switch to high-dimensional planning in those areas where low-dimensional planning cannot find a solution. Similarly, Zhang et al. [14] plan in 2D and switch to high-dimensional planning in the robot vicinity and at key points. As described for multiresolution planning, planning with multiple robot configuration dimensionalities might lead to wrong or bad plans, since a low-dimensional robot representation might assess challenging situations wrongly.

To achieve further planning acceleration, it is an obvious idea to combine multiresolution and multidimensional planning. However, only few works, such as by Petereit et al. [15] address this. Different planning dimensionalities and resolutions are applied by using different sets of motion primitives. A fine resolution is only considered close to the start and goal pose and close to obstacles. A high planning dimensionality is considered for states which will be reached within a given time interval. This allows the planner to provide detailed plans close to the robot while planning times stay feasible. The drawbacks of both, multiresolutional and multidimensional planning also apply to this work.



Fig. 2. Our wheeled-legged robot Momaro is capable of omnidirectional driving (left) and stepping (right).

The platforms which are used in the presented works are quite limited in their configuration capabilities, compared to our robot Momaro. Our approach applies multiresolution and multidimensional planning to the challenging problem of hybrid driving-stepping locomotion. Furthermore, we compensate the loss of information for coarser resolutions and low-dimensional robot representations by enriching those representations with additional semantic features.

## III. HARDWARE

We use our mobile manipulation robot Momaro [2] (see Fig. 2). It offers omnidirectional driving through its four articulated legs ending in directly driven 360° steerable pairs of wheels. The unique design enables manoeuvres which are neither realizable by pure driving nor pure walking robots such as shifting a single foot while maintaining ground contact and thus changing the robot footprint under load. Active leg movements are restricted to the sagittal plane since each leg consists of three pitch joints.

Sensor inputs come from an IMU and a continuously rotating Velodyne Puck 3D laser scanner at the robot head which provides a spherical field-of-view. The laser-range measurements are registered and aggregated to a 3D environment map using the method of Droeschel et al. [16].

## IV. APPROACH

Input to our method is a height map with a resolution of 2.5 cm which is generated from the 3D environment map. In the vicinity of the robot, height information is very precise. With increasing distance from the robot, the accuracy decreases due to measurement errors. Planning is done on foot and body costs. The ground contact costs  $C_{GC}$  describe the costs to place an individual ground contact element (e.g., a foot or a foot pair) in a given configuration on the map.  $C_{GC}$  includes information about the terrain surface and obstacles in the vicinity. The body costs  $C_B(\vec{r}_b)$  describe the costs to place the robot base  $\vec{r}_b = (r_x, r_y, r_\theta)$  with its center position  $r_x, r_y$  and its orientation  $r_\theta$  on the map.  $C_B(\vec{r}_b)$  include information about obstacles under the robot base and about the terrain slope under the robot. The generation of  $C_{GC}$  and  $C_B$  from the height map varies between the different levels of representation and may contain several steps. Ground contact costs and body costs are combined to pose costs  $C(\vec{r})$  which describe the costs to place the robot in a given configuration  $\vec{r}$  on the map.



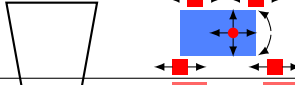



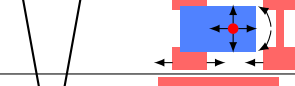





Level	Map Resolution	Map Features	Robot Representation	Action Semantics
1	 <ul style="list-style-type: none"> <li>• 2.5 cm</li> <li>• 64 orient.</li> </ul>	 <ul style="list-style-type: none"> <li>• Height</li> </ul>		 <ul style="list-style-type: none"> <li>• Individual Foot Actions</li> </ul>
2	 <ul style="list-style-type: none"> <li>• 5.0 cm</li> <li>• 32 orient.</li> </ul>	 <ul style="list-style-type: none"> <li>• Height</li> <li>• Height Difference</li> </ul>		 <ul style="list-style-type: none"> <li>• Foot Pair Actions</li> </ul>
3	 <ul style="list-style-type: none"> <li>• 10 cm</li> <li>• 16 orient.</li> </ul>	 <ul style="list-style-type: none"> <li>• Height</li> <li>• Height Difference</li> <li>• Terrain Class</li> </ul>		 <ul style="list-style-type: none"> <li>• Whole Robot Actions</li> </ul>

Fig. 3. The planner includes three levels of representation with decreasing resolution and robot configuration dimensionality. To compensate the loss of information, the semantics for both the environment representation and the robot actions increase.

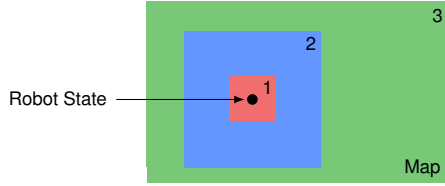


Fig. 4. Size and position of the different levels of representation. *Level 1* covers the vicinity of the robot. *Level 2* is also robot centered and medium sized. *Level 3* covers the whole map.

Path planning is realized through an A\*-search with anytime characteristics (ARA\* [17]) on pose costs. For the current search pose, feasible neighbour poses are generated during the search. They can either be reached by omnidirectional driving or by stepping-related motions. Stepping-related motions are only considered in the vicinity of obstacles where driving is infeasible. Steps are described as abstract steps, the direct transition between a pre-step to an after-step pose. The detailed motion sequence for steps is not considered during planning but generated before execution.

The environment and the robot are described in three different levels of representation with different sizes. In the vicinity of the robot, we use a fine resolution and a high robot configuration dimensionality for planning. We call this *Level 1* representation. With increasing distance from the current robot position, the environment and the robot are represented on higher levels with a coarser resolution and a robot representation with lower dimensionality. This is reasonable, since those parts of the plan are reached in the further future and thus are more uncertain. Moreover, sensor measurements become less precise with increasing distance from the robot. At the same time, we compensate this loss of detail by enriching the environment representation with additional features, which increase the understanding of the situation. Pose costs and robot actions use these semantic features. Higher levels of representation can be derived from lower levels of representation. The approach is visualized in Fig. 3. Level sizes and positions are shown in Fig. 4. For a planning task, the planner only performs a single planning run while including all three levels of representation. Hence, it is important that the same action carries the same costs in different levels of representation to make planning consistent

over all levels. Moreover, the transition between the different levels of representation is challenging. All three levels of representation and the transition between them are described in detail in the following sections.

The resulting path consists of segments in multiple levels of representation. As described before, the contained steps are abstract manoeuvres. Abstract steps in the initial path segment are expanded to detailed motion sequences before executing them. Roll and pitch motions of the robot base as well as single foot shifts stabilize the robot to perform each step safely. In addition, foot heights are derived. See our previous work [1] for more details.

Steps are only expanded for path segments in *Level 1* which is based on our previous work. For higher levels, representations are not detailed enough to derive concrete robot motions. As the robot executes the initial path segment, more measurements are made and a more detailed environment representation becomes available for path segments which have been represented in higher levels before. The path is updated with these updated representations. This can either be done by replanning the whole path or by transforming the respective path segments into more detailed representations, as described in Section IV-F. We call this coarse-to-fine transformation *refinement*.

#### A. Representation Level 1

*Level 1* is based on the approach which we presented in our previous work [1]. Input is a height map with a resolution of 2.5 cm. We derive local unsigned height differences between neighbour cells from this height map to generate ground contact costs for each individual foot. Base costs are derived from the height map itself. A height map and the derived foot costs can be seen in Fig. 5. In this level of representation, a robot pose  $\vec{r} = (\vec{r}_b, f_1, \dots, f_4)$  is represented through the robot base configuration  $\vec{r}_b$  and the individual longitudinal foot positions  $f_1, \dots, f_4$ . At each position, the robot can have 64 different discrete orientations.

Feasible driving neighbour poses can be found within a 20-position-neighbourhood and by turning on the spot to the next discrete orientation, as shown in Fig. 6. If the robot is close to an obstacle, additional stepping-related manoeuvres are considered which are visualized in Fig. 7. Those can be a discrete step, a longitudinal base shift manoeuvre,

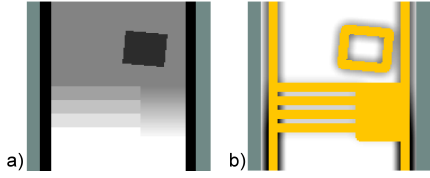


Fig. 5. a) *Level 1* height map showing a corridor with a flight of stairs, an untraversable steep ramp and an obstacle, b) respective foot cost map (yellow = untraversable by driving, olive = unknown).

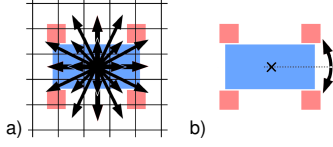


Fig. 6. Driving neighbour poses can be found by either a) straight moves with fixed orientation within a 20-position-neighbourhood or b) by turning on the spot to the next discrete orientation.

shifting individual feet forward or shifting individual feet towards their neutral position. We define the neutral robot pose as the pose visualized in Fig. 7 a, top. The costs for the presented manoeuvres are based on the foot and body costs, the individual robot elements induce during the manoeuvre.

As an extension of the previous work, we want the robot to align its orientation with the stair orientation, when climbing those. This is desirable, since the kinematic only allows for leg movements in the sagittal plane and since this behavior can also be observed when humans climb stairs by themselves or teleoperate robots to do so. If, after a stepping manoeuvre, the two front/rear feet have the same longitudinal position but stand on different heights, this indicates that the robot is not aligned with the stairs it climbs. By punishing such a configuration with an additional cost term, we achieve the desired behavior.

### B. Representation Level 2

We use the input height map with a resolution of 2.5 cm to compute the *Level 2* representation consisting of a height map and a height difference map with a resolution of 5 cm (see Fig. 9 a,b). According to the Nyquist-Shannon sampling theorem, subsampling has to come along with smoothing. To satisfy this theorem, we subsample the *Level 1* height map as shown in Fig. 8. Each *Level 2* height value is computed from

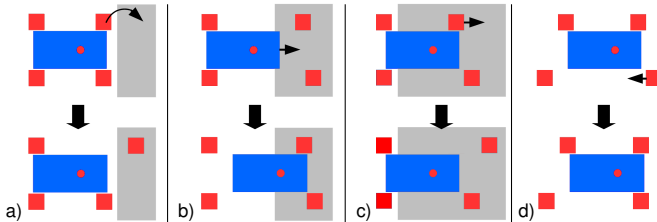


Fig. 7. *Level 1* stepping-related manoeuvres: a) Abstract step, b) longitudinal base shift, c) shifting a front foot forward, and d) shifting any foot back to its neutral position.

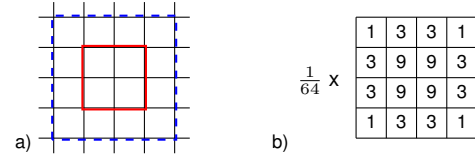


Fig. 8. Subsampling method: a) For a *Level 2* cell (red square) a  $4 \times 4$ -window (blue square) of *Level 1* cells is considered. b) Normalized binomial distribution to weigh heights and height differences.



Fig. 9. *Level 2*: a) height map, b) height difference map, c) foot area pair cost map for the orientation indicated by the red arrow.

the normalized, weighted sum of a  $4 \times 4$ -region of *Level 1* height values. We use a binomial distribution for weighing. A *Level 2* height difference map is generated in the same manner: We generate a *Level 1* height difference map by computing local height differences on the *Level 1* height map. This height difference map is then subsampled to a *Level 2* height difference map.

To decrease the robot configuration space dimensionality, we accumulate individual feet to pairs. This is intuitive, since we observe a tendency to pairwise foot movement in *Level 1* paths. Moreover, instead of describing each foot position precisely, we use foot areas as a more abstract description. We know, that a foot will be placed somewhere in the respective area but since the representation contains some time-related and measurement-related imprecision, a knowledge of the accurate foot position is not necessary. A *Level 2* robot pose  $\vec{r} = (\vec{r}_b, f_f, f_r)$  is consequently represented by its robot base pose  $\vec{r}_b$  and its relative longitudinal front and rear foot area pair coordinates  $f_f$  and  $f_r$ . Note that our platform and planner only allow sagittal leg movement. Lateral foot coordinates are fixed and thus a single variable is sufficient to describe each foot area pair.

We use the generated *Level 2* representation to compute ground contact and body costs. Body cost computation is similar to *Level 1* and only relies on height information. Ground contact costs

$$C_{GC,2} = 1 + k_1 \cdot \Delta H_{avg}, \quad (1)$$

where  $k_1 = 107$ , are costs to place foot area pairs on the map and are generated from the average height differences  $\Delta H_{avg}$  in the respective area. A *Level 2* foot area pair cost map can be seen in Fig. 9 c. Again, a punishing cost term is introduced for after-step poses with different average heights under neighbouring foot areas.

The robot actions are defined accordingly. Driving neighbours can be found similar to *Level 1* but with a doubled action resolution of 5 cm and 32 discrete robot orientations at each position. Additional stepping-related manoeuvres differ



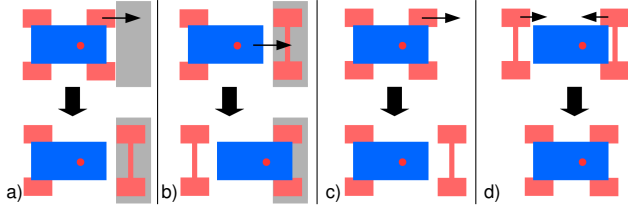


Fig. 10. *Level 2* stepping-related manoeuvres: a) Step, b) longitudinal base shift, c) move the front foot pair forward, and d) move any foot pair towards its neutral position.

from *Level 1* since the robot is only able to move foot pairs instead of individual feet. If the robot is close to an obstacle, it may step with a foot pair or perform another stepping related manoeuvre, as visualized in Fig. 10. To motivate stepping manoeuvres, we define a maximum height difference  $\Delta H_{\max, \text{drive}}$  for the foot area center coordinate which can be overcome by driving. Larger height differences only can be traversed by stepping.

The costs for such a foot pair manoeuvre are the concatenated costs for each individual foot action as described for *Level 1*. If, for example, the robot steps with its front foot pair as visualized in Fig. 10a, the costs for this manoeuvre are the sum of the costs for a step with the front left foot and a step with the front right foot. Since *Level 2* foot pair area costs differ from *Level 1* foot costs, we reparametrized the manoeuvre cost computation. We do this by performing foot pair manoeuvres in a variety of basic scenarios (e.g., drive/turn on a patch of flat/rough underground, step up different height differences, do a base shift) in both representation levels and manually tune the *Level 2* cost parameters until the costs for those manoeuvres in both levels vary by  $\leq 5\%$ .

During planning and execution, it is an important feature to refine *Level 2* path segments into *Level 1*. To refine a *Level 2* path segment between two successive poses  $\vec{r}_{2,i}$  and  $\vec{r}_{2,i+1}$ , we transform both poses into *Level 1* and generate a set  $S$  of feasible robot base poses by interpolating between  $\vec{r}_{1,i}$  and  $\vec{r}_{1,i+1}$ .  $S$  is then inflated with a radius of two position steps and one orientation step as visualized in Fig. 11. A local planner, which is restricted to  $S$ , searches for a *Level 1* path between  $\vec{r}_{1,i}$  and  $\vec{r}_{1,i+1}$ . If

- either one of the two poses becomes infeasible when transformed to *Level 1* because *Level 2* assessed the given situation wrongly or
- the costs for the refined *Level 1* path differs by  $> 25\%$  from the original costs for the path segment,

we call this path segment *not refineable*.

### C. Representation Level 3

We apply the described subsampling process (see Section IV-B) to generate a *Level 3* height map and height difference map with a resolution of 10 cm from the *Level 2* height map and height difference map. To increase the semantics of the environment representation, we categorize each *Level 2* map cell into one of the following terrain classes:

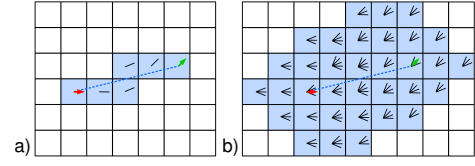


Fig. 11. Generating a set of feasible robot base poses for path refinement: a) For a given start ( $\vec{r}_{1,i}$ , red arrow) and goal ( $\vec{r}_{1,i+1}$ , green arrow) robot base pose, we generate a set of feasible robot base poses (black lines) by interpolating between the two. b) Inflation by two position steps and one orientation step.

- *flat*: easily traversable by driving,
- *rough*: traversable by driving with high effort,
- *step*: includes height differences which are too large to be traversed by driving but can be traversed by stepping,
- *wall*: occurring height differences are too large to be traversed by stepping, and
- *unknown*: cell cannot be classified.

First, we search for cells of the terrain type *step*. This is done by searching for cell pairs  $c_i$  and  $c_j$  that fulfill the following criteria:

- $\Delta H(c_i) < \Delta H_{\max, \text{drive}}$ :  $c_i$  is on a drivable surface,
- $\Delta H(c_j) < \Delta H_{\max, \text{drive}}$ :  $c_j$  is on a drivable surface,
- $\|c_i - c_j\| < 0.45 \text{ m}$ : The distance between  $c_i$  and  $c_j$  is within a maximum step length, and
- for the set  $T$  of cells  $c_k$  on the straight line between  $c_i$  and  $c_j$ ,  $C_{GC}(c_k) = \infty$  counts for all cells  $c_k \in T$ : A direct foot movement from  $c_i$  to  $c_j$  requires a step.

For all pairs of  $c_i$  and  $c_j$  which fulfill these criteria, each cell  $c_s \in c_i \cup c_j \cup T$  is assigned the terrain class *step*. In addition, we compute the angle  $\alpha_{i,j}$  between  $c_i$  and  $c_j$  and save it for  $c_s$ . Since most *step* cells are detected several times, we collect several angles for each cell.  $\alpha_{\text{avg},s}$ , the *mean of circular quantities* of these angles describes the estimated step orientation in  $c_s$ .

Second, we classify the remaining cells by their *Level 2* height difference value  $\Delta H^1$ :

- *flat* if  $\Delta H(c_i) \in [0 \text{ m}, 2 * 10^{-4} \text{ m}]$ ,
- *rough* if  $\Delta H(c_i) \in [2 * 10^{-4} \text{ m}, 0.05 \text{ m}]$ ,
- *wall* if  $\Delta H(c_i) \in [0.05 \text{ m}, \infty]$ , and
- *unknown* if  $\Delta H(c_i)$  is unknown.

The height difference intervals are tuned manually with respect to a maximum terrain height difference of 4 cm which can be overcome by driving and a maximum terrain height difference of 30 cm which can be overcome by stepping. The terrain class of a *Level 3* map cell is generated from the respective four *Level 2* cells by either choosing the terrain class with most members or, if this cannot be identified, the least difficult occurring terrain class.

Another source for terrain class segmentation can be camera images as shown in [18]. Fig. 12 a,b gives an example for a *Level 3* height map and terrain class map.

The *Level 3* robot representation  $\vec{r} = \vec{r}_b$  only consists of the robot base pose. Individual feet positions are not

<sup>1</sup>These height difference values are subsampled and smoothed and thus cannot be directly transferred to occurring height differences in the terrain.

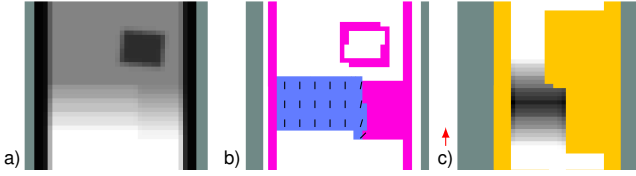


Fig. 12. *Level 3*: a) height map, b) terrain class map (white = flat, blue = stepping, pink = wall, black lines = step orientations), c) robot area cost map for the orientation indicated by the red arrow.

considered but we assume that the feet are somewhere in a ground contact area around the robot  $a_r$  (see Fig. 3). Hence, the robot is not able to perform foot or foot pair movements in this representation. The whole robot is rather moved over the terrain and traverses different terrain classes with different costs. Path search neighbour poses can be found similar to the driving neighbours described for *Level 1*. In this level of representation, the action resolution is 10 cm and the robot may have 16 different orientations at each position. When moving over *step* cells, a robot state is only feasible if the difference between the robot orientation and the step orientation of each *step* cell  $c_r$  is less than one discrete orientation step:  $\text{abs}(\alpha_{\text{avg},r} - r_\theta) < \frac{1}{16} \cdot 2\pi$ . Moreover, the robot is only allowed to move parallel and orthogonal to step orientations. These restrictions are required to enforce a behavior, which is induced by the robot kinematic in lower representation levels but not represented in *Level 3* otherwise.

Regarding cost generation, each cell  $c_i$  is assigned a cost value  $C_c(c_i)$  depending on its terrain class:

- *flat*:  $C_c(c_i) = 1.0$ ,
- *rough*:  $C_c(c_i) = 1.4$ ,
- *step*:  $C_c(c_i) = 76.0 + 2.95 \cdot \Delta H(c_i)$ ,
- *wall*:  $C_c(c_i) = \infty$ , and
- *unknown*:  $C_c(c_i) = \text{nan}$ .

The pose cost  $C(\vec{r})$  does not combine individual ground contact and body cost but averages the cost values of all cells in  $a_r$ . The described terrain class specific cell costs are manually tuned by comparing the cost of *Level 1* and *Level 3* manoeuvres for the same set of basic scenarios, as mentioned in Section IV-B. While constant values were sufficient for *flat* and *rough* cells, costs for stepping manoeuvres depend on the height difference to overcome. The presented computation method for *step* cells is required to keep cost differences for these basic manoeuvres  $\leq 5\%$ . A resulting robot area cost map can be seen in Fig. 12c.

*Level 3* paths can be refined to *Level 2* paths in the following way: As described for *Level 2*, we generate a set  $S$  of feasible robot base poses. In contrast to *Level 2*, we do not only consider two successive poses but the whole path segment  $\vec{r}_{3,s}, \dots, \vec{r}_{3,g}$  that needs to be refined at once. The first and last robot pose  $\vec{r}_{3,s}$  and  $\vec{r}_{3,g}$  of this *Level 3* path segment are transformed to a *Level 2* start and goal pose and a local *Level 2* planner, which is restricted to  $S$ , searches for a path between  $\vec{r}_{2,s}$  and  $\vec{r}_{2,g}$ . If a *Level 3* path needs to be refined to *Level 1*, *Level 2* is taken as an intermediate refinement step.

#### D. Level Transition

All three levels of representation are combined in a single planner, which chooses the lowest available level for each pose to provide the most detailed planning. Since planning in a low level of representation is slower, we provide *Level 1* data only in a small area around the robot position which is sufficiently large to plan the next manoeuvres. *Level 2* data is provided for a medium-sized region around the current robot position while *Level 3* covers the whole map.

The planner checks for each manoeuvre (e.g., drive into one direction, do a step, ...) if both, start and goal pose of this manoeuvre, are part of the same level of representation. If the goal pose is not part of the start pose level of representation, the start pose is transformed to the next higher level of representation and the same manoeuvre is replanned in this level if it is still available in this level. Note that the transformation of the start pose to the next higher level of representation might induce costs. Due to different map resolutions, the robot might be shifted to fit into the next level map cell and discrete orientation. Due to increasing foot restrictions, feet might be shifted to fit the next level robot representation (e.g., individual feet have to align within foot area pairs). We check each transformation for feasibility and generate costs from the occurring manoeuvre costs.

#### E. Heuristic

In our previous work, a combination of the Euclidean distance and the orientation difference was used as an admissible A\* heuristic (*Euclidean heuristic*). However, this heuristic does not consider the terrain which has large influence on the path costs. We propose a *Level 3*-based heuristic which includes such terrain features (*Dijkstra heuristic*).

After the goal pose  $\vec{r}_{1,g}$  is set, it is transformed to *Level 3*. We then start a one-to-any 3D Dijkstra search in *Level 3* starting from  $\vec{r}_{3,g}$ . Hence, we get for each *Level 3* pose a cost estimation to reach the goal pose. During path planning, we can estimate the costs from any robot pose to the goal by transforming it to *Level 3* and get the respective cost value.

Note that the quality of this heuristic strongly depends on the quality of the *Level 3* cost model in comparison to costs for the same manoeuvres in other levels of representation. Further note that we cannot prove that this heuristic always underestimates costs, which is necessary to prove admissibility for the generation of optimal paths. However, since we also utilize the suboptimal ARA\* algorithm, we do not aim to generate optimal paths for a given problem. In fact, we focus on generating paths with satisfying quality in feasible time. The performance of this heuristic is evaluated in Section V.

#### F. Continuous Refinement

As the robot moves along the initial path, the sensors provide new measurements and high-detailed environment representations are generated in the vicinity of the current robot position. We include these updated representations in the path by continuously refining the respective path segments, as shown in Fig. 13. If a cost difference  $> 25\%$  between the original and the refined path segments indicates

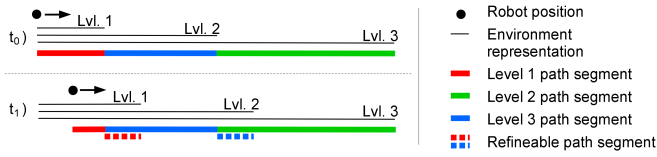


Fig. 13. As the robot moves along the path, the *Level 1* and *Level 2* representations move with it. Consequently, those path segments which are represented in a higher level and for which a more detailed representation becomes available, can be refined to this more detailed representation.

that the higher-level planning assessed a situation wrongly, we initiate a new planner run. With this approach, we can guarantee that path segments in the vicinity of the robot are always represented in *Level 1* and thus, included steps can be expanded and the result can be executed by the controller.

## V. EXPERIMENTS

We evaluate the proposed approach in two experiments. Both are done on one core of a 2.6 GHz Intel i7-6700HQ processor using 16 GB of memory. An additional video is available online<sup>2</sup> which also contains a Gazebo experiment to demonstrate the continuous refinement strategy.

A first experiment evaluates the planning performance of the different levels of representation individually and combined, as shown in Fig. 4. For this, we choose the *Level 1* size to be 3×3 m. This is sufficiently large to plan the next robot manoeuvres in high detail, but still small enough to avoid long high-dimensional planning. The *Level 2* size is chosen to be 9×9 m so that the *Level 2* path segment is about twice as long as the *Level 1* path segment. We utilized the *Euclidean heuristic* to compare the results to our previous work. The height map and a resulting path are shown in Fig. 14. Since we use an ARA\* algorithm which works with several heuristic weights  $\mathcal{W}$ , we evaluate the influence of these. Fig. 15 shows the planner performance.

It can be seen that planning on levels of representation  $> 1$  and with combined levels is faster by at least one order of magnitude compared to pure *Level 1* planning. The *Level 1* path for  $\mathcal{W} = 1.0$  could not be computed due to memory limitations. We distinguish between the path costs in the respective levels of representation (estimated cost) and the costs each path carries when refined to *Level 1*. Comparing the estimated costs to the refined *Level 1* costs gives an assessment about the quality of cost generation in each level of representation. The comparison of the refined *Level 1* costs to the original *Level 1* costs indicates the quality of the resulting path. It can be seen that the estimated costs always underestimate the refined *Level 1* costs. Especially for  $\mathcal{W} \leq 1.5$  the estimation is close with a difference  $\leq 7.7\%$ . Furthermore, the results show that for  $\mathcal{W} \leq 1.5$  the refined *Level 1* costs differ to the original *Level 1* costs by  $\leq 15\%$ .

In a second experiment, we compare the presented *Dijkstra heuristic* to the *Euclidean heuristic*. The scenario shown in Fig. 16 is larger and more challenging, compared to

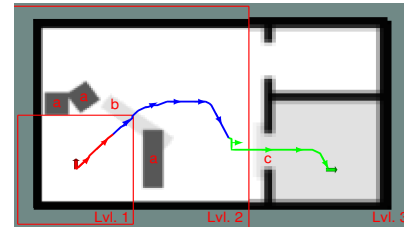


Fig. 14. Height map of the first experiment scenario. From its start position (red arrow), the robot needs to navigate between multiple objects (a), over a bar obstacle (b), step up to an elevated platform and through a door (c) to the goal pose (green arrow). The resulting path for  $\mathcal{W} = 1.125$  and combined levels of representation is shown. *Level 1* path segments = red, *Level 2* segments = blue, *Level 3* segments = green. Arrows show  $r_0$ .

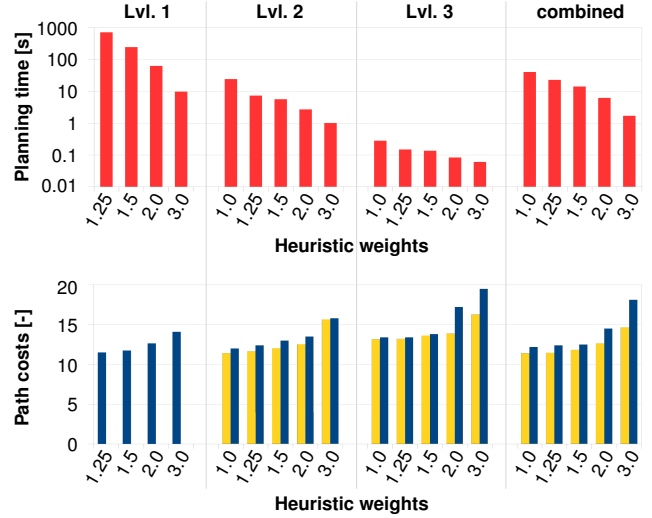


Fig. 15. Planning performance for different levels and different  $\mathcal{W}$  for the first experiment. estimated costs = yellow, refined *Level 1* costs = blue.

the first scenario. The starting pose is pose *a*. Planning is performed on combined levels of representation. A resulting path is shown in Fig. 17. Planning times and resulting costs are shown in Fig. 18. Preprocessing the *Dijkstra heuristic* took 0.52 s of the presented planning times. It can be seen that the *Dijkstra heuristic* further accelerates planning while the resulting costs stay comparable at least for  $\mathcal{W} \leq 1.5$ . E.g., for  $\mathcal{W} = 1.25$ , planning is accelerated by more than two orders of magnitude while the refined path costs only differ by 3.3%. Moreover, the resulting path illustrates how the robot aligns with the stairs and only moves parallel and orthogonal to them.

We finally compare the planner performance when started from different poses, as shown in Fig. 16. The results in Fig. 19 indicate that an important factor for the planner performance is the complexity of the planning within *Level 1* but higher  $\mathcal{W}$  lead to feasible performances in any case.

## VI. CONCLUSION

In this paper, we presented a hybrid locomotion planning approach which is able to provide plans for large scenarios with high detailing in the vicinity of the robot. We achieve this by introducing three levels of representation with de-

<sup>2</sup>[https://www.ais.uni-bonn.de/videos/ICRA\\_2018\\_Klamt/](https://www.ais.uni-bonn.de/videos/ICRA_2018_Klamt/)

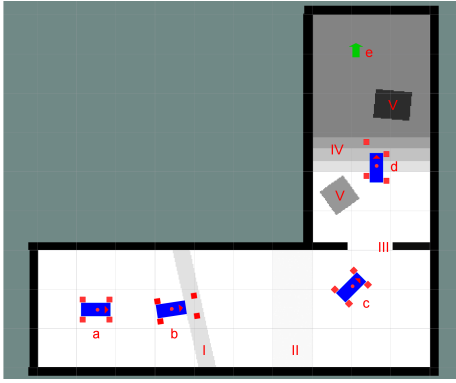


Fig. 16. Height map for the second experiment containing a bar obstacle (I), a rough area (II), a door (III), a flight of stairs (VI) and two obstacles (V). a - d are different starting poses for the planner, e is the goal pose.

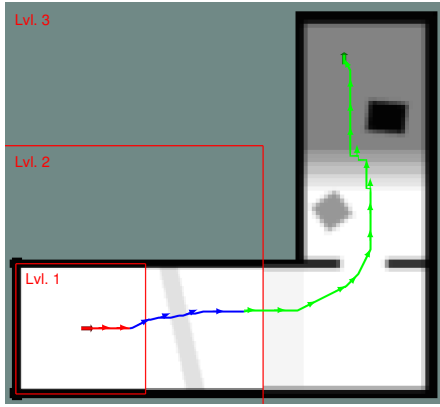


Fig. 17. Resulting path for planning with the *Dijkstra heuristic* and combined levels with  $\mathcal{W} = 1.25$ .

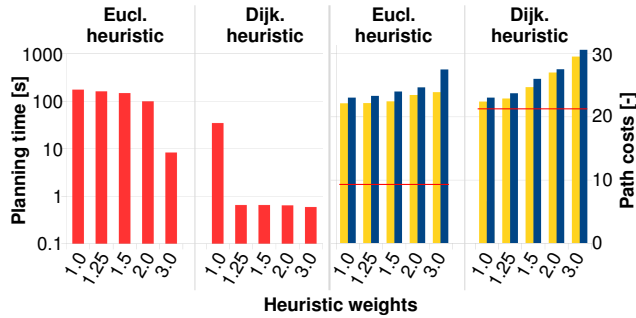


Fig. 18. Planning performance for combined levels of representation to compare the *Euclidean heuristic* with the *Dijkstra heuristic*. Red lines indicate the cost estimation for the path by each heuristic.

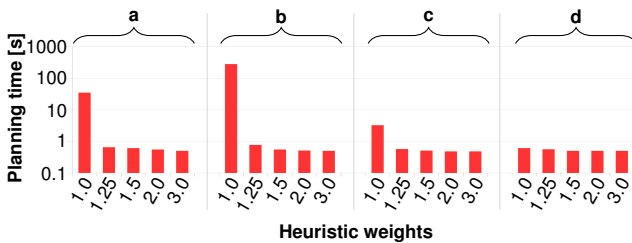


Fig. 19. Planning time for different starting poses (see Fig. 16) and different  $\mathcal{W}$ , using the *Dijkstra heuristic*.

creasing resolution and robot configuration dimensionality but increasing semantics of the situation. The most abstract level of representation can be used as a heuristic which poses a second acceleration strategy. Experiments show that the presented approach significantly accelerates planning while the result quality stays feasible and, hence, significantly larger scenarios can be handled in comparison to our previous work.

## REFERENCES

- [1] T. Klamt and S. Behnke, "Anytime hybrid driving-stepping locomotion planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.
- [2] M. Schwarz, T. Rodehutsors, M. Schreiber, and S. Behnke, "Hybrid driving-stepping locomotion with the wheeled-legged robot Momaro," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.
- [3] R. Bohlin, "Path planning in practice; lazy evaluation on a multi-resolution grid," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2001.
- [4] C. Kohrt, A. G. Pipe, J. Kiely, R. Stamp, and G. Schiedermeier, "A cell based voronoi roadmap for motion planning of articulated robots using movement primitives," in *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2012.
- [5] Z. Ziaei, R. Oftadeh, and J. Mattila, "Global path planning with obstacle avoidance for omnidirectional mobile robot using overhead camera," in *IEEE International Conference on Mechatronics and Automation (ICMA)*, 2014.
- [6] T. M. Howard and A. Kelly, "Optimal rough terrain trajectory generation for wheeled mobile robots," *The International Journal of Robotics Research*, vol. 26, no. 2, pp. 141–166, 2007.
- [7] M. Brunner, B. Brüggemann, and D. Schulz, "Motion planning for actively reconfigurable mobile robots in search and rescue scenarios," in *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2012.
- [8] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krüsi, R. Siegwart, and M. Hutter, "Navigation planning for legged robots in challenging terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.
- [9] N. Perrin, C. Ott, J. Engelsberger, O. Stasse, F. Lamirault, and D. G. Caldwell, "Continuous legged locomotion planning," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 234–239, 2016.
- [10] S. Behnke, "Local multiresolution path planning," in *RoboCup 2003: Robot Soccer World Cup VII*, Springer, 2003, pp. 332–343.
- [11] A. González-Sieira, M. Mucientes, and A. Bugarín, "An adaptive multi-resolution state lattice approach for motion planning with uncertainty," in *Robot 2015: Second Iberian Robotics Conference*, Springer, 2016, pp. 257–268.
- [12] M. Pivtoraiko and A. Kelly, "Differentially constrained motion replanning using state lattices with graduated fidelity," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008.
- [13] K. Gochev, B. Cohen, J. Butzke, A. Safonova, and M. Likhachev, "Path planning with adaptive dimensionality," in *Fourth Annual Symposium on Combinatorial Search*, 2011.
- [14] H. Zhang, J. Butzke, and M. Likhachev, "Combining global and local planning with guarantees on completeness," in *IEEE/RSJ International Conference on Robotics and Automation (ICRA)*, 2012.
- [15] J. Petereit, T. Emter, and C. W. Frey, "Mobile robot motion planning in multi-resolution lattices with hybrid dimensionality," *IFAC Proceedings Volumes*, vol. 46, no. 10, pp. 158–163, 2013.
- [16] D. Droschel, M. Schwarz, and S. Behnke, "Continuous mapping and localization for autonomous navigation in rough terrain using a 3D laser scanner," *Robotics and Autonomous Systems*, vol. 88, pp. 104–115, 2017.
- [17] M. Likhachev, G. J. Gordon, and S. Thrun, "ARA\*: Anytime A\* with provable bounds on sub-optimality," in *Conference on Neural Information Processing Systems (NIPS)*, 2003.
- [18] F. Schilling, X. Chen, J. Folkesson, and P. Jensfeld, "Geometric and visual terrain classification for autonomous mobile navigation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.