Footstep Planning in Rough Terrain for Bipedal Robots using Curved Contact Patches

Dimitrios Kanoulas[1], Alexander Stumpf[2], Vignesh Sushrutha Raghavan[1], Chengxu Zhou[1], Alexia Toumpa[1], Oskar von Stryk[2], Darwin G. Caldwell[1], and Nikos G. Tsagarakis[1]

***Abstract*— Bipedal robots have gained a lot of locomotion capabilities the past few years, especially in the control level. Navigation over complex and unstructured environments using exteroceptive perception, is still an active research topic. In this paper, we present a footstep planning system to produce foothold placements, using visual perception and proper environment modeling, given a black box walking controller. In particular, we extend a state-of-the-art search-based planning approach (ARA*) that produces 6DoF footstep sequences in 3D space for flat uneven terrain, to also handle rough curved surfaces, e.g. rocks. This is achieved by integrating both a curved patch modeling system for rough local terrain surfaces and a flat foothold contact analysis based on visual range input data, into the existing planning framework. The system is experimentally validated using real-world point clouds, while rough terrain stepping demonstrations are presented on the WALK-MAN humanoid robot, in simulation.**

I. INTRODUCTION

The need for humanoid robots to start operating reliably in unstructured environments is critical for their usability in real world tasks. Exteroceptive perception, e.g. visual and range, plays an important role in legged robot navigation and footstep planning. The most recently developed humanoid and bipedal robots, have shown impressive locomotion capabilities for various types of terrain (flat and rough), using low-level proprioceptive feedback control [1]. The state-of-the-art footstep planning systems use exteroceptive perception to navigate in uneven, but flat terrain surfaces [2], [3], while leaving the stabilization of small environment uncertainties to the controller. Thus, the need to plan a bipedal robot's footholds over rough terrain, i.e. on a rocky trail, remains an open problem in legged robot navigation.

In this paper, we propose a novel footstep planning framework that enables navigation over rough terrain surfaces (Fig. 1). The system builds on top of the search-based planning framework introduced in [4]. In particular, given lidar point cloud data as input, ARA* [5] is applied to produce the 6 Degrees-of-Freedom (DoF) footstep sequence in 3D uneven surfaces. The method works only when the environment has flat support surfaces for the feet positioning. To overcome this limitation, we use bounded curved contact patches to model foot-size local surfaces in the point cloud representation of the environment. We integrate the patch-based foot contact reasoning methodology introduced in [6] to produce (potentially partial) foothold poses in rocky terrains. This was left open in the previous work, where the operator had to pick manually the places where the foot contact analysis had to take place. In addition, it provides a novel solution to the autonomous footstep planning problem over rough surfaces, which is a critical part for some tasks such as rock climbing. The planner utilizes a gait generation walking module and a stabilizing controller [7] as black box, to handle the planned footsteps.

The system is implemented in C++ using the Point Cloud Library (PCL) [8], the Surface Patch Library (SPL) [9] (with the code available on our webpage), and the VIGIR footstep planning framework [4]. Next, we cover related work, followed by a summary of the flat-surface footstep planner and the patch-based contact reasoning (Sec. II). Then, we present the integration algorithm of the curved patch surface modeling into the search-based footstep planner (Sec. III). Last, we present experimental footstep planning results, using real-world point cloud data and locomotion demonstration in simulation, using the WALK-MAN humanoid robot.

A. Related Work

Bipedal locomotion has a long history in legged robotics navigation, especially in the direction of control and trajectory planning. During the last decade, the use of visual sensing enabled bipedal platforms to move towards autonomous footstep planning. Initially, the vision-based footstep planning methods were focused on the 2D cases, to allow bipeds navigate among objects. A graph-based planner for the ASIMO robot was successfully developed, using monocular camera data in [10] and [11]. A similar planner was used for 2D terrain navigation with the NAO mini-humanoid, using range sensing in [12] and [13]. In [14] a range sensing-based 2D planner for dynamically changing environments was introduced for the NAO.

The extension of bipedal footstep planners to 3D was first done for stair and obstacle climbing. In one of the earliest works in 3D, the humanoid robot HRP2 used stereo vision to plan horizontal stair climbing [15] and later the humanoid robot QRIO used range sensing to plan steps over inclined and elevated terrains [16]. In [17] and [18] planning horizontal obstacle climbing, using laser scanning, was introduced and applied on the HRP2 biped. In [5], [19], and [20], stereo vision and range sensing were used on NAO and HRP, respectively, to generate height maps

[1]Department of Advanced Robotics, Istituto Italiano di Tecnologia (IIT), Via Morego 30, 16163, Genova, Italy. {Dimitrios.Kanoulas, Vignesh.Raghavan, Chengxu.Zhou, Alexia.Toumpa, Darwin.Caldwell, Nikos.Tsagarakis}@iit.it

[2]Department of Computer Science, TU Darmstadt. {stumpf, stryk}@sim.tu-darmstadt.de

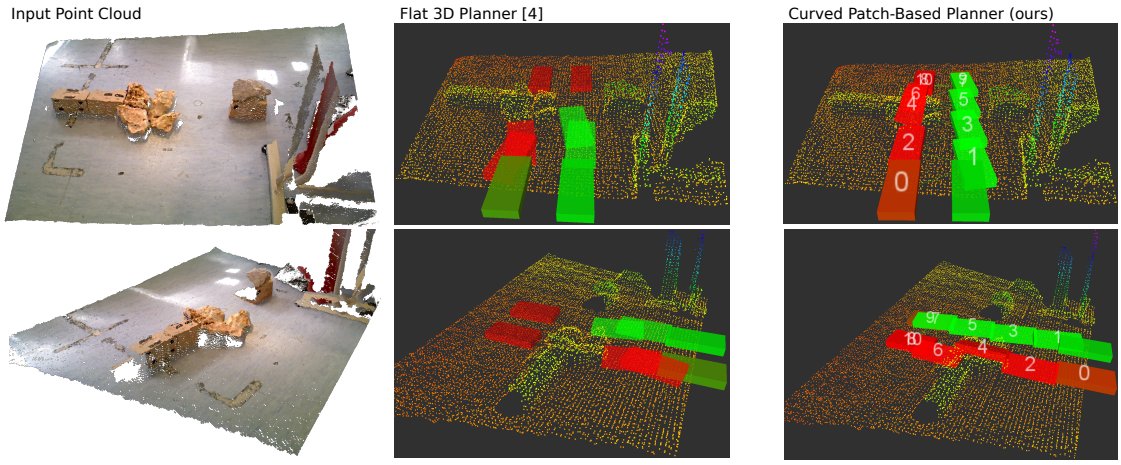

Fig. 1. The footstep planning over a rough curved terrain for the flat 3D planner [4] (middle: plan with non-suitable solutions) and our introduced system (right). In the latter, the non-numbered green/red poses are the start/goal planner poses, while the numbered green/red ones are the right/left foot footsteps.

for stepping on horizontal obstacles. Planar multi-contact foothold reasoning for a simulated HRP2, using point cloud data, was introduced in [21].

The DARPA Robotics Challenge (DRC) in 2015 played an important role in bipedal locomotion in outdoors environments. Humanoid robots gained capabilities to locomote under uncertainty, traversing 3D uneven flat terrains. In order to enable semi-autonomous crossing of such type of terrain, a set of path planners producing 6DoF footstep sequences were investigated. Stereo vision was used on the ATLAS humanoid robot [22] for optimization-based [2] footstep planning, where the operator determines manually obstacle free polygon regions. In [4] and [3] lidar sensing was used to achieve similar results, with a graph-based planning methodology and a heightmap/point cloud-based terrain modeling. A similar planner was used for a simulated ATLAS model and real range sensing data in [23]. The aforementioned 3D footstep planning algorithms were focused on uneven, but flat environments. In this paper, we try to extend a graph-based planner to handle also rough/curved surfaces, where partial footholds may be needed.

Towards rough terrain contact reasoning, point cloud data were used in [24] and [25] to determine the points of harsh foot impacts with the ground on a simulated HRP2 humanoid. These methods are focused on contact points for a single foothold and are not yet integrated into a path planner. In [26], a mini-biped performed a single foot placement on rocky surfaces, using curved patches [27] as terrain models. The contact planning system that was introduced in [28] for rocky trails assumes single foot contact point with the environment, while it was tested only in simulation, without visual sensing. In this paper, we extend a state-of-the-art graph-based footstep planner [4], [3] to handle contacts with rough surfaces, based on appropriate rough surface modeling through curved local patches.

Notice that significant research on footstep planning was done for four-legged [29]–[33], six-legged [34], or even hybrid [35] robots, but the single contact point between their feet and the environment enables different type of planning compared to the one we propose in this paper.

II. Flat Surface Footstep Planner

In this section, we briefly review the 3D footstep planning method for uneven flat terrain [4] that we use as the main planner framework for the proposed extension to curved surfaces, as well as the curved patch-based foot contact reasoning [6]. For more details, we reference the readers to the corresponding papers. We also identify the place where the planner will be extended.

A. Uneven Flat Terrain Footstep Planning

The footstep planner that was introduced in [4], will be the baseline system for our developments. It is able to produce footstep planning sequences on uneven, but only flat surfaces in the environment. We will extend this graph-based planner to handle also rough curved surfaces and potentially partial footholds. Thus, we first review the states, actions, and transition model of the planner. Then, we will describe the cost functions and the heuristics that are considered for the planning solution. We use the A* (ARA*) method [36] to find the shortest path from a starting to a goal position, expressed as a footstep sequence.

1) States, Actions, and Transition Model: We consider a 6DoF state space s, with the foot pose $\mathbf{s} = (x, y, z, \phi, \psi, \theta, f)$, where (x, y, z) is the foot position, (ϕ, ψ, θ) is its orientation as roll, pitch, and yaw values, and $f \in \{\text{left}, \text{right}\}$ foot. The transition t from a state $\mathbf{s}$ to a successive one $\mathbf{s}'$ applying an action $a \in A$ (footstep primitives), from a set of actions A, is a displacement in the pose of the supporting foot:

$$\mathbf{s}' = t(\mathbf{s}, a) = \mathbf{s} \xrightarrow{a} \mathbf{s}'. \tag{1}$$

The cost function $c(\mathbf{s}, \mathbf{s}')$ is similar to the one introduced in [13], but applied on the full step representation, i.e. considering the starting $\mathbf{s}$ and goal $\mathbf{s}'$ states of the moving foot, relative to the supporting foot $\mathbf{s}_0$. The set of actions A, for the footstep primitive set, are sampled discretely from a reachability polygon P next to the supporting foot $\mathbf{s}_0$.

2) Cost Functions and Heuristics: A hierarchical set of cost functions was used in the planner. The cost, representing

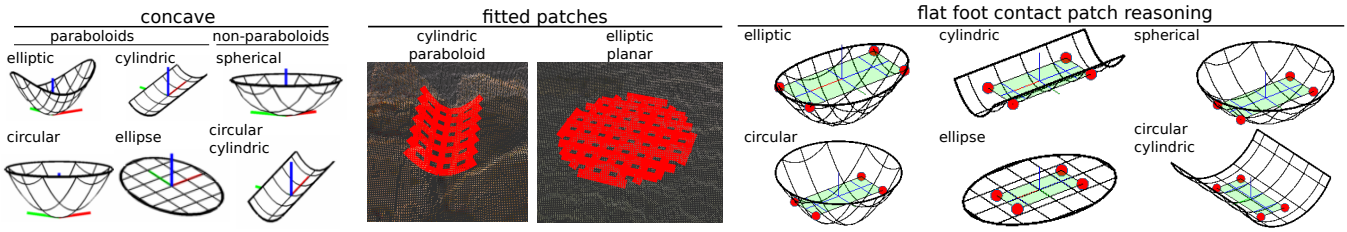

Fig. 2. Left: concave environment patch models, with their local coordinate frames. Middle: a concave cylindric paraboloid and an elliptic planar patch, fitted to 3D points. Right: the contact analysis between the environment patches (in black) and a flat foot patch (in green), with the contact points (in red).

the effort to execute a step sequence, is based on the following functions: 1) constant—generates minimal number of steps, 2) Euclidean—generates shortest paths, 3) boundary—eliminates experimentally determined violations that cause failures, 4) dynamics—generates low torso accelerations, and 5) ground contact—eliminates footsteps without enough terrain support. In parallel, the reachability polygon P is selected such that step feasibility, i.e. risk, balances the effort, i.e. cost. The anytime ARA* requires some heuristics for the remaining cost estimation, when local minima are found from the planner and is defined as follows:

$$\hat{h}(\mathbf{s}) = \|(\mathbf{s} - \mathbf{s}_{\text{goal}})\| + c_\theta \cdot \|\Delta\theta\| + c_{\text{step}} \cdot n_{\text{steps}} \quad (2)$$

where $\Delta\theta$ is the angle between θ and θ_{goal}, c_θ and c_{step} are constant angle and step costs, and

$$n_{\text{steps}} = \lfloor \Delta\hat{x}/\Delta x_{\max} + \Delta\hat{y}/\Delta y_{\max} \rfloor \quad (3)$$

is the minimum number of straight (with max distance $\Delta x_{\max}$) and side steps (with max distance $\Delta y_{\max}$) that are required to reach the goal, where $\Delta\hat{x}$ and $\Delta\hat{y}$ are the corresponding requested distances without rotations. The latter is used to penalize side-walking vs. forward stepping.

3) Collision Check, and Ground Contact Support: A 3D planner requires collision checking and ground contact estimation. The first one is solved in [4], [13] and considers full body and feet collision checking. The ground contact estimation is more important for this paper, since we propose an extension to its concept, such that rough curved terrain can be supported by the planner. The terrain is modeled as a height-map and each point in the captured point cloud is associated with a local normal. When a state s (i.e. foothold) is generated in the xy-plane, its z-value (i.e. foot center) is the corresponding vertical height of the height-map, while the roll ψ and pitch θ values come from the normal vector. Then, the foot is sampled equidistantly in order to check for: 1) collision and 2) ground contact support. Only when a sufficient amount of samples have ground contact support and no collision has been detected, the foot pose is considered as valid. This works perfectly when the ground is flat (depending also on the noise of the input point cloud), but when the surface is curved the foot either collides with the terrain (concave surfaces) or does not have enough support (convex surfaces). For this reason, step plans over rough curved surfaces cannot be supported by this planner. A new modeling for these surfaces and the foot contacts is required as will be described next. An example of a successful and a failing footstep planning case, due to these limitations, is shown in Fig. 1.

B. Curved Patch-Based Foot Contact Reasoning

Given the need to represent contacts between rough (curved) surfaces and flat foot soles, in [6] we introduced a contact reasoning analysis between curved environment patches that fit local point clouds on the terrain and flat patches that represent the foot itself. In this work, we use a subset of the introduced patches; those that have at least 3 non-collinear points of contacts (Fig. 2, right).

An environment patch of size slightly bigger than the length of the foot, is modeled as a bounded curved patch (paraboloid, spherical, or circular cylindric) [37]. The explicit parametrization of a paraboloid in the world frame is:

$$z = zl(x, y) = \frac{1}{2}\mathbf{u}^T \operatorname{diag}(\mathbf{k})\mathbf{u} \in \mathbb{R} \quad (4)$$

where $\mathbf{k} \triangleq [\kappa_x \ \kappa_y]^T$ are the intrinsic principle curvatures and $(x, y, z) \in \mathbb{R}^3$ is a point on the patch in the local frame. The extrinsic rotation $\mathbf{r}$ and translation $\mathbf{t}$ vectors, transform a patch to the world frame. The spherical and circular cylindric patch modeling are defined analogously (see [6]). The contact analysis between any of these patches that fit to the local areas in the environment and the foot, results in a 6DoF foot pose (the frame attached to its center). We represent this pose by its position (x_p, y_p, z_p), its orientation $(\mathbf{r}_p, \mathbf{t}_p)$ (rotation/translation vectors), and the N contact points with the environment surface pt_{c_i}, for $i \in [1, N]$. The patch fitting and contact analysis for sizes that fit to our robot feet sizes, take up to 1ms per patch.

III. Curved Surface Footstep Planner

In this section, we describe the algorithm in which the 3D flat planner (Sec. II-A) utilizes the curved patch contact analysis (Sec II-B) to configure and plan footholds on rough curved surfaces, when the ground–foot contact support is not guaranteed. First, we present the data input and pre-processing, and then the patch-based planner extension.

A. Visual Input Data and Pre-processing

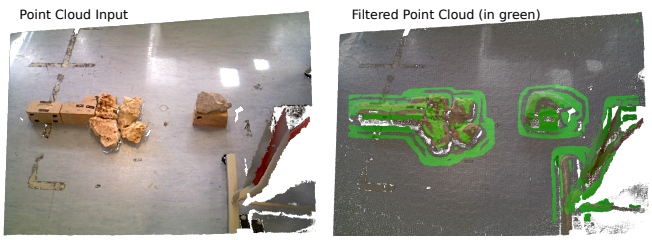

Fig. 4. Left: an input point cloud, including rocky curved surfaces on flat terrain. Right: the filtered point cloud (in green) after the pre-processing.

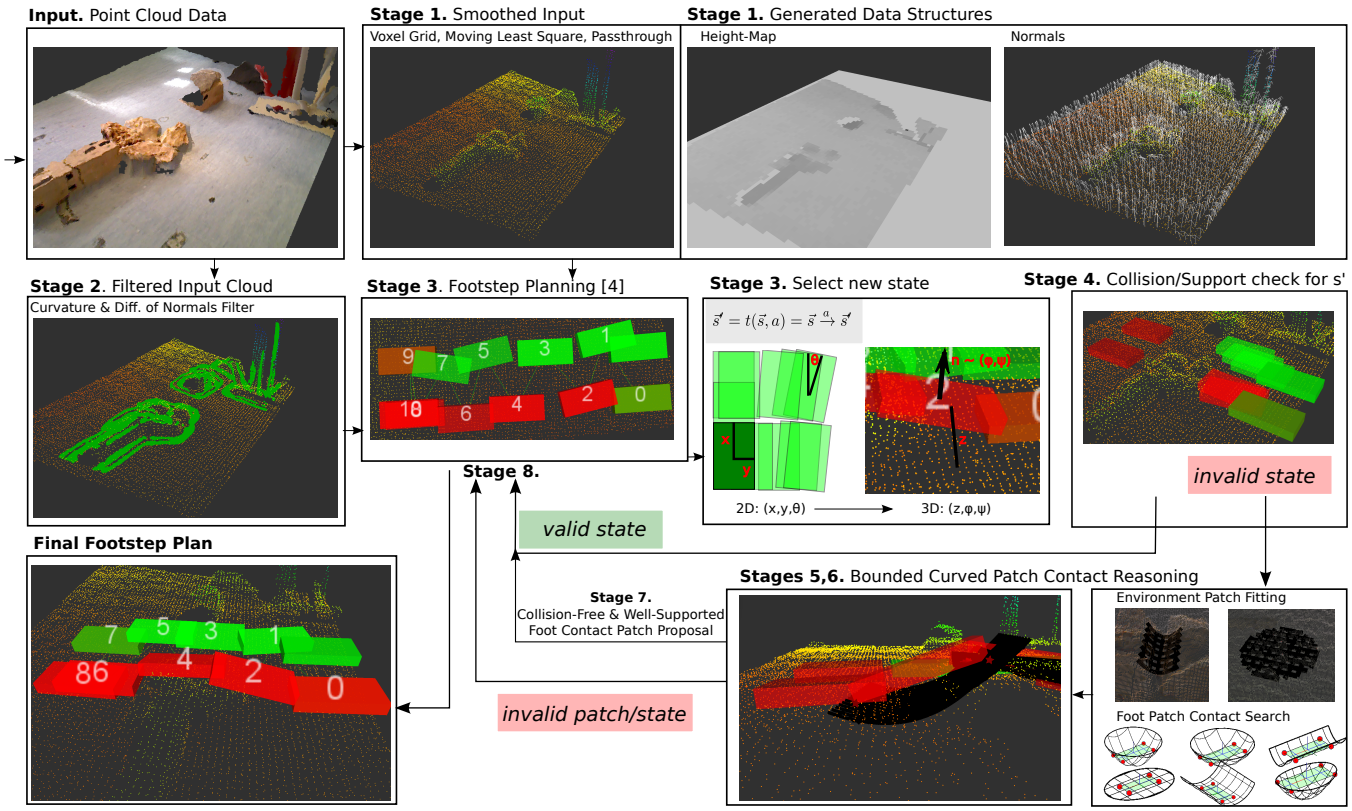

Fig. 3. The outline algorithm of the proposed rough curved terrain 3D footstep planner.

The introduced footstep planning method requires as input, point cloud data produced by any range sensor, such as RGB-D, stereo-vision, or lidar sensing systems. In contrary to the original planner [4], we require the data to be organized (i.e. grid organized, with one-to-one correspondence with the image) and from a projective device. This assumption is true for all the range sensors we are using (e.g. ASUS Xtion, the stereo camera and 2D lidar of the Multisense-SL). Notice that the assumption holds even when point cloud data are accumulated in a voxel-based grid structure (e.g. OctoMap), since the raycasting to produce a point cloud takes place from a virtual projective camera in the space (we have theoretically and experimentally validated this in [26]). The requirement of organized point clouds play a significant role in the computational complexity of the algorithm, but also in keeping a lookup table of the structures (i.e. patches) that were fitted in a particular position in the space. In particular, given the focal lengths f_x, f_y and the principal point offsets c_x, c_y (coming from the intrinsic matrix of the projective device), one can map a pixel (u, v) to the corresponding 3D point (x, y, z) in the space (and the other way around) using:

$$u = (f_x x)/z + c_x, \quad v = (f_y y)/z + c_y. \tag{5}$$

For every point in the cloud, a normal vector $\mathbf{n}$ can be computed fast using integral images [38]. We use the covariance-based method, where eigen-decomposition on the point cloud neighborhood takes place. The normal $\mathbf{n}$ is the eigenvector that corresponds to the smallest eigenvalue λ_0 ($\lambda_0 \leq \lambda_1 \leq \lambda_2$). The method is particularly fast and works only on organized point clouds. For instance, a point cloud from the ASUS RGB-D sensor (307k points) is computed in 160ms. We have tweaked the original integral images algorithm, which works directly on pixel-based neighborhood sizes (i.e., each point uses a fixed number of surrounding pixels for the computation), to a radius-based one, where the number of pixels that are used for each point depends on its position. This is possible through backprojection from the world to the camera image frame, using Eq. 5. The methodology also extracts a curvature measure c, as the largest eigenvalue over the sum of all:

$$c = \lambda_2/(\lambda_0 + \lambda_1 + \lambda_2) \tag{6}$$

Notice that the original 3D footstep planner works efficiently for flat surfaces. Thus, the curvature measure and the normals are used to filter out areas that are flat. For these areas, patches can be excluded from fitting during the planning, as the original flat 3D pose fitting will already quickly provide suitable results.

Using the integral images process, for every point in the cloud, we compute two normal vectors $\mathbf{n}_f$ and $\mathbf{n}_h$, one for radius r (i.e., the length of the foot) and one for $r/2$. Using these two normals per point one can calculate the angle between them:

$$\phi_{nn} = cos^{-1}(\mathbf{n}_f^T \cdot \mathbf{n}_h) \tag{7}$$

This can be used as a measure of the irregularity of the surface at the particular point. Such an area may not be possible to be represented by a second degree polynomial (such as our fitted curved patches). Thus, these areas can also be filtered out and excluded from the patch fitting process.

The curvature and the angle between two-scaled normals for each point is crucial to guarantee a computationally efficient planner, since the fitting and contact analysis method per patch may take up to 1ms, which could be slow if it is applied to thousands of points. Notice also that given the ability for our patches to handle point uncertainties during fitting, no other filtering is required for the fitting process. An example of such areas appear in Fig. 4, where in green are points that were not filtered out after applying a 0.02 and 30deg threshold for c and ϕ_{nn}, correspondingly.

B. Footstep Planning Adaptation Algorithm

Given the input point cloud, published both in the camera (as organized) and in the world (left foot) frames, we give the adaptation of the 3D footstep planning algorithm that handles rough curved surfaces. A diagram of the algorithm is visualized in Fig. 3.

Input: An organized point cloud pc. The foot size (length, width) $f_s = (f_l, f_w)$.

Stage 1. (smoothed input and data structures) Filter the input cloud pc to a smoothed one pc_s, after applying a voxel grid, moving least square, and pass through filtering, as described in [4]. Generate also the height-map and extract the point normals $\mathbf{n}$'s. These are needed to have the original flat-surface footstep planner working. Proceed to Stage 2.

Stage 2. (filtered input cloud) Find the curvature c (Eq. 6) and the angle ϕ_{nn} (Eq. 7) between the fine and coarse normals (n_f, n_h) for each point in cloud pc, as described in Sec. III-A. We set the neighborhood radius slightly bigger than the foot's length: $r = f_l/2 + \epsilon$, for some small $\epsilon \sim 1 - 2cm$. Filter out, to a new point cloud pc_f, the points for which:

$$c < c_{thres} \qquad \phi_{nn} > \phi_{thres} \tag{8}$$

for the experimentally determined thresholds $c_{thres} = 0.02$ and $\phi_{thres} = 30$deg. Proceed to Stage 3.

Stage 3. (footstep planner core) Run the 3D footstep planner as described in Sec. II-A. Each new state (footstep) $\mathbf{s}'$, coming from Eq. 1, consists of an (x_s, y_s) and θ_s (yaw) sample in the 2D ground plane, while the z_s value comes from the height-map and the ϕ_s and ψ_s (roll and pitch) from the corresponding normal vector $\mathbf{n}_s$. Proceed to Stage 4.

Stage 4. (state's collision/support check) For the new state proposal $\mathbf{s}'$, check whether the ground contact support estimation fails, either due to collisions between the foot and the environment or due to the absence of sufficient contact points (see Sec. II-A.3). In such failing case go to Stage 5, otherwise continue the planner as is, back from Stage 3.

Stage 5. (contact patch generation) If the point $pt_s = (x_s, y_s, z_s)$, is not in the filtered point cloud pc_f, go back to Stage 3 and continue the planner as is. If a patch has been already fitted for pt_s, extract it from the lookup table (see Sec. III-C). Otherwise, fit an environment patch $\mathbf{p}_e$ to pt_s's f_s-sized point neighborhood and determine the corresponding foot contact patch $\mathbf{p}_c$ (Sec. II-B). Proceed to Stage 6.

Stage 6. (contact patch re-estimation) If the foot contact patch $\mathbf{p}_c$ is one of those that appear in Fig. 2-right, with four points of contact go to Stage 7, otherwise continue the planner as is in Stage 3. Re-orient the foot contact patch $\mathbf{p}_c$ around the z-axis, such that the difference θ_{diff} between its yaw value $\theta_{\mathbf{p}_c}$ and the state's θ_s one is minimized. See Sec. III-C for the details. Proceed to Stage 7.

Stage 7. (contact patch validation) Validate the foot contact patch $\mathbf{p}_c$, with respect to the following two metrics, such that contact patches that cannot support a footstep are filtered out:

1) Positive residual ρ: the root-mean-square Euclidean distance between the patch's N-neighborhood data points $\mathbf{p}_i = (x_i, y_i, z_i)$ and the corresponding nearest points $\mathbf{q}_i$ on the patch itself:

$$\rho = \sqrt{\frac{\sum_{i=1}^{N} \|\mathbf{p}_i - \mathbf{q}_i\|^2}{N}} \tag{9}$$

such that $z_i \geq 0$ in the local patch frame, i.e., the points are above the patch's surface. These points maybe cause early contact with the environment. The metric also validates the patch fitting process itself. If the positive residual exceeds an experimentally defined threshold, $\rho \geq \rho_{thres} = 2$cm, go to Stage 3.

2) Contact points neighborhood ν: the number of points close to the contact points between the foot contact and the environment patch. This metric represents whether the surface can support the foothold. If the number of points does not exceed some experimentally defined threshold, $\nu \leq \nu_{thres} = N/25$, for an $f_l/5$ neighborhood, go to Stage 3 (as above, N is the number of sample points of the patch's neighborhood). Otherwise, proceed to Stage 8.

Stage 8. (contact patch state estimation) In this stage, a foot contact patch $\mathbf{p}_c$ has been generated to potentially update the ϕ_s, ψ_s, and z_s values of the current planner state $\mathbf{s} = (x_s, y_s, z_s, \phi_s, \psi_s, \theta_s)$. Given the foot contact patch position (x_p, y_p, z_p) and orientation ϕ_s, ψ_s, θ_s, we set:

$$z_s = z_p, \quad \phi_s = \phi_p, \text{ and } \psi_s = \psi_p \tag{10}$$

if the distance between (x_s, y_s) and (x_p, y_p) is small, as well as the angle between the patch's local z-axis (its normal) and the normal vector $\mathbf{n}_s$ of state $\mathbf{s}$, that comes from the roll and pitch values. Proceed to Stage 4, unless a solutions was found and no other state search is required.

Next, we discuss the way that the contact patches are cached (Stage 5) and correctly re-oriented (Stage 6).

C. The Foot Contact Patch State

In our previous work [6], we have described a way to detect a foot contact patch $\mathbf{p}_c$ given a fitted environment one $\mathbf{p}_e$, for all the possible environment patch types and foot patch orientations. In this paper, we select a subset of these environment patches $\mathbf{p}_e$'s, that guarantee at least 4 points of contact with the surface and are shown in Fig. 2-right.

The original footstep planner [4] guarantees very efficient computational timings, due to the height-map and normals generation that can be quickly accessed for every visiting

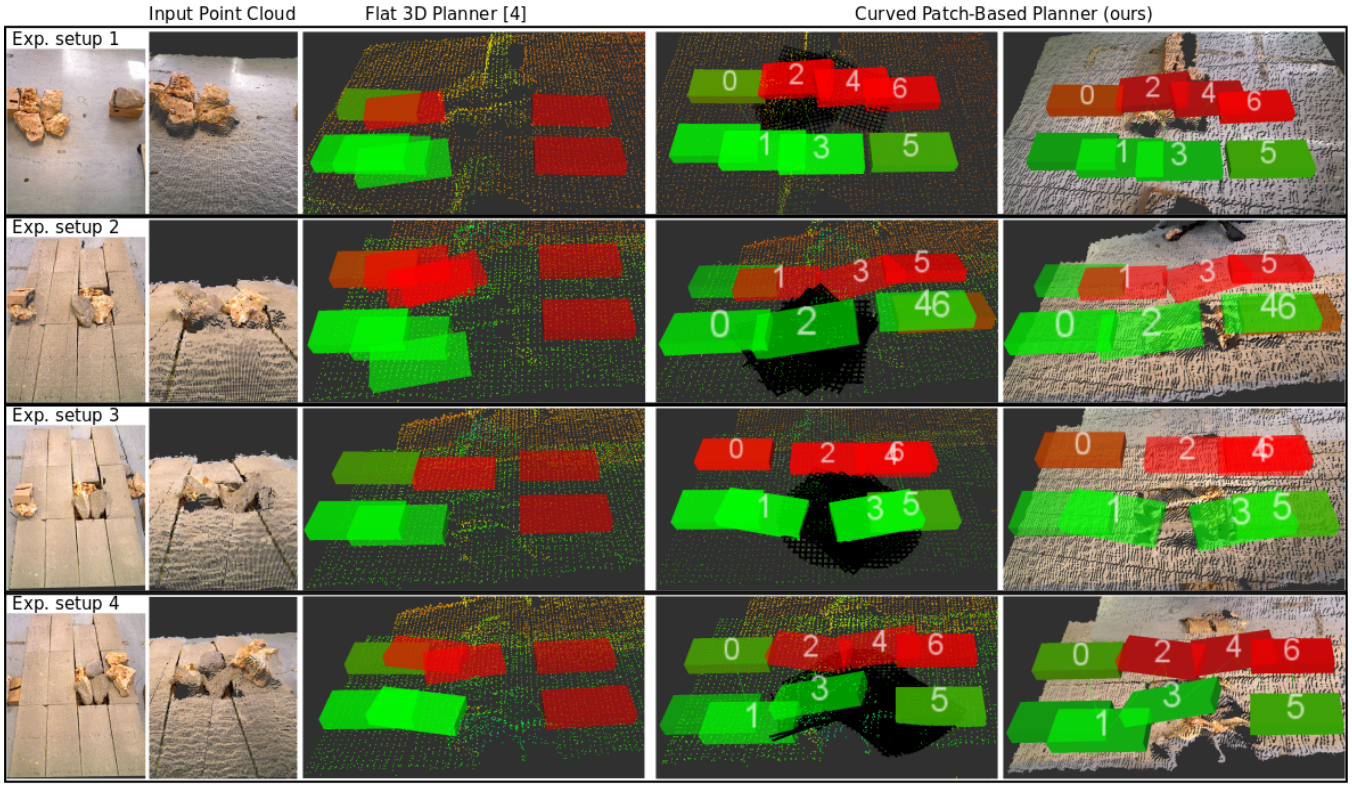

Fig. 5. Four setups: the input point cloud, the flat surface footstep planner [4] without suitable solutions, and our introduced footstep planning (in black the fitted patches). The non-numbered green/red poses are the start/goal poses. The numbered green/red poses are the right/left footsteps.

state, which is computed in roughly 0.01ms. In our case, the patch fitting and the foot patch contact analysis take around 1ms, so a caching method is also required in Stage 5 of the algorithm so that we do not re-fit patches at the same 3D point neighborhoods. Given that the cloud is organized, we hold a $u_s \times v_S$ patch lookup matrix (u_s/v_S is the width/height of the input in pixels). For each 3D sample seed point we save the corresponding fitted patch, using Eq. 5 for indexing. Through this lookup matrix, fitted patches can be quickly accessed. Notice that our system is thread protected and thus can be executed in multiple cores to speed up more the process.

In Stage 6 of the algorithm, when a state $\mathbf{s}$ is visited during the planning, its yaw value θ_s is part of the search sample tree. Thus, it is required to orient the contact patch $\mathbf{p}_c$ towards that direction. The circular/planar paraboloid and the spherical patch have revolute symmetry around their local z-axis. Thus, $\mathbf{p}_c$ can align $\mathbf{s}$'s yaw orientation, while keeping 4 points of contact with $\mathbf{p}_e$. In contrary, the elliptic/cylindric paraboloids and the circular cylindric patch have 4 points of contact only when $\mathbf{p}_c$'s local x-axis aligns with $\mathbf{p}_e$'s x and y local axes. If $\theta_{\mathbf{p}_c}$ is the yaw value of $\mathbf{p}_c$, then we pick the one that minimizes:

$$\theta_{\text{diff}} = \theta_{\mathbf{p}_c} - \theta_s. \tag{11}$$

In the next section, we present the experimental validation of the proposed footstep planner.

	Exp. 1	**Exp. 2**	**Exp. 3**	**Exp. 4**
steps (curved):	6 (2)	6 (2)	6 (1)	6 (1)
exp. states:	18	64	71	36
retr. steps:	270	809	759	210
fit. patches:	32	44	34	28
drop. patches:	28	19	27	24
path cost:	1.32	1.55	1.34	1.43
total planning t:	16.3s	21s	15.3s	17.4s

TABLE I
FOOTSTEP PLANNING STATISTICS

IV. EXPERIMENTS

In order to test our system, we perform two types of experiments. First, we acquire four different point clouds using a range sensor and we show footstep plans through curved rough surfaces. In the second one, we create a virtual environment that includes a rough surface and using our developed planner we let the WALK-MAN humanoid robot [39] locomote, in simulation. One of its steps is on the curved surface, where partial foothold contact is required.

A. Rough Curved Terrain Planning

For the perceptual experiment, we collected point cloud data from four different scenes, as shown in Fig. 5. Each one includes some rocks placed on a flat surface, that block the right part of the path. We first show that the original footstep planning [4] is not able to produce a suitable solution based on flat footholds, due to the rough curved surfaces in the path. Then, we run our proposed planner for the same path, ten times each. As shown in Fig. 5, our planner is able to always find a footstep sequence using one or two footsteps on the rocky terrain. We also collected statistics on the experiments

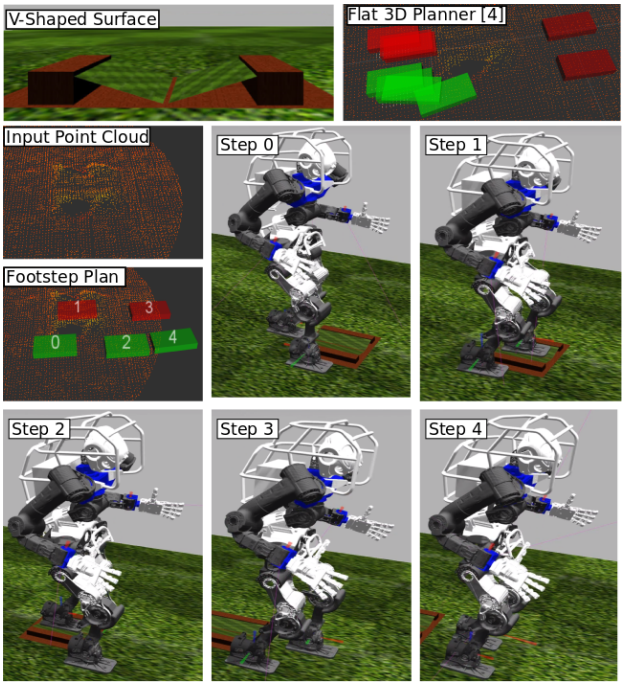

Fig. 6. Four steps (one on a curved surface) planning and locomotion experiments using the simulated WALK-MAN robot model in Gazebo.

and recorded the average values in Table I. The statistics are: the total number of produced footsteps and the count of footsteps produced on curved surfaces (noted as: *steps (curved)*), the total number of expanded states during the planning (noted as: *exp. states*), the number of the retrieved steps coming from the lookup tables (noted as: *retr. steps*), the total number of fitted patches (noted as: *fit. patches*), the number of dropped patches due to the evaluation (noted as: *drop. patches*), the path cost (noted as: *path cost*), and the total planning time (noted as: *total planning t*). The experiments ran on an Intel Core i7-4790K CPU at 4.00GHz, using 8 CPU cores and 16GB RAM.

In order to compare our results with the original planner's, we run the footstep planner on flat surfaces (using the same point clouds that we acquired), to produce 6 steps. In this case, $\sim 2,000$ states were expanded in average, for a total planning time of $\sim$ 10sec. From the statistical results in Tb. I, we can first see that our algorithm expands a small number of states, since it finds optimal paths through the rough terrain. Notice also that any planning of the original planner around an obstacle, may cause a significant increase in the number of states expansions. Secondly, we notice that the number of patches that are required are not that many, but most of them are declined due to bad fitting and, thus, representation of the underlying surface. Moreover, the overall planning is sufficiently fast, compared also to the original planner. The timing increases with the number of expanded states, the retrieved steps, and the fitted patches. Our code is not optimized and even though the patch fitting process does not take much time (1ms per patch), the implementation of the lookup tables/matrices and the conversions between the world and the camera frame are not computational optimal and should be further improved.

B. Locomotion Experiments in Simulation

In order to prove that the footstep plans that are generated from our algorithm are feasible for locomotion, we created a virtual environment in Gazebo, with a flat terrain that has a curved v-shaped surface in the path (Fig. 6). The two planes that compose the v-shaped surface, have an inclination of 17° with respect to the ground. Then, we placed the simulated WALK-MAN humanoid robot on the terrain, such that the curved surface is in front of its left foot. Using the virtual Multisense-SL stereo camera, which is at the robot's head, we acquired the terrain point cloud data. We gave as goal pose, for the footstep planning and locomotion, the area after the end of the curved surface.

We first run the original flat footstep planner [3] and show that it is unable to find a feasible solution in less than 60sec. This is because the planner needs to expand a lot of states to generate footsteps around the obstacle, i.e. curved surface, which is time consuming. On the contrary, we show that our introduced planner is able to find a footstep sequence with a step on the curved surface in 17.7sec. The planned footsteps are then sent to the robot's gait pattern generator [7], to create the CoM and feet trajectories based on the generated plan. The gait generator is responsible to check and generate a feasible execution sequence of the footsteps [40]. From the controller part, some ground impacts (due to early foot landings) are reduced with a feed-forward joint angle compensations that eliminates the unexpected deflections. Fig. 6 illustrates this experiment, where WALK-MAN is following the planned footsteps in a successful locomotion task.

V. CONCLUSIONS

In this paper, we presented a new footstep planner to deal with rough curved surfaces in the environment. We extended a state-of-the-art flat footstep planner, by integrating a curved contact patch analysis, to deal with rough curved terrains. We proved experimentally that our footstep planner is able to generate efficient footsteps, where the original flat planner cannot, and that the planned footholds can be used for locomotion. For the latter, we used the WALK-MAN humanoid robot in simulation.

From the engineering point of view, the method can be further optimized to produce faster footstep plans (using for instance GPU-based approaches), while the patch fitting algorithm needs to handle better non-regulars areas. In this way, the patches may represent better the underlying environment surfaces, decreasing the number of the dropped ones and thereby improving the time efficiency of the method. We also plan to test our algorithm in more complicated scenarios, such as planning hiking paths in rocky trails, where multiple footsteps are on curved surfaces. In this way, more RGB-D data can be analyzed, especially in cases such as environment edges, where the depth information may cause planning issues. We plan at using the hiking dataset that was generated in [41], and reason also the threshold selection (for instance using machine learning approaches). Moreover, real robot experiments need to be performed, though a more reliable

controller that can stabilize the robot in a quasi-static mode needs to be implemented. The experiments will focus also on multiple steps over rough terrain, where the support foot may be on some rough surface during locomotion. Moreover, stability results will be studied accordingly. Last but not least, a SLAM or state-estimation method is required to be integrated in the system. Given that the robot may slip from the original footstep plan after some steps, it should be able to follow the steps by an on-line trajectory correction or footstep planning re-generation.

ACKNOWLEDGMENT

This work is supported by the WALK-MAN & CogIMon (ICT-2013-10 & ICT-23-2014, under grant agreements no 611832 & 644727) EU projects. The NVIDIA graphics card used for this research was donated by the NVIDIA Corporation.

REFERENCES

[1] G. Wiedebach *et al.*, "Walking on Partial Footholds Including Line Contacts with the Humanoid Robot ATLAS," in *IEEE-RAS 16th Int. Conf. on Humanoid Robots (Humanoids)*, 2016, pp. 1312–1319.

[2] R. Deits and R. Tedrake, "Footstep Planning on Uneven Terrain with Mixed-Integer Convex Optimization," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2014, pp. 279–286.

[3] A. Stumpf, S. Kohlbrecher, O. von Stryk, and D. C. Conner, "Open Source Integrated 3D Footstep Planning Framework for Humanoid Robots," in *IEEE Int. Conf. on Humanoid Rob. (Humanoids)*, 2016.

[4] 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 *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2014, pp. 287–294.

[5] A. Hornung, D. Maier, and M. Bennewitz, "Search-based Footstep Planning," in *ICRA Workshop on Progress and Open Problems in Motion Planning and Navigation for Humanoids*, 2013.

[6] D. Kanoulas, C. Zhou, A. Nguyen, G. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis, "Vision-Based Foothold Contact Reasoning using Curved Surface Patches," in *IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, 2017, pp. 121–128.

[7] C. Zhou, Z. Li, X. Wang, N. Tsagarakis, and D. Caldwell, "Stabilization of Bipedal Walking Based on Compliance Control," *Autonomous Robots*, vol. 40, pp. 1041–1057, 2016.

[8] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2011.

[9] D. Kanoulas and M. Vona, "The Surface Patch Library (SPL)," in *In the 2014 IEEE ICRA Workshop: MATLAB/Simulink for Robotics Education and Research*, 2014, dkanou.github.io/projects/spl/.

[10] J. Chestnutt, M. Lau, G. Cheung, J. Kuffner, J. Hodgins, and T. Kanade, "Footstep Planning for the Honda ASIMO Humanoid," in *IEEE Int. Conf. on Rob. and Automation (ICRA)*, 2005, pp. 629–634.

[11] P. Michel, J. Chestnutt, J. Kuffner, and T. Kanade, "Vision-guided Humanoid Footstep Planning for Dynamic Environments," in *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2005, pp. 13–18.

[12] J. Garimort, A. Hornung, and M. Bennewitz, "Humanoid Navigation with Dynamic Footstep Plans," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 3982–3987.

[13] A. Hornung, A. Dornbush, M. Likhachev, and M. Bennewitz, "Anytime Search-Based Footstep Planning with Suboptimality Bounds," in *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2012.

[14] P. Karkowski and M. Bennewitz, "Real-time Footstep Planning using a Geometric Approach," in *IEEE ICRA*, May 2016, pp. 1782–1787.

[15] K. Okada, T. Ogura, A. Haneda, and M. Inaba, "Autonomous 3D Walking System for a Humanoid Robot Based on Visual Step Recognition and 3D Foot Step Planner," in *IEEE ICRA*, 2005, pp. 623–628.

[16] J.-S. Gutmann, M. Fukuchi, and M. Fujita, "3D Perception and Environment Map Generation for Humanoid Robot Navigation," *The Int. J. of Robotics Research*, vol. 27, no. 10, pp. 1117–1134, 2008.

[17] J. Chestnutt, Y. Takaoka, K. Suga, K. Nishiwaki, J. Kuffner, and S. Kagami, "Biped Navigation in Rough Environments Using On-board Sensing," in *IEEE/RSJ IROS*, 2009, pp. 3543–3548.

[18] K. Nishiwaki, J. Chestnutt, and S. Kagami, "Autonomous Navigation of a Humanoid Robot over Unknown Rough Terrain using a Laser Range Sensor," *IJRR*, vol. 31, no. 11, pp. 1251–1262, 2012.

[19] D. Maier, C. Lutz, and M. Bennewitz, "Integrated Perception, Mapping, and Footstep Planning for Humanoid Navigation Among 3D Obstacles," in *IEEE/RSJ IROS*, 2013, pp. 2658–2664.

[20] R. Ueda, S. Nozawa, K. Okada, and M. Inaba, "Biped Humanoid Navigation System Supervised through Interruptible User-Interface with Asynchronous Vision and Foot Sensor Monitoring," in *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2014, pp. 273–278.

[21] S. Brossette, J. Vaillant, F. Keith, A. Escande, and A. Kheddar, "Point-Cloud Multi-Contact Planning for Humanoids: Preliminary Results," in *CISRAM*, vol. 1, 2013, p. 6.

[22] M. Fallon, P. Marion, R. Deits, T. Whelan, M. Antone, J. McDonald, and R. Tedrake, "Continuous Humanoid Locomotion over Uneven Terrain using Stereo Fusion," in *IEEE-RAS Humanoids*, 2015.

[23] P. Karkowski, S. Oßwald, and M. Bennewitz, "Real-time footstep planning in 3D environments," in *IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, 2016, pp. 69–74.

[24] O. E. Ramos, M. García, N. Mansard, O. Stasse, J.-B. Hayet, and P. Souères, "Towards Reactive Vision-Guided Walking on Rough Terrain: An Inverse-Dynamics Based Approach," in *Workshop on Visual Navigation for Humanoid Robots (IEEE ICRA)*, 2013.

[25] N. Kita, M. Morisawa, and F. Kanehiro, "Foot Landing State Estimation from Point Cloud at a Landing Place," in *13th IEEE-RAS Humanoids*, 2013, pp. 252–259.

[26] D. Kanoulas, "Curved Surface Patches for Rough Terrain Perception," Ph.D. dissertation, CCIS, Northeastern University, August 2014.

[27] D. Kanoulas and M. Vona, "Sparse Surface Modeling with Curved Patches," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 4209–4215.

[28] O. Khatib and S.-Y. Chung, "SupraPeds: Humanoid Contact-Supported Locomotion for 3D Unstructured Environments," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014, pp. 2319–2325.

[29] C. Plagemann, S. Mischke, S. Prentice, K. Kersting, N. Roy, and W. Burgard, "A Bayesian Regression Approach to Terrain Mapping and an Application to Legged Robot Locomotion," *Journal of Field Robotics*, vol. 26, no. 10, pp. 789–811, Oct. 2009.

[30] J. Z. Kolter, Y. Kim, and A. Y. Ng, "Stereo Vision and Terrain Modeling for Quadruped Robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2009, pp. 3894–3901.

[31] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Learning, Planning, and Control for Quadruped Locomotion over Challenging Terrain," *IJRR*, no. 2, pp. 236–258, 2010.

[32] C. Mastalli, I. Havoutis, A. W. Winkler, D. G. Caldwell, and C. Semini, "On-line and On-board Planning and Perception for Quadrupedal Locomotion," in *IEEE TePRA*, 2015.

[33] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krusi, R. Siegwart, and M. Hutter, "Navigation Planning for Legged Robots in Challenging Terrain," in *IEEE/RSJ IROS*, 2016, pp. 1184–1189.

[34] A. Chilian and H. Hirschmller, "Stereo Camera Based Navigation of Mobile Robots on Rough Terrain," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2009, pp. 4571–4576.

[35] T. Klamt and S. Behnke, "Anytime Hybrid Driving-Stepping Locomotion Planning," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2017, pp. 4444–4451.

[36] M. Likhachev, G. Gordon, and S. Thrun, "ARA*: Anytime A* with Provable Bounds on Sub-Optimality," in *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2004.

[37] M. Vona and D. Kanoulas, "Curved Surface Contact Patches with Quantified Uncertainty," in *IEEE/RSJ IROS*, 2011, pp. 1439–1446.

[38] S. Holzer, R. B. Rusu, M. Dixon, S. Gedikli, and N. Navab, "Adaptive Neighborhood Selection for Real-time Surface Normal Estimation from Organized Point Cloud Data using Integral Images," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2012, pp. 2684–2689.

[39] N. G. Tsagarakis *et al.*, "WALK-MAN: A High Performance Humanoid Platform for Realistic Environments," *JFR*, 2017.

[40] C. Zhou, X. Wang, Z. Li, and N. Tsagarakis, "Overview of Gait Synthesis for the Humanoid COMAN," *Journal of Bionic Engineering*, vol. 14, no. 1, pp. 15–25, 2017.

[41] D. Kanoulas and M. Vona, "Bio-Inspired Rough Terrain Contact Patch Perception," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014, pp. 1719–1724.