International Journal of Humanoid Robotics
Vol. 14, No. 1 (2017) 1650022 (18 pages)

DOI: 10.1142/S0219843616500225

Supervoxel Plane Segmentation and Multi-Contact Motion Generation for Humanoid Stair Climbing

Tianwei Zhang*
Department of Mechano-Informatics,
School of Information Science and Technology,
The University of Tokyo,
7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
zhang@ynl.t.u-tokyo.ac.jp

Stéphane Caron
LIRMM CNRS-UM2,
161 rue Ada, Montpellier, France
stephane.caron@normalesup.org

Yoshihiko Nakamura†
Department of Mechano-Informatics,
School of Information Science and Technology,
The University of Tokyo,
7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
nakamura@ynl.t.u-tokyo.ac.jp

Received 4 November 2015
Accepted 11 June 2016
Published 29 August 2016

Stair climbing is still a challenging task for humanoid robots, especially in unknown environments. In this paper, we address this problem from perception to execution. Our first contribution is a real-time plane-segment estimation method using Lidar data *without* prior models of the staircase. We then integrate this solution with humanoid motion planning. Our second contribution is a stair-climbing motion generator where estimated plane segments are used to compute footholds and stability polygons. We evaluate our method on various staircases. We also demonstrate the feasibility of the generated trajectories in a real-life experiment with the humanoid robot HRP-4.

Keywords: Plane segmentation; humanoid stair climbing; static stability.

1. Introduction

The advantage of a humanoid robot originates from its human-like properties, namely, of shape, size, and mass. They ease non-verbal and intuitive communication

*,†Corresponding authors.

with the humans in the daily human-life scenes. The advantage is not only for communication, but also for automation in factory works. They allow the humanoid robots to accept the ordinary human settings and environments. The extreme cases would be the automated or teleoperated works in the human environments in life hazards after accidents and disasters. An advanced humanoid robot in the future will join rescue operations or go into the hazardous environments for the circumstances of accidents. The humanoid will do its best to use the environments made for the humans, such as the stairs, doors, tables, even ladders even if they are half-broken in collapsed buildings. The DARPA Robotics Challenge was a symbolical event held in June 2015 highlighting such a future advantage of humanoid robots.

Previous works on humanoid stair climbing[1,2] estimate geometric features of staircases and match them with prior models to achieve localization and motion generation. As they rely on prior models, such frameworks cannot be used when the number of steps or the shape of the staircase are determined at execution time. To achieve autonomous stair climbing in unknown environments, the robot needs the ability to estimate the geometry of the staircase, which implies, in particular, plane segment estimation. Reference 3 proposed a real-time staircase recognition method for stair climbing without prior models. However, their perception method suffers from the noise of stereo sensors, and they only executed their motions on small-size humanoid and staircases. In this paper, we estimate staircase plane segments, i.e., rectangular areas located in space, from point clouds data (PCD) obtained from a Lidar sensor. The reconstructed plane segments are subsequently sent to the motion generator.

Previous works on humanoid motion planning focused on horizontal, flat ground settings.[4,5] However, the stability of bipedal climbing patterns is delicate. Traditional methods to ensure such stability on flat ground include the well-known zero moment point (ZMP),[6–9] which assumes infinite friction and a plane horizontal ground. However, there are still problems to apply traditional motion generation and stabilization methods to nonflat multi-contact scenarios like stair climbing. To tackle this issue, we use a general static-stability criterion[10] which takes into account both friction and noncoplanar surfaces and integrates it within a quasi-static motion generator.

The contributions of this paper are therefore:

- a real-time plane segment estimation method for Lidar data without prior model,
- a stair-climbing motion generator using the reconstructed plane segments and a general static-stability criterion.

As shown in Fig. 1, we integrate both of these solutions into an efficient humanoid locomotion framework that we validate in a stair-climbing experiment by the HRP-4 humanoid robot.

The rest of the paper is organized as follows. Section 2 introduces related works, while Sec. 3 describes the point clouds acquisition setup. Our plane-segment

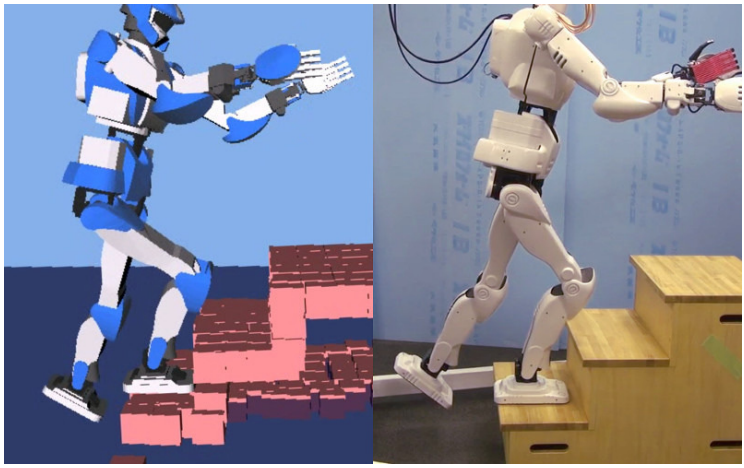


Fig. 1. Experimental setting of the paper. Motion planning in OpenRAVE with reconstructed surfaces (left) and real case execution on HRP-4 (right).

estimation method is described in Sec. 4 and evaluated in Sec. 5. Our climbing motion generator is described and evaluated in experiments in Sec. 6.

2. Related Works

2.1. *Plane segmentation*

Plane segment estimation and surface reconstruction with depth sensors[11,12] are most widely studied families of methods for unknown environment perception. Point normal is one of the essential features of 3D point clouds. Rusu introduced a normal estimation framework in Ref. 13, in which point normal is computed by analyzing the eigenvectors and eigenvalues of a covariance matrix created by its k-nearest neighbors. The computed normal vectors may have two opposite orientations. If the point clouds are captured from a single camera (viewpoint), the orientation towards to viewpoint is chosen. This framework has the limitation that the edge and cornerpoints' normals lose their sharp features. Moreover, in Ref. 14, these boundary regions are estimated based on the difference of estimated normals with different neighbor sizes. Another normal estimation method in Ref. 15 achieved real-time performance by building integral images of input PCD. However, it was designed for organized PCD, which assumes data structure is known. This condition limits its application on some sensors.

In the work of Gutmann *et al.*[16] a plane-segment estimation method is proposed to grow segments from straight scan lines. This plane estimation method is extremely fast since most input points are processed only in line-segmentation phase. However, plane normals which are important for biped robot locomotion generation are not

estimated. Another work of Rusu *et al.*[17] used random sample consensus (RANSAC) to generate polygons upon point clouds which are represented by small volumes, called cells or voxels. RANSAC is fast but tends to combine small local segments into big slopes, especially in large clutter scenes. In recent works, Papon *et al.*[18] extends an over-segmentation approach to real-time stereo data processing. This method cluster voxels with same features, such as voxels' position, color and normal features, to "supervoxels". Supervoxels decrease PCD size complexity, real-time plane segmentation is achieved in Ref. 18 by using supervoxels as input. However, this method is designed for dense RGB-D data. In large scenes the algorithm suffers from slow space dividing.

2.2. *Humanoid stair climbing*

Osswald contributed a series of works in autonomous stair-climbing system.[1,2,19] In these works, they estimated plane segments of a spiral stair with Lidar sensor and estimated their edges and corners with the RGB camera. Robot posture identification is done with multi-sensor fusion of inertial measurement unit (IMU), joint sensor, force sensor, RGB camera and laser scanner. This system succeeds to fulfill an autonomous climbing of hard spiral stair with a Nao robot. Multi-sensor fusion provides more information for environment learning and contributes precise robot posture estimation, but involves more computations and the sensibility to noise. Furthermore, this line of work assumes that the full environment model is known to the robot, thus limits its application in unknown and changing environments. Gutmann's stair climbing[3] combined stair recognition and motion planning, in their work. Stereo sensors are used to estimate the back and front edge of each staircase and final stair model is merged from multi-frame. Due to the noise of stereo sensors, their stair segments are not precious, finally, a small-size humanoid robot climbed a short staircase model (3 cm high each step) by using the merged stair model. However, there is no general stabilizer in his motion generator. In this paper, staircase plane segments are estimated and directly used for planning. The general stabilizer is involved to make a real-size humanoid robot to climb a real stair.

Traditional methods to ensure the stability of the robot (i.e., the dynamic balance of gravity and inertia forces by contact forces) include the well-known ZMP condition.[6–8] On horizontal ground and assuming infinite friction, a motion is dynamically balanced if and only if the ZMP lies inside the convex hull of contact points, which is called the support area. Control laws focus on maintaining the measured ZMP inside this support area while compensating for motion and force disturbances.[4,5] However, the concept of support area is still unclear in non-horizontal settings like stair climbing. Some extensions to "virtual" ZMPs have been proposed[4,9] yet they were not applied to nonflat settings. In this paper, we deal with this limitation by applying the general center-of-mass (CoM) static-stability polygon,[10] which is valid in general

nonflat settings. We show how it successfully applies to nonflat stair-climbing scenarios when the ZMP is maintained close to the CoM.

3. Point Clouds Acquisition

In this paper, the environment information is obtained from PCD by a Hokuyo UTM-30LX-EW 2D laser scanner. Comparing with high-frequency RGB-D stereo sensors, e.g., Microsoft Kinect, laser scanner has the superiorities of a larger field of view, further detectable range and suitability to bad lighting conditions. The chosen Hokuyo Lidar has 270° field of view, comparing with 57.8° of Kinect. Therefore, the robot can see more features without turning the neck, which is important for real-time mapping and perception. Kinect works from about 0.6 m to 4 m, while, Hokuyo UTM30 works in 0.1 m to 60 m with ±1% error. The baseline limitation makes a big problem for legged robot stair climbing. If the robot cannot see the staircase or obstacles in front of its feet, the stepping has to be finished without visual guidance. In our case, Hokuyo scanner can help the robot see the stair close to its standing feet accurately. It is also a reason why we used Hokuyo Lidar that it could provide robust measurements even under the direct sunlight, while Kinect showed significant disturbances under the sunlight.

Hokuyo UTM-30LX-EW returns a raw of points with 0.25° resolution every 25 ms, to achieve dense 3D point clouds, the row data are assembled according to the

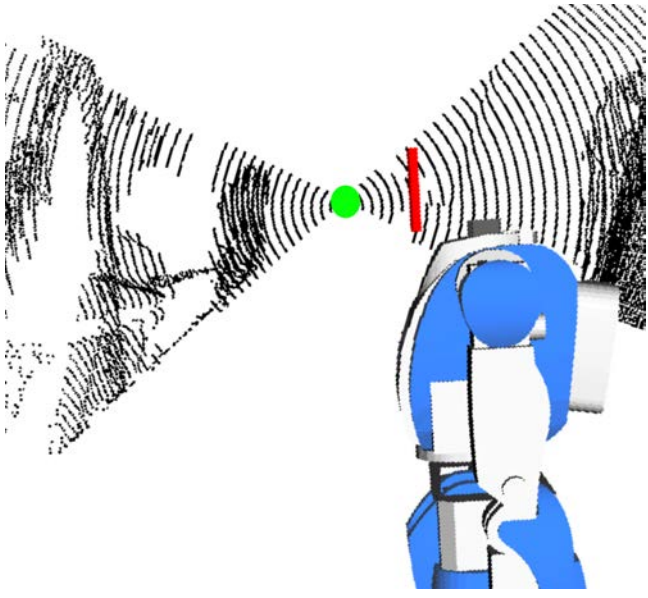


Fig. 2. A Hokuyo scanner actuated by a dynamixel MX-64 servo mounted on HRP-4's neck. When we combine row data to 3D PCD, robot neck joint and the servo's base (the read squire) are static, and the tilting angle of the center of scanner (the green point) respect to actuator base is accurately recorded by the actuator.

tilting angle of the scanner. This assembling processing limits the frame frequency. In Ref. 19, the assembled 3D PCD is noisy with an error up to 5 cm, and the author pointed out that the error mainly comes from shaking and the estimation of scanner's posture when robot tilts its head. To overcome this problem, our scanner is actuated by dynamixel MX-64 servo-actuator which is fixed on the robot neck's joint. As shown in Fig. 2, the robot body keeps static when the scanner is tilting to make a 60° scan, and the row tilting angle with respect to actuator's turning axis is recorded accurately by dynamixel MX-64. Note that, parameters such as tilt angle and pitching speed can be tuned to access dense or sparse PCD.

4. Unknown Environment Perception

In this paper, the task is set to climb stairs in unknown environments without moving obstacles, therefore, the goal of visual processing phase is to estimate suitable surfaces for robot climbing. Here, suitable means big enough for feet size and safe to step on. Although, we can estimate and reconstruct all the surfaces of the environment, to save time cost, we focus on the estimation of horizontal plane segments of the staircases, which are suitable for robot's feet.

Aiming at real-time perception, input PCD is filtered as much as possible, thus, we define three filters as follows:

Filter (1), we filter the points on steeps which are not tolerated to our robot, Algorithm 1 line [5–7].
Filter (2), we filter the sparse point cloud since they are not reliable, Algorithm 1 line [11–14].
Filter (3), we filter the isolated supervoxels since we do not consider about small obstacles in this setting.

The flowchart of our method is given in Fig. 3, from left to right, the assembled 3D PCD is input, and then Filters (1) and (2) are called to filter horizontal PCD out, after these filters, we cluster points on plane segments. Then, the boundaries of the staircases are identified as rectangles, which are used as input for motion planning. Finally, we execute the climbing motion on HRP-4 robot.

4.1. *Normal estimation and filtering*

The normal estimation and filtering algorithm is shown in Algorithm 1. In our case, the normals of PCD input are computed (function COMPUTE_NORMAL) by using the algorithm from.[13] In GET_CENTER, the surface normal is estimated by analyzing of the eigenvectors and eigenvalues of the covariance matrix created from the nearest neighbors of the query point. In this method, neighbor size is an important factor to the final performance. Large neighbor size leads to smooth normals changing in boundary regions and costs more computing time, while small neighbor size may lead to wrong normal estimation. The performances of different neighbor sizes will be evaluated in Sec. 5.

Algorithm 1. Normal Estimation and Filtering

Input: Point cloud data $\mathbb{P}$, voxel resolution v, neighborhood threshold publisher=SAGE Publications n_{t}, normal distance threshold T, distance function D_n, gravity vector $\mathbf{g}$
Output: Voxel center point cloud $\mathbb{C}$

1: $\mathbf{g} \leftarrow$ *acceleration_sensor*
2: **for** each point $p_i \in \mathbb{P}$ **do**
3: $\quad \mathbf{n}_i \leftarrow$ COMPUTE_NORMAL(p_i)
4: $\quad d_i \leftarrow D_n(\mathbf{n}_i, \mathbf{g}) = 1 - \mathbf{n}_i(p_i) \cdot \mathbf{g}$
5: $\quad$ **if** $d_i < T$ **then**
6: $\quad\quad$ delete p_i
7: $\quad$ **end if**
8: **end for**
9: $O_v \leftarrow$ BUILD_OCTREE$(v, \mathbb{P})$
10: Clear $\mathbb{C}$
11: **for** each voxel $v_i \in O_v$ **do**
12: $\quad$ **if** ADJACENCY SIZE OF $v_i > n_t$ **then**
13: $\quad\quad \mathbb{C} \leftarrow \mathbb{C} \cup \{GET_CENTER(v_i)\}$
14: $\quad$ **end if**
15: **end for**
16: **return** $\mathbb{C}$

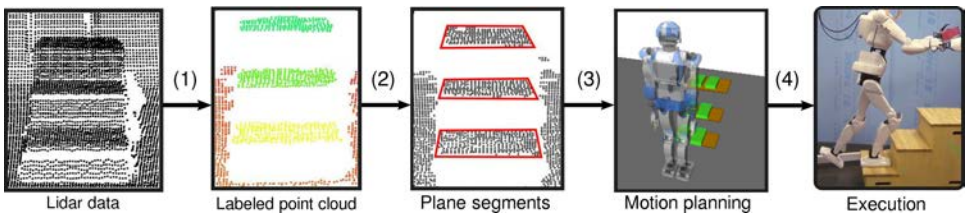


Fig. 3. Flowchart of our method: from left to right, we apply Filters (1)–(2) and supervoxel clustering in (a), in which we filtered non-horizontal plane segments and cluster adjacent points. After getting the labeled PCD, Filter (3) and staircase plane segment estimation are done within (b). The output of (b) are the boundaries of the staircases (red rectangles). Then, in (c), we apply the foothold planner, and in (d), the joint angle trajectories are generated from motion generator, finally, they are sent to robot controller.

Aiming at real-time surface estimation and reconstruction, the useless point clouds should be filtered out. In Filter (1), we compare point normals with gravity vector $\mathbf{g}$, which is obtained from acceleration sensor when the robot is statically standing on the ground. Then, we define normal distance D_n, and filter the points by threshold T which is related to the cosine of an acceptable slope angle, as shown in Algorithm 1 lines 8–11. Then comes Filter (2), in which the PCD is divided into small voxels with dimension v, and the voxels that contain not enough points are filtered

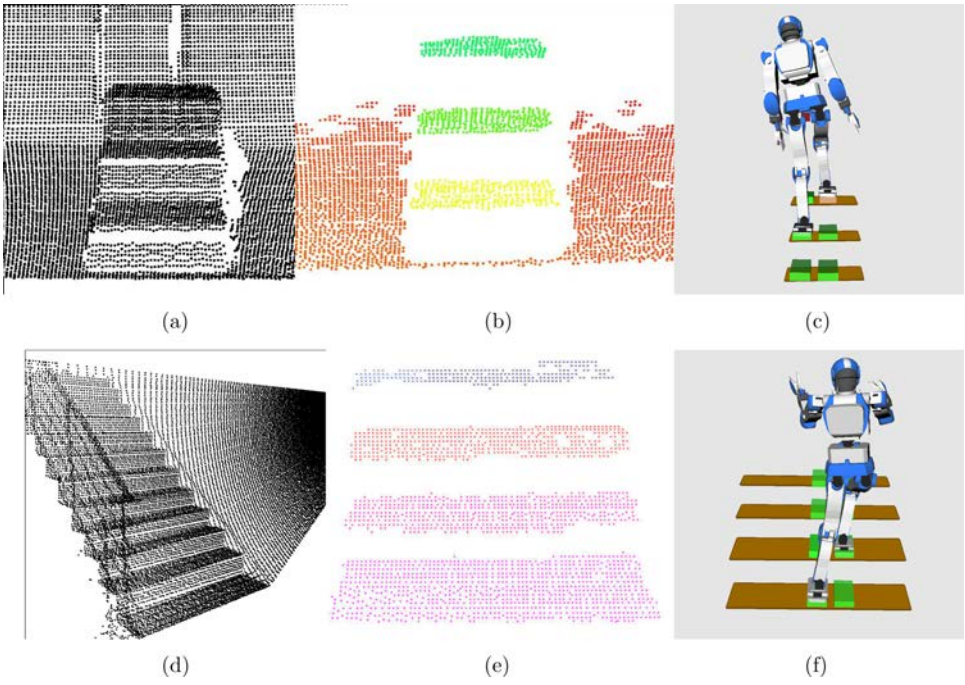


Fig. 4. Surface reconstruction process. (a) Original PCD of a three-level staircase. (b) Estimated plane segments in different colors. (c) Reconstructed surfaces with supervoxels of scene (a). (d) Original PCD of the real staircases. (e) Estimated horizontal planes of the real stair. (f) Reconstructed surfaces of (c).

out as sparse points. We take the point which is closest to the space volumes' mean coordinate as representative point. The sets of filtered point clouds $\mathbb{P}$ is shown in Fig. 4(b) and 4(d).

4.2. *Supervoxel clustering*

Similar to other methods, we compute a small number of representative points to reconstruct estimated surfaces. We use supervoxel centers as representative points. To cluster supervoxels, we need to measure the similarity between point clouds. Our distance function D measuring spatial information and local plane's normal feature is defined as:

$$D(v_i, v_j) = \sqrt{\mu \frac{D_s(v_i, v_j)^2}{3R_{\text{seed}}^2} + (1-\mu) D_n(v_i, v_j)^2}, \tag{1}$$

where v_i, v_j are voxels, and R_{seed} is seed resolution which determines searching scale and influence the size of the final supervoxels. $\mu \in (0, 1)$ is influence factors of the Euclidean distance D_s and normal distance D_n. D_n is defined as:

$$D_n(\mathbf{n}_i, \mathbf{n}_j) = 1 - \mathbf{n}_i \cdot \mathbf{n}_j. \tag{2}$$

The clustering in our method is a k-means similarity process. We use the voxel centers as inputs for supervoxels clustering. The octree O_s (Algorithm 1) is built with resolution R_{seed}, and the center points of its voxels are called "seeds". We then add voxels from $\mathbb{P}$ one-by-one, attaching them to the closest seed supervoxel center in O_s according to the distance measure D. This iteration keeps on going until the boundary of each searching volume is reached or there is no neighbor left.

Supervoxels expand synchronously, and the center of each supervoxel is updated as the mean of all the members after each expansion until there is no voxel left. Then, we cluster adjacent supervoxels. Clusters that contain too few voxels are removed since they represent local plane segments smaller than the robot's feet (Filter (3)). Finally, we identify the rectangular boundaries of each plane segment.

4.3. *Plane fitting boundary estimation*

We apply RANSAC to the whole supervoxel point cloud to fit a set of plane segments. However, some of them connect points belonging to different steps of the staircase. We filter them out as follows.

We chose to represent plane segments as vectors $\mathbf{V} = (\Delta x, \Delta y, \Delta z)$, where Δx, Δy and Δz are the scales of the segment on x-, y- and z-coordinates, respectively. Noting that the points belonging to a step plane segment should all have roughly the same z-coordinate ($\Delta z \approx 0$), we filter out the segments output by RANSAC for which Δz is bigger than a threshold ε, which we chose as $\varepsilon = 3$ cm. Next, as staircase plane segments should be represented by rectangles of similar widths and lengths, we cluster the remaining plane segments in the $(\Delta x, \Delta y)$ 2D-space using the Euclidean distance. We identify the biggest resulting cluster as the staircase, i.e., the set of plane segments corresponding to the staircase steps.

Take the scenes depicted in Fig. 4 as an example. The original input PCDs of a three-level staircase and a real stair are shown in Fig. 4(a) and 4(d). Meanwhile, Fig. 4(b) and 4(e) indicate the output of filtering supervoxel clustering. Only horizontal planes which are bigger than robot's feet are kept. Small obstacle planes and steep slopes are removed. Final reconstructed staircase surfaces are shown in Fig. 4(c) and 4(f). Note that only the first-level stairs are well represented since they are near to get dense point clouds. While the higher ones which are partly estimated will be fully reconstructed after the robot stepping on the lower ones.

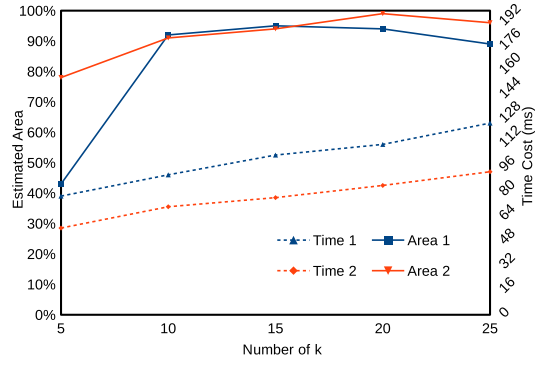
5. Plane Segmentation Evaluation

Detection of horizontal or near-horizontal plane segments for foot placement is crucial for humanoid robots. Therefore, we take estimated areas of the nearest staircase and the time cost as algorithm performance. We test these performances with different parameter values in the scenarios described in Fig. 4. Although, there are many parameters in clustering phase, the neighbors size for normal estimation has the biggest effect on whole processing time. To test this property, we take PCD of two scenarios and process them with different k (the number of k-nearest neighbors).

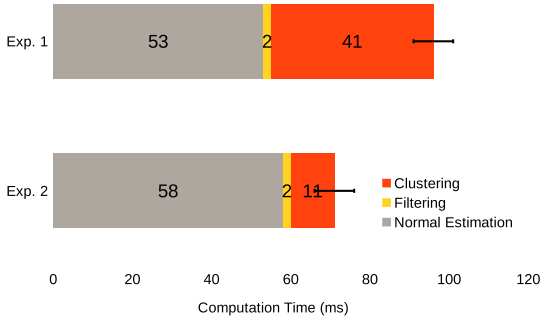
Table 1. Parameters used in experiments.

Seed resolution (m)	R_{seed}	0.1
Voxel resolution (m)	v	0.02
Normal distance threshold	T	$1+\cos(15^\circ)$
Distance impact factor	μ	0.2
Limit voxels number	n_t	5

The staircase used in Experiment 1 (Exp. 1) has three levels with 50 cm long, 20 cm high and 27 cm wide, which is hard enough for HRP-4 robot since its foot is a 14 cm width and 24 cm length square. Experiment 2 (Exp. 2) is taken in a real staircase scene, which has 11-levels staircases, each level is 123 cm long and 26.5 cm wide and 18 cm high. The fixed parameters are listed in Table 1. The computer used in these experiments has a 4 cores Intel(R) Core(TM) i7-4710MQ CPU@2.50 GHz, 15 GiB system memory, all the data is average value of 100 frames. The normal estimation phase is done with multiple thread computation.



(a)



(b)

Fig. 5. In chart (a), solid lines show the estimated area (percent value of real-stair size) and the dotted lines show the whole processing time with different neighbor size k. Red lines is about real staircases scenario, and blue lines is about the three-levels stair. (b) Shows the time cost (ms) distributions of the two experiments when $k = 10$.

See Fig. 5(a), dotted lines show the time costs of experiments when neighbor size k ranges from 5–25, while solid lines illustrate estimated area of the first step. The percent value of estimated area p is computed as follows:

$$p = \frac{a}{A} \sum_{i \in \text{segments}} n_i, \tag{3}$$

where n_i is the number of voxels included in the estimated plane segment i, a is voxel area ($4\,\text{cm}^2$), while A is the real area of the staircase. Note that, here we only talk about the estimated area of the first step which is nearest to the robot, the higher steps will be reconstructed after the robot climbing to the lower ones. Blue lines are about Exp. 1, and red lines are about Exp. 2. As expected, larger neighbor size costs more normal estimation time, and we observed experimentally that $k = 10$ was the best compromise between estimated area and computation time. Figure 5(b) shows the distribution of time cost when $k = 10$. As seen, the normal estimation time accounts more than supervoxel clustering. The yellow part is steep slope filtering phase. Clustering costs less in Exp. 2, since its space is smaller than Exp. 1, which is taken in a larger working room ($11 \times 7 \times 3$ m). According to filters setting, walls and ceilings are removed, but ground plane segments are kept, so that around 190 supervoxels clustered in Exp. 2 compared with 550 in Exp. 1.

Overall, considering about time cost and estimated area, neighbor size is chosen as 10 results plane segmentation phase costs around 100 ms per frame. The fact that our algorithm applies directly (with the same set of parameters) to different staircase shapes and sizes (Exps. 1 and 2) and for various robot postures suggests that it is robust for real-case applications.

6. Generation of Stair-Climbing Motions

6.1. *Motion generation*

In our setup, the humanoid robot HRP-4 starts with both feet parallel in front of the staircase. The whole stair-climbing motion is decomposed step-by-step, see Fig. 6, each step being decomposed in five segments:

(1) Transfer the CoM above the ankle of the left foot.
(2) Move the right foot to its target pose, making contact.
(3) Move the CoM above the contact polygon of the right foot.
(4) Lift the CoM while moving the left foot to its final pose.
(5) Bring the CoM to the middle of the support area.

For each segment, a linear trajectory is interpolated for the CoM and, if appropriate, for the pose of the free foot. The footholds are located at the middle of the reconstructed plane segment, i.e., their x (sagittal) coordinate is chosen in the middle of the segment, while their y (coronal) coordinate is constant for each foot (the robot is walking straight). From the CoM and free link trajectories, a whole-body trajectory

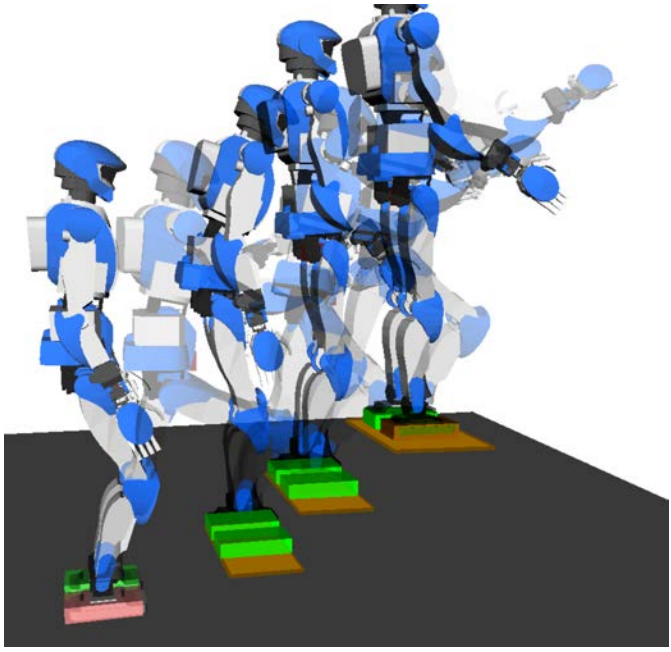


Fig. 6. Motion generated by the planner in simulations. Reconstructed plane segments (in yellow) are used to calculate footholds (green boxes), which are in turn used as end-effector positions for the left and right foot. The motion is finally generated by simultaneous tracking of the CoM and feet desired positions using inverse kinematics with a weighted cost function.

$t \mapsto \mathbf{q}(t)$ is computed using a QP-based IK tracker,[20] which minimizes a combined objective function under linear constraints. In our setting, we used the following:

Objective function:

(1) CoM deviation (weight $w_{\text{CoM}} = 1$).
(2) Free foot pose deviation (weight $w_{\text{free}} = 0.2$).
(3) Deviation of the upper-body degrees of freedom from a reference pose (weight $w_{\text{ref}} = 0.001$). This term was added to prevent the robot from accidentally touching the following staircase steps.

Constraints:

(1) Kinematic contact constraints $\mathbf{J}_i \dot{\boldsymbol{q}} = 0$, with $\mathbf{J}_i$ the Jacobian of the ith-contact.
(2) First-order velocity regulation

$$\dot{\boldsymbol{q}} \leq \min(\dot{\boldsymbol{q}}_{\max}, K(\boldsymbol{q}_{\max} - \boldsymbol{q})),$$

where $\dot{\boldsymbol{q}}_{\max}$ is a fixed maximum velocity, K is the DOF-limit velocity gain and $\boldsymbol{q}_{\max}$ specifies the upper joint limits of the robot (lower joint limits are enforced similarly).

We enforce that the ZMP stays close to the CoM through small values of $\dot{\boldsymbol{q}}_{\max}$ and of the regularization term in the objective function. The CoM trajectory is itself interpolated inside static-stability polygons.

6.2. *Static-stability polygons*

When walking on a horizontal floor, the static-stability condition for a biped is that its CoM lies inside the support polygon, that is the convex hull of ground contact points (in this case it coincides with the ZMP support area). When contact surfaces are not coplanar, this polygon still exists but its computation is more involved. In Ref. 10, the authors proposed a recursive polygon expansion method; we chose a different approach based on the double-description method, which we will now see in detail.

According to Ref. 21, and assuming the robot has enough actuation power (this assumption is checked later on in the OpenHRP dynamics simulator), the equations of motion for a humanoid robot can be written as:

$$\sum_{i=1}^{K} \boldsymbol{f}_i = m(\ddot{\boldsymbol{p}}_G - \boldsymbol{g}), \tag{4}$$

$$\sum_{i=1}^{K} \boldsymbol{p}_i \times \boldsymbol{f}_i + \boldsymbol{\tau}_i = m\boldsymbol{p}_G \times (\ddot{\boldsymbol{p}}_G - \boldsymbol{g}) + \dot{\mathbf{L}}_G, \tag{5}$$

where m is the total mass of the robot, $\boldsymbol{p}_i$ is the fixed position of the ith-contact frame (the rotation of which is assumed to be the identity), $\boldsymbol{p}_G$ is the position of the CoM, $\boldsymbol{w}_i = (\boldsymbol{f}_i, \boldsymbol{\tau}_i)$ is the resultant contact wrench applied by the environment at the ith-surface contact, and $\mathbf{L}_G$ is the angular momentum computed with respect to the CoM G. The right-hand side of the equations above is known as the *gravito-inertial wrench*. These equations are coupled with the kinodynamic contact constraints:

- *Kinematic*: The ith-contact frame is stationary with position $\boldsymbol{p}_i$ and orientation matrix $\mathbf{R}_i$ equal to the identity.
- *Dynamic*: Each contact wrench $\boldsymbol{w}_i$ belongs to a polyhedral convex cone known as the contact wrench cone (CWC),[22] whose formula depends on the geometry of the ith-contact area.

For rectangular contact surfaces (such as a humanoid foot) and approximating friction cones by four-sided friction pyramids, the CWC consists in 16 complementarity inequalities applying to the contact wrench $\boldsymbol{w}_i$. It is a minimal complete contact condition (see Ref. 22 for details). Thus, since all $(\boldsymbol{f}_i, \boldsymbol{\tau}_i)$ lie in polyhedral convex cones and Eqs. (4) and (5) are linear, these two equations can be equivalently rewritten in terms of the gravito-inertial wrench:

$$\mathbf{H}_{\text{stab}}(\boldsymbol{p}_1, \ldots, \boldsymbol{p}_K) \begin{bmatrix} \ddot{\boldsymbol{p}}_G - \boldsymbol{g} \\ \boldsymbol{p}_G \times (\ddot{\boldsymbol{p}}_G - \boldsymbol{g}) + \dot{\mathbf{L}}_G/m \end{bmatrix} \leq \mathbf{0},$$

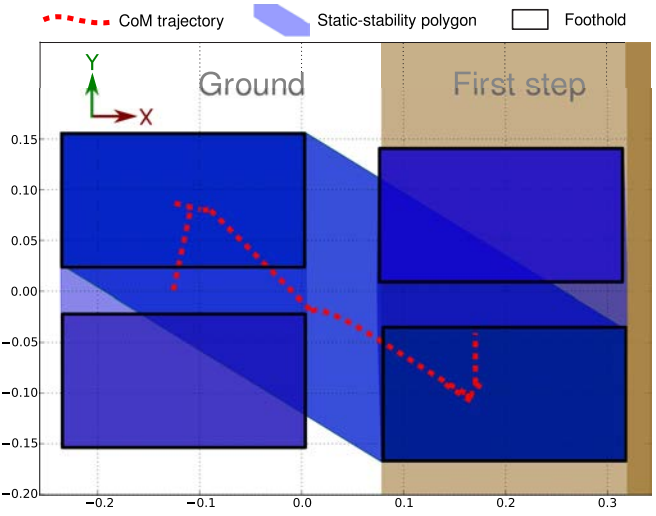


Fig. 7. View of the generated CoM trajectory in the transverse plane. Consecutive support areas are represented by blue polygons while the staircase is in brown. Dotted red lines depict the CoM trajectory. Note that the sagittal vector is pointing rightward, so that the motion goes from left to right. The unit of both axes is the meter.

where the matrix $\mathbf{H}_{\text{stab}}$ only depends on the contact positions $\boldsymbol{p}_i$. In the case at hand, we chose to generate so-called *statically stable* trajectories, i.e., enforcing low accelerations in order to neglect the terms $\ddot{\boldsymbol{p}}_G$ and $\dot{\mathbf{L}}_G$ in the formula above. Under this approximation, the inequality becomes $\mathbf{H}'_{\text{stab}}(\boldsymbol{p}_1, \ldots, \boldsymbol{p}_K)\boldsymbol{p}_G \leq \mathbf{0}$ after reduction of the constant terms. It can be shown from Eqs. (4) and (5) that this inequality always defines a right cylinder with a polygonal basis in the transverse plane,[10] known as the *static-stability* polygon. This polygon can be computed using e.g., a recursive polytope projection algorithm[10] or the double-description method.[23] We chose the latter. For more details, the source code used in this experiment is available at Ref. 20. A view of the transverse plane is shown in Fig. 7.

Because, we enforce slow velocities and keep the ZMP close to the CoM, we do not need to check the stability condition at each time instant: each segment has a stationary matrix $\mathbf{H}^{(i)}_{\text{stab}}$, which makes stability checking straightforward: when interpolating the linear CoM trajectory for the segment i, suffices to check whether its two extremities $\boldsymbol{p}^{(0)}_G$ and $\boldsymbol{p}^{(1)}_G$ lie in the support polygon, i.e., $\mathbf{H}^{(i)}_{\text{stab}}\boldsymbol{p}^{(j)}_G \leq 0$ for $j = 0, 1$.

6.3. *Execution on* HRP-4

Figure 8 shows the execution of the motion on the real robot. A complete video of the experiment can also be found at Ref. 20. Compared to the simulation environment, we needed to perform an additional fitting of the CoM coordinates at the single-support postures. The reason for this tuning is that, as the motion deals with non-coplanar surfaces, we could not use HRP-4's stabilizer (which assumes all contacts are made with a horizontal floor). A general solution to avoid this would be to

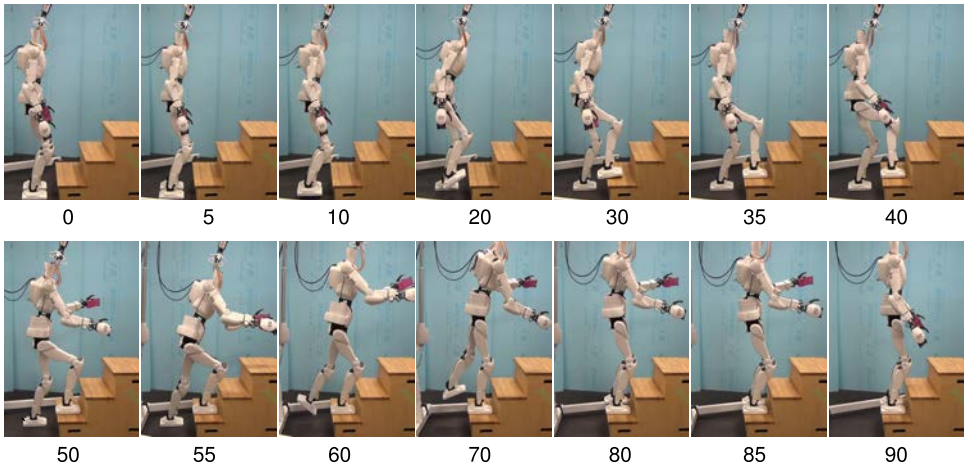


Fig. 8. Execution of the planned motion by HRP-4 for climbing one step. The height of the step is 24 cm. The motion is generated at low velocity, which is the range of validity of our multi-contact stability criterion. The total execution duration is 1 min 30 s. Time stamps (in seconds) are indicated below each picture. A complete video of the experiment can be found at Ref. 20.

develop a stabilizer that takes into account the static-stability polygon (rather than the traditional convex hull of ground contact points).

6.4. *Specificities of stair climbing*

We observed that, to climb a complete staircase in the absence of obstacles, the robot only needs to plan two sequences of footsteps: one to go from the initial configuration to the first staircase step, and the other to go from the first to the second staircase step. The latter can then be repeated for each pair of consecutive steps until the robot has fully climbed the staircase. In this regard, stair climbing is similar to walking on a horizontal floor: repetitions in the structure of the environment can be leveraged by generating a corresponding repeatable motion pattern, that is to say, a *gait*.

7. Conclusion

Multi-contact motion is the essential feature of the humanoid robots. Even walking in the daily human-life or factory environments needs to search and optimize footprint positions. It is not easy in the cluttered environments of outdoors or even indoors after accidents. Finding plane segments for the candidates of footprints is the crucial technology. The plane segmentation will also be necessary to find the position of hand prints for body supports in the more critical situation.

This paper proposed to use the supervoxel plane segmentation to reduce the computational cost taking account of the feature of Lidar point cloud data. The implemented results showed that the plane segmentation of the stair environments was done at 10 Hz by Intel Core-i7 PC.

The supervoxel plane segmentation was applied to stair-climbing motion generation of a humanoid robot, HRP-4. The motion was generated according to the previous work of the authors by using the gravito-inertial wrench and the support polygons taking the frictional inequality-constraints into consideration. In the implementation to HRP-4, the quasi-static assumption was made for conservativeness because of the weak actuator specification of the humanoid robot. The humanoid robots could successfully make stair climbing by the generated motion planned based on the computed plane segments.

Acknowledgment

This research was supported by 2014–2015 NEDO International R&D and Demonstrative Project in Environmental and Medical Fields/International R&D and Demonstrative Project in Robotics Fields/"Research and Development of Robot Open Platform for Disaster Response" (PI: Yoshihiko Nakamura).

References

1. S. Osswald, A. Hornung and M. Bennewitz, Improved proposals for highly accurate localization using range and vision data, in *Int. Conf. Intelligent Robots and Systems (IROS)*, 2012 IEEE/RSJ (IEEE, 2012), pp. 1809–1814.
2. S. Osswald, A. Gorog, A. Hornung and M. Bennewitz, Autonomous climbing of spiral staircases with humanoids, in *Int. Conf. Intelligent Robots and Systems (IROS)*, 2011 IEEE/RSJ (IEEE, 2011), pp. 4844–4849.
3. J.-S. Gutmann, M. Fukuchi and M. Fujita, Stair climbing for humanoid robots using stereo vision, in *Proc. 2004 IEEE/RSJ Int. Conf. Intelligent Robots and Systems, 2004 (IROS 2004)*, Vol. 2 (IEEE, 2004), pp. 1407–1413.
4. T. Sugihara, Y. Nakamura and H. Inoue, Real-time humanoid motion generation through ZMP manipulation based on inverted pendulum control, in *IEEE Int. Conf. Robotics and Automation, 2002.* Proc. ICRA'02, Vol. 2 (IEEE, 2002), pp. 1404–1409.
5. C. Santa Cruz and Y. Nakamura, Reactive stepping strategies for bipedal walking based on neutral point and boundary condition optimization, in *IEEE Int. Conf. Robotics and Automation (ICRA 2013)* (IEEE, 2013), pp. 3110–3115.
6. M. Vukobratović and B. Borovac, Zero-moment point thirty-five years of its life, *Int. J. Hum. Robot.* **1**(1) (2004) 157–173.
7. M. Vukobratović, B. Borovac and V. Potkonjak, ZMP: A review of some basic misunderstandings, *Int. J. Hum. Robot.* **3**(2) (2006) 153–175.
8. M. Vukobratović and J. Stepanenko, On the stability of anthropomorphic systems, *Math. Biosci.* **15**(1) (1972) 1–37.
9. P. Sardain and G. Bessonnet, Forces acting on a biped robot. Center of pressure-zero moment point, *IEEE Trans. Syst. Man Cybern. A, Syst. Humans* **34**(5) (2004) 630–637.
10. T. Bretl and S. Lall, Testing static equilibrium for legged robots, *IEEE Trans. Robot.* **24**(4) (2008) 794–807.
11. Z. C. Marton, R. B. Rusu and M. Beetz, On fast surface reconstruction methods for large and noisy point clouds, in *IEEE Int. Conf. Robotics and Automation, 2009.* ICRA'09 (IEEE, 2009), pp. 3218–3223.

12. F. Steinbrucker, C. Kerl and D. Cremers, Large-scale multi-resolution surface reconstruction from RGB-D sequences, in *IEEE Int. Conf. Computer Vision (ICCV 2013)* (IEEE, 2013), pp. 3264–3271.
13. R. B. Rusu, Semantic 3D object maps for everyday manipulation in human living environments, *KI-Künstliche Intelligenz* **24**(4) (2010) 345–348.
14. Y. Ioannou, B. Taati, R. Harrap and M. Greenspan, Difference of normals as a multi-scale operator in unorganized point clouds, in *Second Int. Conf. 3D Imaging, Modeling, Processing, Visualization and Transmission (3D IMPVT 2012)* (IEEE, 2012), pp. 501–508.
15. 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 *2012 IEEE/RSJ Int. Conf. Intelligent Robots and Systems* (IROS 2012, Vilamoura, Algarve, Portugal), 7–12 October 2012 (IEEE, 2012), pp. 2684–2689.
16. J.-S. Gutmann, M. Fukuchi and M. Fujita, 3D perception and environment map generation for humanoid robot navigation, *Int. J. Robot. Res.* **27**(10) (2008) 1117–1134.
17. R. Bogdan Rusu, A. Sundaresan, B. Morisset, K. Hauser, M. Agrawal, J.-C. Latombe and M. Beetz, Leaving flatland: Efficient real-time three-dimensional perception and motion planning, *J. Field Robot.* **26**(10) (2009) 841–862.
18. J. Papon, A. Abramov, M. Schoeler and F. Worgotter, Voxel cloud connectivity segmentation-supervoxels for point clouds, in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR 2013)* (IEEE, 2013), pp. 2027–2034.
19. S. Osswald, J.-S. Gutmann, A. Hornung and M. Bennewitz, From 3D point clouds to climbing stairs: A comparison of plane segmentation approaches for humanoids, in *Int. Conf. Humanoid Robots (Humanoids 2011)*, 11th IEEE-RAS (IEEE, 2011), pp. 93–98.
20. https://github.com/stephane-caron/ijhr-2016.
21. P.-B. Wieber, Holonomy and nonholonomy in the dynamics of articulated motion, *Fast Motions in Biomechanics and Robotics* (Springer, New York, 2006), pp. 411–425.
22. S. Caron, Q.-C. Pham and Y. Nakamura, Stability of surface contacts for humanoid robots: Closed-form formulae of the contact wrench for rectangular support areas, in *IEEE Int. Conf. Robotics and Automation (ICRA 2015)* (IEEE, 2015).
23. K. Fukuda and A. Prodon, Double description method revisited, in *Combinatorics and Computer Science* (Springer, New York, 1996), pp. 91–111.

Tianwei Zhang received his Master Degree of Electronic Science and Technology in 2013, Peking University, China, and is working toward the Doctor Degree in the Department of Mechano-Informatics, The University of Tokyo, Japan. His research interests include robot visual perception and motion planning. He has published several articles in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) and International Conference on Robotics and Biomimetics (ROBIO).

Stéphane Caron is a Post-Doctoral Researcher at the Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM). Born in Toulouse, France, in 1988, he received the B.Sc. and M.Sc. degrees from the École Normale Suprieure (45 rue d'Ulm), Paris, before moving to Japan for his doctoral studies. He received the Ph.D. from the University of Tokyo, Japan, in 2016. His dissertation was on whole-body motion planning for humanoid robots, with a focus on the problem of multi-contact stability.

Yoshihiko Nakamura is Professor at Department of Mechano-Informatics, University of Tokyo. He received Doctor of Engineering Degree from Kyoto University. For 1987–1991, he worked at University of California, Santa Barbara, as Assistant and Associate Professor. Humanoid robotics, cognitive robotics, neuro-musculoskeletal human modeling, biomedical systems, and their computational algorithms are his current fields of research. He is Fellow of JSME, Fellow of RSJ, Fellow of IEEE, and Fellow of WAAS. Dr. Nakamura serves as President of IFToMM (2012–2015). Dr. Nakamura is Foreign Member of Academy of Engineering Science of Serbia, and TUM Distinguished Affiliated Professor of Technische Universitét München.