Proceedings of the 2005 IEEE
International Conference on Robotics and Automation
Barcelona, Spain, April 2005

A Floor and Obstacle Height Map for 3D Navigation of a Humanoid Robot

Jens-Steffen Gutmann, Masaki Fukuchi, and Masahiro Fujita
Intelligent Systems Research Laboratory
Sony Corporation, Tokyo, Japan
Email: {steffen,fukuchi,mfujita}@pdp.crl.sony.co.jp

***Abstract*— With the development of biped robots, systems became able to navigate in a 3 dimensional world, walking up and down stairs, or climbing over small obstacles. We present a method for obtaining a labeled 2.5D grid map of the robot's surroundings. Each cell is marked either as floor or obstacle and contains a value telling the height of the floor or obstacle. Such height maps are useful for path planning and collision avoidance. The method uses a novel combination of a 3D occupancy grid for robust sensor data interpretation and a 2.5D height map for fine resolution floor values. We evaluate our approach using stereo vision on the humanoid robot QRIO and show the advantages over previous methods. Experimental results from navigation runs on an obstacle course demonstrate the ability of the method to generate detailed maps for autonomous navigation.**

***Index Terms*— 3D perception and navigation, obstacle avoidance, humanoid robot.**

I. Introduction

Mobile robot navigation in a 2 dimensional world using range data is a well studied problem with many publications on the subject [3]. With the development of precise 3D range sensors, increasing computational power, and the introduction of legged robots, 3 dimensional navigation became possible. Nowadays, humanoid robots can climb stairs [2], [5], [7], [16], step over barriers [2], and change its shape and posture for passing through narrow space [12], [19].

This paper addresses one of the aspects in 3D navigation, the creation of a 2.5D height map of the environment around the robot. The world is divided into a regular, 2 dimensional grid of evenly spaced cells. Each cell holds the height of an obstacle or the floor covering the area at the cell's location (thus the term 2.5D). Such maps are useful for finding paths through cluttered environments and for collision checking [1], [11]. A limitation of this approach is that multiple height levels for the same cell location cannot be represented. In such a situation we only store the highest level and assume the lower ones are not of interest for navigation. This assumption is reasonable for many man-made environments.

There are two problems one has to take care of when generating such maps from sensor data. First, 2.5D height maps do not allow to represent uncertainties and thus can suffer from noise in measurements. Integrating measurements by averaging can reduce such noise but bare outliers can still disturb the map in a significant way. The second problem is how to distinguish between obstacles (the robot should avoid) and planar surfaces (the robot could step on). Computing the curvature around a cell's location [1] can provide such information but requires fine-spaced grid maps for obtaining reasonable quality.

We tackle both problems using a novel representation where a coarse 3D occupancy grid provides the probabilistic backbone of the 2.5D height map. By inserting measurements into the 3D grid, an environment representation robust to sensor noise is obtained. A floor grid stores the height values of planar surfaces. Sensor measurements can be tested for outliers by verifying the corresponding cells in the 3D occupancy grid. In the same way, cells in the height map can be checked for validity. This enables a reliable and precise interpretation of the noisy sensor data.

For a humanoid robot, it is important to distinguish between the floor and obstacles for its safe navigation [11], [16], [17]. This problem can be addressed by employing plane segmentation [6], [16] on the range data. Measurements that are segmented into horizontal planes represent floor the robot can step on, and all others correspond to obstacles the robot should avoid. It is important that the floor height is precise since kinematic constraints of humanoid robots need accurate information to ensure stable walking. On the other hand, obstacle heights are allowed to be imprecise since our only aim is to avoid them. Therefore, we only store the height of the floor and compute the (coarse) height of obstacles by finding the highest index of occupied cells in the 3D grid at each location.

Our final map for navigation is a 2.5D floor and obstacle grid (FOG) generated from the 3D grid and the floor height map. Each cell holds a pair describing the type of cell (floor, obstacle or unknown) and its height, which can be computed in a straight forward way from the above two representations.

The rest of this paper is structured as follows. The next section contains prior work to ours and motivates our approach. Section III presents our novel representation using 3D occupancy grid and floor height map. The implementation of the algorithm on the humanoid robot QRIO is described in Section IV. Experimental results are reported in Section V and we conclude in Section VI.

II. Related Work

The probabilistic backbone of our approach are occupancy grid maps. For a comprehensive overview of such grid maps, see the article by Martin and Moravec [14]. Moravec recently reported new results with high-resolution, color-textured 3D evidence grids [15]. However, such fine-grained maps demand for large memory and processing time on sensor update, and require additional post-processing steps in order to detect flat surfaces a robot can step on. Coarse 3D grids are computationally efficient. If the vertical resolution is variable,

0-7803-8914-X/05/$20.00 ©2005 IEEE.

a *layered* occupancy grid is obtained [8]. This allows for the generation of useful maps for 2D navigation but lacks precise floor height information.

Raw data points or the position of visual features can be maintained in a 3 dimensional *stochastic map* [18]. However, computational requirements are high and further steps for extracting surface information are necessary which prohibit the navigation in 3D in real-time.

Probably the best source of information about where a humanoid can place its foot, is plane information segmented from range image data. A variety of plane extraction methods [6], [10], [16] and systems for obtaining detailed polygonal maps of indoor environments [9], [20] have been developed. The disadvantages of such approaches are that non-planar obstacles cannot be represented adequately, changes in the world are difficult to detect in the map, and outliers (wrong segmentation results) can disturb the integrity of the map.

If only monocular image data is available, simple box-like objects can be recognized by image edge extraction, hypotheses generation and verification. Although this approach seems brittle and might fall into traps easily, it enables the humanoid robot *Johnnie* to step over a barrier, walk around an obstacle, and climb up a staircase [2].

The method most relevant to ours is the approach of Kagami *et al.* [11]. They introduced the 2.5D terrain height map for humanoid robot navigation where each cell holds height information of the environment at the cell's 2D position. However, they do not distinguish between obstacles and planar surfaces and thus need further methods for deciding where the robot can place its foot [1]. Furthermore, they integrate information only by averaging data over time, a method that can suffer from outliers. Finally, changes in the environment might not be reflected in the map by their approach as we will show.

III. Floor and Obstacle Height Map

Our approach is similar to the one of Kagami *et al.* but we only store the height of horizontal planar surfaces in the 2.5D map, and additionally maintain a 3D occupancy grid updated with all measurements. This combination allows to verify the height values in the floor map (e.g. when the environment changes), enables the detection and filtering of outliers, distinguishes between obstacles and places the robot can step on, and provides precise floor height and coarse obstacle height information.

Formally, the 3D occupancy grid is a function of 3D positions to probabilities:

$$OG : (x, y, z) \mapsto p, \qquad p \in [0, 1] \tag{1}$$

and the floor height grid relates 2D positions to height values:

$$Floor : (x, y) \mapsto h, \qquad h \in R. \tag{2}$$

We reserve a special value $invalid = -\infty$ to indicate there is no floor height at a given position.

Both representation are organized as grids, that is, they store information in cells (or voxels). Throughout this paper we abstract from the detail that coordinates have to be translated to or from grid indices. We only require that both grids have the same x and y resolution. Using this convention we have a convenient way to e.g. look up the probability corresponding to a cell's floor height:

$$P_{Floor}(x, y) \quad = \quad OG(x, y, Floor(x, y)). \tag{3}$$

We use this value later for deciding whether a stored floor height still has enough support in the 3D occupancy grid.

For practical applications, the domain of both grid maps is limited to a range of values (e.g. we use a grid size of 100×100). By convention we define $OG(x, y, z) = 0.5$ and $Floor(x, y) = invalid$ for values x, y and z outside this domain. We also initialize the maps using these values:

$$\begin{array}{lcll} OG(x, y, z) & = & 0.5, & \forall x, y, z \\ Floor(x, y) & = & invalid, & \forall x, y. \end{array} \tag{4}$$

By maintaining and updating these two maps, we can compute a floor and obstacle grid (FOG) that tells for each location if there is a flat surface the robot can step on, or an obstacle we have to avoid:

$$FOG : (x, y) \mapsto (t, h) \tag{5}$$

where $t \in \{floor, obstacle, unknown\}$ and $h \in R$ is the height of the floor or obstacle. In the following sections, we describe how the 3D occupancy and floor height grids are updated on sensor input, and how the FOG is computed.

A. Approach Overview

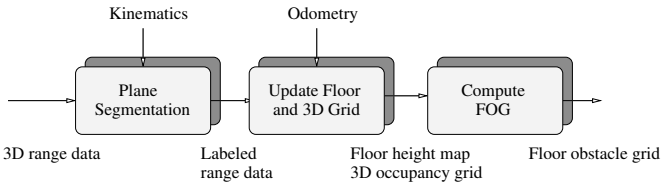

Fig. 1. System overview of floor and obstacle map generation.

Fig. 1 shows an outline of our approach. First, 3D data (x_i^C, y_i^C, z_i^C) captured by a range sensor C are segmented into planes. Kinematic information provides the method with pose information of the sensor relative to the robot coordinate system R. Segmentation returns labeled range data $(x_i^R, y_i^R, z_i^R, h_i^R)$ in robot coordinates where the additional value h_i^R is the height of the flat horizontal plane the 3D point (x_i^R, y_i^R, z_i^R) belongs to, or $invalid$ if the point was not segmented into such a plane. Basically h_i^R tells whether the robot could potentially step on this point ($h_i^R \neq invalid$), or if it refers to an obstacle ($h_i^R = invalid$). The labeled range data is then used to updated the 3D occupancy grid and floor height map from which in turn the FOG generated.

Updating OG and $Floor$ is performed in 5 steps (see Fig. 2). The first step is to transform the labeled range data into grid coordinates. This can be achieved by maintaining the robot pose $\mathbf{p}_R^G = (x_R^G, y_R^G, z_R^G, \theta_R^G)$ in the grid coordinate system G[1]. Updating the robot pose is usually referred to as *robot localization* which is a research topic by its own

[1]Note that we do not maintain the full 6D pose of the robot since roll and pitch are zero if we assume the robot always stands on a horizontal surface.

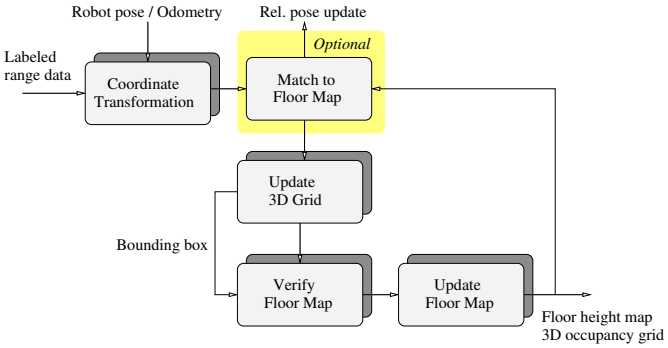

Fig. 2. Data flow for updating floor height map and 3D occupancy grid.

and behind the scope of this paper. In this work we assume the robot pose is given and is accurate enough for obtaining precise maps. Our approach to this problem will be presented in Section IV. Given the robot pose, the labeled range data is transformed into grid coordinates by first rotating about z of angle θ_R^G and then translating with (x_R^G, y_R^G, z_R^G). The result are labeled point data $\mathbf{p}_i^G = (x_i^G,\ y_i^G,\ z_i^G,\ h_i^G)$, where $h_i^G = h_i^R + z_R^G$. Note that $h_i^G = invalid$, if $h_i^R = invalid$.

The second step matches the labeled point data to the floor height map and is optional in our approach. If all range data, joint sensors, and the robot pose were within tight error bounds, we could skip this part. In practice, however, these quantities are affected with noise and by matching the new sensor data to the existing floor map, position errors can be reduced. For example, it is possible to compute a position update for robot localization using map matching [11]. Details of our solution are presented in Section IV.

The other 3 steps, updating the 3D grid, verifying the floor map, and updating the floor map, are presented below.

B. 3D Occupancy Grid

The 3D occupancy grid OG can be updated by ray tracing. Given the robot pose $\mathbf{p}_R^G$ and relative camera pose $\mathbf{p}_C^R$, we compute the camera pose $\mathbf{p}_C^G$ in grid coordinates. Then, for each labeled data point $\mathbf{p}_i^G$, we draw a straight line from $\mathbf{p}_C^G$ to $(x_i^G,\ y_i^G,\ z_i^G)$ and update all cells along this line according to a sensor model of the range data. Typically this updates the cells between sensor and data point as being empty and cells at the data point as occupied. We employ the Bayesian approach for updating the probability $P(c) = OG(x, y, z)$ of each grid cell c along the line according to:

$$\begin{aligned} P(c) \quad &\leftarrow \quad P(c \mid \mathbf{p}_i^G) \\ &= \quad \frac{P(\mathbf{p}_i^G \mid c)}{P(\mathbf{p}_i^G \mid c)P(c) + P(\mathbf{p}_i^G \mid \neg c)(1 - P(c))} \end{aligned} \tag{6}$$

where $P(\mathbf{p}_i^G \mid c)$ and $P(\mathbf{p}_i^G \mid \neg c)$ are the sensor model and define the likelihood of the measurement $\mathbf{p}_i^G$ given the cell is occupied or empty respectively.

The Bayesian method assumes the world is static. However, for practical applications one can limit the probability values to a minimum and maximum one which allows the occupancy grid to adapt itself to a (slowly) changing world.

When inserting all sensor data points into the grid, the method also computes the (x, y) bounding box of the changed area in the grid (called the affected area). This allows to limit the operations in the subsequent steps to a small area inside the potentially large grid.

C. 2.5D Floor Height Map

For obtaining reliable floor height information robust to sensor noise, we examine the cells in the 3D occupancy grid which correspond to the height information in floor grid and sensor data. We require that a cell in the 3D grid has accumulated enough probability before floor height information that corresponds to this cell can be treated as reliable. Formally, floor height h is considered reliable for position (x, y) if the following inequality holds:

$$OG(x, y, h) \quad > \quad P_{occ} \tag{7}$$

where $P_{occ} > 0.5$ is a sufficient large threshold.

Since cells in the affected area of the occupancy grid might have dropped below this threshold after updating the 3D grid, we need to verify the state of the existing floor map in the affected area. Each cell $Floor(x, y)$ is updated as:

$$Floor(x, y) \leftarrow \begin{cases} Floor(x, y), & \text{if } P_{Floor}(x, y) > P_{occ} \\ invalid, & \text{otherwise} \end{cases} \tag{8}$$

where P_{Floor} is defined in (3). This verification of the floor map is an advantage over previous 2.5D methods since it exploits the 3D structure of the range sensor data.

The last step for updating floor and 3D occupancy grid is to insert the floor data extracted by plane segmentation into the height map. For each labeled data point $(x_i^G,y_i^G,z_i^G,h_i^G)$, if $OG(x_i^G,y_i^G,h_i^G) > P_{occ}$, the floor grid is updated as:

$$Floor(x_i^G, y_i^G) \leftarrow \begin{cases} Floor(x_i^G,y_i^G), & \text{if } h_i^G < Floor(x_i^G,y_i^G) - \varepsilon \\ h_i^G, & \text{if } h_i^G > Floor(x_i^G,y_i^G) + \varepsilon \\ \nu Floor(x_i^G,y_i^G) + (1-\nu)h_i^G, & \text{otherwise} \end{cases} \tag{9}$$

where ε is a small gate threshold for testing if floor map and measurement refer to the same floor surface, and $\nu \in [0, 1]$ is a smoothing factor that filters readings over time. This method prevents the floor grid from step wise changes depending on where the robot looks in situations with multiple floor heights at the same position. It only keeps an estimate of the highest floor surface and ignores measurements of lower ones and of value $invalid$. Note that the smoothing is only possible since we know that existing floor height and measurement are both supported by high probabilities in the 3D occupancy grid.

D. Floor and Obstacle Grid (FOG)

The final step in our system is to compute the FOG from the floor and 3D occupancy grid. If we maintain this map over time, we can again reduce the computation time by only updating the cells in the affected area.

We define the obstacle height $Obstacle(x, y)$ for a position as the largest z coordinate of the corresponding 3D grid cells that contain enough probability for being occupied:

$$Obstacle(x, y) \quad = \quad \max_{OG(x,y,z) > P_{occ}} z \tag{10}$$

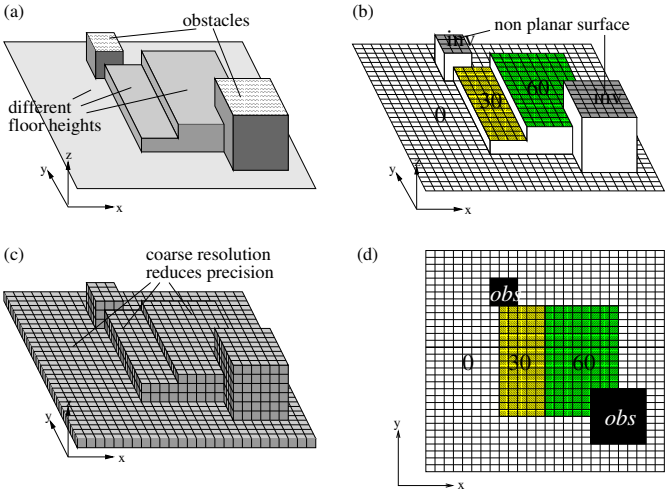

Fig. 3. Sample result in a simulated world. (a) World sensed by system. (b) Floor height map. (c) 3D occupancy grid. (d) Floor obstacle grid.

where max returns $invalid$ if no cells exceed P_{occ}. Note that the obstacle height is only a coarse estimate since it depends on the z resolution of the 3D grid.

Using this notation, the FOG can be computed as:

$$FOG(x,y) = \begin{cases} (obstacle, h_o), & \text{if } h_o > h_f + \delta \\ (floor, h_f), & \text{elif } h_f \neq invalid \\ (unknown, invalid), & \text{otherwise} \end{cases} \quad (11)$$

with $h_o = Obstacle(x,y)$, $h_f = Floor(x,y)$, and δ is a small threshold that accounts for noise in the grids by ignoring small obstacles close to an existing floor height.

The FOG representation ignores obstacles that are below the floor height of a given cell since these are not of interest for navigation. On the other hand, if there is a floor height with an obstacle above it (e.g. a curtain hanging above the floor) then the cell is marked as an obstacle. This is desirable since the robot cannot possible stand at this location.

E. Simulated example

Fig. 3 illustrates our approach with an example. A simulated world containing 3 different floor levels and 2 obstacles exhibiting a rough surface at their top is fully observed by a perfect sensor (Fig. 3(a)). The floor height map captures the precise floor height but cannot represent any obstacles since the data from the rough surfaces could not be segmented into planes. Therefore, areas corresponding to the obstacles obtain a value of $invalid$ (Fig. 3(b)). The 3D occupancy grid (Fig. 3(c)) contains a complete representation of the scene but the floor height information is only coarse. The top of the obstacles show the same flat surface as the floor levels. Our combination (Fig. 3(d)) is able to represent both types of information: precise height map of the floor and coarse height information about obstacles.

IV. Implementation

We implemented our approach on QRIO, a small humanoid robot with 38 degrees of freedom. Three MIPS R5000 CPUs clocked at 400MHz provide the computational power for the robot [4]. For visual perception, QRIO is equipped with stereo cameras with a field of view of 47° horizontally and 39° vertically, and a FPGA module for stereo processing. This sub system provides disparity images with 12.5 fps in resolutions of 176x144 and 88x72 pixels [17]. We utilize the lower 88x72 pixel resolution in our system which allows to process all frames in real-time on the robot. This includes the plane segmentation, the operations needed to maintain the grid maps, and further navigation modules which are behind the scope of this paper.

The stereo images are segmented into planes by a variation of the *scan line grouping* approach of Jiang and Bunke [10]. In our experiments this method gave the best results with respect to under segmentation which we attribute to our method's ability to adapt segmentation to noise locally in the data [6]. Fig. 4 shows the left image of a sample image pair and the achieved segmentation result by our method.

For the grid maps we use a size of 100x100 with cells of 4 cm side length. Additionally, the 3D grid uses 16 layers (each 4 cm) in the z direction. The maps are centered on the robot such that they reflect a 4x4 m area around the robot. When the robot moves, the grids are shifted according to odometry information obtained by the kinematic information from one foot to the other [17]. The robot's z_R^G and θ_R^G coordinates are stored as absolute entities and the grids are not shifted or rotated on changes in z_R^G respectively θ_R^G.

By keeping the grids centered on the robot pose, the impact of incremental (x,y)-position errors can be kept low. As the robot moves on, older parts of the maps are erased and new terrain is entered. A few problems reside though. When the robot constantly rotates, the error in orientation becomes significant. This should be avoided in the motion control. The robots elevation z_R^G also suffers from incremental errors. We filter z_R^G with the floor height at the robot position, $Floor(x_R^G, y_R^G)$. This significantly reduces the errors accumulated in z_R^G. However, we note that these techniques only yield a reduction of the pose drift and a more advanced approach is needed for keeping errors bounded.

A further source of noise is in the encoders of the robot's joints rendering estimates of the camera pose $\mathbf{p}_C^R$ unreliable. We observed errors of up to 7° in rotation and 10 mm in translation. Such high errors degrade the quality of the 3D grid and consequently the floor height map. Therefore, plane

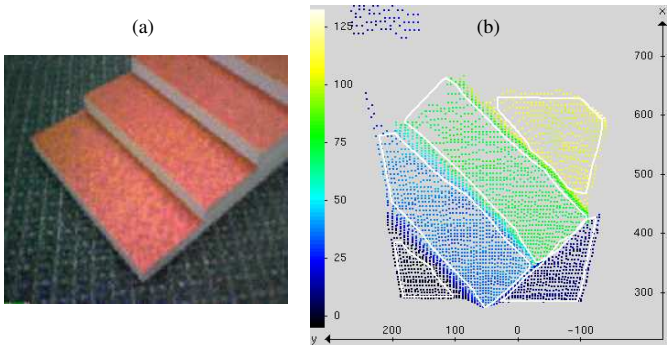

Fig. 4. Plane segmentation. (a) Camera image. (b) Segmented range data.

segmentation rotates the range data according to the direction of planes that are approximately horizontal. It also searches for the plane that is closest to an expected ground plane, and, if found, shifts all data points accordingly.

Still, the labeled range data can be unreliable, e.g. when a ground plane could not be found. A degeneration of the grids can be avoided by matching the sensor data to the floor height map (see Fig. 2). In our implementation, we search for a z shift that minimizes the sum of squared distances between measurements and floor map. This z shift, if found, is then added to the z_i^G and h_i^G coordinates. If there are only few matching pairs, the result of ground plane detection decides whether the data should be considered reliable or withdrawn completely.

A last implementation detail concerns the Bayesian cell update in our 3D grid. We employ a sharp sensor model where one constant P_{sense} defines the model as

$$P(\mathbf{p}_i^G \mid c) = \begin{cases} P_{sense}, & \text{if } (x_i^G, y_i^G, z_i^G) = c \\ 1 - P_{sense}, & \text{otherwise} \end{cases} \quad (12)$$

$$P(\mathbf{p}_i^G \mid \neg c) = 1 - P(\mathbf{p}_i^G \mid c). \quad (13)$$

By discretization of probabilities (we use 8 bit integers), the sensor model can be pre-computed such that updating an occupancy grid cell results into a simple table lookup.

V. Results

In a first experiment we compare our approach to one that simply averages over the z observations for each grid cell similar to the approach of Kagami *et al.* [11]. We placed a flat box in front of the robot and let the system observe it. Fig. 5(a) shows the experimental setup and Fig. 5(b) the result of simple averaging after several frames. The result of our method is very similar to the averaging one.

The box is then removed and the methods continue updating their maps. Simple averaging is inherently not able to erase all data corresponding to the box as the floor area closer to the robot cannot be observed by the robot without changing its field of view. Thus a portion of the obstacle remains in the map as can be seen in Fig. 5(c). Altering of these cells requires to explore the 3D structure of range sensing. Our approach, on the other hand, exploits this structure through the 3D occupancy grid. After several updates, cells corresponding to the box fall below P_{occ} and thus, the floor height vanishes from the map completely (see Fig. 5(d)).

Another effect can be observed when comparing our approach with simple averaging. If the box is slowly moved through the camera's field of view, the averaging method does not produce a sharp floor height map but one where height values between the height of the box and the floor are present. Our method does not show this behavior and the floor height jumps directly according to the motion of the box after the 3D occupancy grid has adapted to the changed situation.

Thus, our method produces more accurate and reliable results than simple averaging due to the robust nature of the employed 3D grid.

In order to evaluate our method in a larger environment we setup an obstacle course on a 4 x 1 meter large stage and placed several obstacles, a sill and a staircase leading

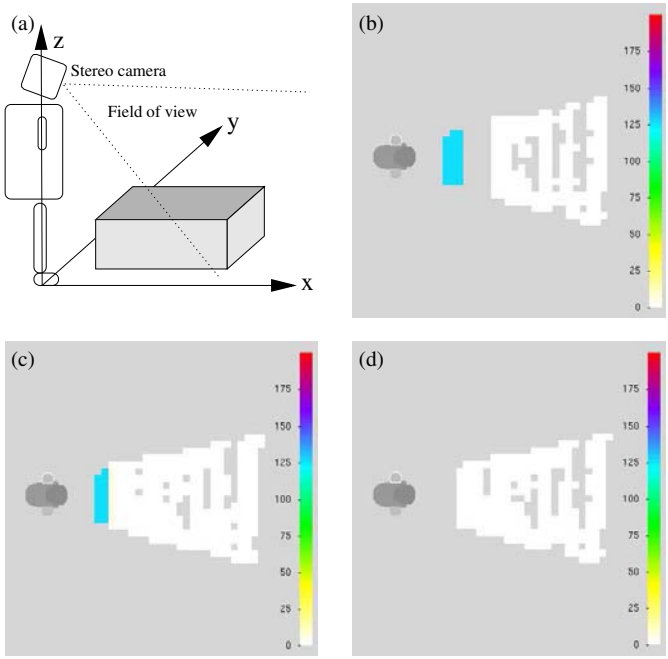

Fig. 5. Comparison of our approach with simple averaging. (a) A box is placed in front of the robot. (b) Initial map of both approaches. (c) After removing the box, simple averaging is unable to "forget" the box. (d) Our approach erases the height data corresponding to the box.

to a higher level platform (see Fig. 6). In the past we used this setup for demonstrating QRIO to the public, e.g. for the presentation of our 3D navigation system at RoboCup 2004 and IAV 2004 in Lisbon, and at IROS 2004 in Sendai.

We let QRIO walk between the obstacles and climb up and down the sill using our stair recognition and climbing system [5]. During its motion QRIO continuously captured stereo data which was processed by our map-building system. Fig. 7 shows the floor and obstacle grid result of our approach before QRIO was climbing up the stairs.

The resulting map presents a good approximation of the real world. Obstacles are found close to their true position and the sill and stair case are recognized correctly. Table I compares the estimated heights of the different objects in the map with ground truth measured by hand. We feel that these results are quite satisfactory and the quality of the maps very useful for the navigation of the robot.

We measured the maximum time on sensor update as 12 msec on the robot's on-board CPU (corresponds to 3 msec on a Pentium III, 1.4GHz). This leaves enough computing resources to other modules for navigation.

floor	estimated	ground truth
sill	37 mm	38 mm
1^{st} step	31 mm	31 mm
2^{nd} step	61 mm	61 mm
3^{rd} step	94 mm	92 mm
higher platform	126 mm	122 mm

TABLE I
ESTIMATED HEIGHT VS. GROUND TRUTH OF FLOOR LEVELS IN THE MAP.

Fig. 6. Experimental setup. Several obstacles and a sill are placed on a stage. A staircase leads to a platform on a higher level.

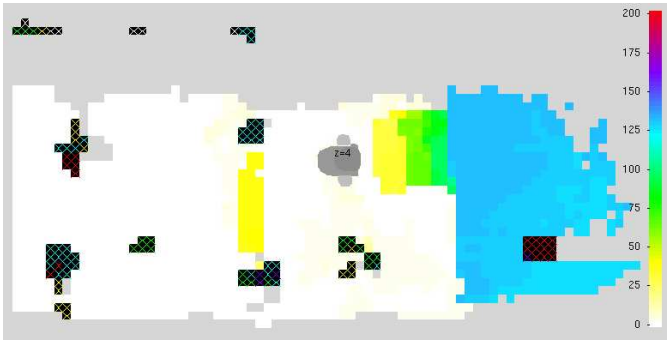

Fig. 7. Floor and obstacle map generated by QRIO. Gray areas indicate unknown terrain, the floor height is drawn in different colors, and obstacles are marked in black with a colored cross indicating their coarse height. The indicator on the right maps colors to height values in mm.

VI. Conclusion

We presented a novel method for building a floor and obstacle grid representation of a real world observed by a range sensor. Grid cells hold either the precise height of a horizontal surface or a coarse height of an obstacle covering the cell. The method is robust to sensor noise and changes in the environment thanks to a 3D occupancy grid which verifies and filters floor height information and sensor data. The method advances previous methods which only maintain a 2.5D terrain map and average data over time.

A limitation of our approach is that only horizontal planes are mapped into the floor grid. It should be possible to extend the method to allow for inclined surfaces by also estimating tilt and roll of the robot pose and floor cells. However, it is unclear how well such an estimation would work in practice since the additional two degrees of freedom are subject to incremental estimation errors similar to the orientation and elevation estimate of the robot.

We believe that maps generated by our approach can be employed in a variety of existing path planning systems that have been developed in the past for simulated 2.5D worlds [1], [13], [19]. Thus, our approach fills in the gap between path-planning in simulation and real robots by providing detailed floor and obstacle maps of real world environments.

A possible extension of our method would be to also compute a ceiling map of the environment by finding the smallest index of occupied cells in the 3D occupancy grid. This would further enable the application of behavior planning systems [12], [19] where the robot could e.g. crawl underneath obstacles. We follow this direction in the future.

References

[1] J. Chestnutt, J. Kuffner, K. Nishiwaki, and S. Kagami. Planning biped navigation strategies in complex environments. In *Int. Conf. on Humanoid Robotics (Humanoids)*, 2003.

[2] R. Cupec, O. Lorch, and G. Schmidt. Vision-guided humanoid walking - concepts and experiments. In *Int. Workshop on Robotics in Alpe-Adria-Danube Region (RAAD)*, Cassino, Italy, May 2003.

[3] L. Feng, J. Borenstein, and H.R. Everett. "Where am I?" Sensors and methods for autonomous mobile robot positioning, 1996.

[4] M. Fujita, Y. Kuroki, T. Ishida, and T.T. Doi. A small humanoid robot SDR-4X for entertainment applications. In *Int. Conf. on Advanced Intelligent Mechatronics (AIM)*, Kobe, Japan, 2003.

[5] J.-S. Gutmann, M. Fukuchi, and M. Fujita. Stair climbing for humanoid robots using stereo vision. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, Sendai, Japan, 2004.

[6] J.-S. Gutmann, K. Kawamoto, K. Sabe, and M. Fukuchi. Multiple plane segmentation with the application of stair climbing for humanoid robots. In *Symposium on Intelligent Autonomous Vehicles (IAV)*, 2004.

[7] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka. The development of Honda humanoid robot. In *Int. Conf. on Robotics and Automation (ICRA)*, Leuven, Belgium, 1998.

[8] H. Hirschmüller. Real-time map building from a stereo camera under constrained 3D motion. In *Faculty Research Conference*, Faculty of Computing Sciences and Engineering, De Montfort University, 2003.

[9] L. Iocchi, K. Konolige, and M. Bajracharya. Visually realistic mapping of a planar environment with stereo. In *Int. Symposium on Experimental Robotics (ISER)*, Hawaii, 2000.

[10] X.-Y. Jiang and H. Bunke. Fast segmentation of range images into planar regions by scan line grouping. *Machine Vision and Applications*, 7(2):115 – 122, 1994.

[11] S. Kagami, K. Nishiwaki, J.J. Kuffner, K. Okada, M. Inaba, and H. Inoue. Vision-based 2.5D terrain modeling for humanoid locomotion. In *Int. Conf. on Robotics and Automation (ICRA)*, Taipei, Taiwan, 2003.

[12] F. Kanehiro, H. Hirukawa, K. Kaneko, S. Kajita, K. Fujiwara, K. Harada, and K. Yokoi. Loclomotion planning of humanoid robots to pass through narrow spaces. In *Int. Conf. on Robotics and Automation (ICRA)*, New Orleans, 2004.

[13] T.-Y. Li, P.-F. Chen, and P.-Z. Huang. Motion planning for humanoid walking in a layered environment. In *Int. Conf. on Robotics and Automation (ICRA)*, Taipei, Taiwan, 2003.

[14] M.C. Martin and H.P. Moravec. Robot evidence grids. Technical report CMU-RI-TR-96-06, Carnegie Mellon University, 1996.

[15] H.P. Moravec. Dense 3D perception for broad mobile robot applications. Proposal abstract DARPA BAA0221, Carnegie Mellon University, The Robotics Institute, 2003.

[16] K. Okada, S. Kagami, M. Inaba, and H. Inoue. Plane segment finder: Algorithm, implementation, and applications. In *Int. Conf. on Robotics and Automation (ICRA)*, Seoul, Korea, May 2001.

[17] K. Sabe, M. Fukuchi, J.-S. Gutmann, T. Ohashi, K. Kawamoto, and T. Yoshigahara. Obstacle avoidance and path planning for humanoid robots using stereo vision. In *Int. Conf. on Robotics and Automation (ICRA)*, New Orleans, 2004.

[18] S. Se, D. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *Int. Journal of Robotics Research*, 21(8):735 – 758, 2002.

[19] Z. Shiller, K. Yamane, and Y. Nakamura. Planning motion patterns of human figures using a multi-layered grid and the dynamics filter. In *Int. Conf. on Robotics and Automation (ICRA)*, Seoul, Korea, 2001.

[20] S. Thrun, C. Martin, Y. Liu, D. Hähnel, R. Emery-Montemerlo, D. Chakrabarti, and W. Burgard. A real-time expectation maximization algorithm for acquiring multi-planar maps of indoor environments with mobile robots. *IEEE Transactions on Robotics and Automation*, 20(3):433 – 442, 2004.