Fast and Accurate Environment Modeling using Three-Dimensional Occupancy Grids

Katrin Pirker, Matthias Rüther, Horst Bischof
Institute for Computer Graphics and Vision
University of Technology, Graz, Austria
{kpirker,ruether,bischof}@icg.tugraz.at

Gerald Schweighofer
Institute of Digital Image Processing
Joanneum Research GmbH, Graz, Austria
gerald.schweighofer@joanneum.at

Abstract

Building a dense and accurate environment model out of range image data faces problems like sensor noise, extensive memory consumption or computation time. We present an approach which reconstructs 3D environments using a probabilistic occupancy grid in real-time. Operating on depth image pyramids speeds up computation time, whereas a weighted interpolation scheme between neighboring pyramid layers boosts accuracy. In our experiments we compare our method with a state-of-the-art mapping procedure. Our results demonstrate that we achieve better results. Finally, we present its viability by mapping a large indoor environment.

1. Introduction

With the launch of Microsoft's Kinect [1], dense three-dimensional environmental mapping has become increasingly popular. It is a basic pre-requisite in many vision applications such as human pose estimation [6], visual inspection, augmented reality or object detection [3]. Furthermore, it allows us to plan high-level robotic tasks such as path planning [18], obstacle avoidance or object manipulation.

Currently, most existing dense modeling algorithms either use a surface based or volumetric representation of the environment. Surface based approaches fit geometric primitives such as planes, spheres or cubes to three-dimensional pointcloud data [13] or construct a triangular mesh out of it. Although such methods can be computed efficiently, they face severe problems: Updating an existing surface mesh with new incoming sensor data or modeling dynamic environments requires sophisticated remeshing techniques. Because of that, many algorithms perform surface modeling in an offline step after a reconstruction pipeline. Algorithms operating online usually require some regularization during surface reconstruction to deal with disturbed or wrong depth measurements [11]. Besides, surface based approaches do not model free, occupied or unseen space, which is necessary when thinking of many robotic tasks. Using pointcloud or surface based representations, memory consumption also increases with the number of incoming sensor readings.

Volumetric approaches instead represent the surrounding as an evenly spaced grid, where each grid cell (voxel) holds a random variable of being occupied [20]. Because of their probabilistic nature, occupancy grids allow for easy integration of noisy sensor data. Furthermore, they can be used for any spatial reasoning task (i.e. grasping, path planning), since they also model free and unseen space. Different sensor modalities such as stereo and laser can be simply integrated into a single environment map [9]. Unfortunately, their accuracy highly depends on the degree of discretization and the underlying probabilistic model. Because of the huge amount of data provided by currently available range image devices, sparse environment representations are preferred to stay real-time capable.

Hence, there is a need for efficient and accurate algorithms constructing realistic environments out of dense, noisy range image data in reasonable time. In this work we propose a fast and accurate mapping routine operating on a range image device. To account for sensor noise, our environment is represented as a three-dimensional occupancy grid. In order to gain efficiency and accuracy, we combine pyramidal depth image processing together with an interpolation scheme between neighboring pyramid layers. Furthermore, a GPU implementation allows for fast pyramid generation as well as parallel voxel processing during occupancy grid mapping. Although we present its application using a depth camera, the approach can be easily adapted to any other range scanner. Synthetic experiments and comparison to groundtruth data attest the accuracy of our approach. We demonstrate that the used interpolation scheme outperforms state-of-the-art algorithms compared to groundtruth. Implementing occupancy operations and pyramid generation on the GPU allows for real-time performance.

2011 IEEE International Conference on Computer Vision Workshops
978-1-4673-0063-6/11/$26.00

2. Related Work

Dense environment mapping using some kind of range sensing device has seen great progress in the last few years. Many researchers focused on environment mapping using a 3D laser range scanner mounted on a mobile robot [4, 12]. They combined an Extended Kalman Filter for pose estimation together with scan matching routines to construct dense pointcloud maps. More recently, May et al. [8] used a single Time-of-flight camera with a modest resolution of 176×144 to construct indoor environments using an iterative closest point algorithm to register consecutive scans. Pan et al. [13] proposed a probabilistic tetrahedron carving algorithm to construct small, three-dimensional models from a structure from motion (SFM) pointcloud. Similarly, Newcombe and Davison [11] take sparse feature maps and camera poses produced by a monocular SFM algorithm and build local surface reconstructions of neighboring views which are merged afterwards. Dense environment mapping using a Kinect solely has been addressed by Henry et al. [7]. They propose an ICP algorithm using SIFT features as additional nearest neighbor constraint to establish point correspondences. Surfels [15] are further used for realistic, watertight environment modeling. Unfortunately, they do not take sensor uncertainty into account.

Many authors focused exclusively on occupancy grid mapping. First introduced by Moravec and Elfes [10] using a wide angle sonar as sensor, many researchers concentrated on sensor fusion [9] or the learning of inverse sensor models [19] for highly accurate occupancy mapping. Yguel et al. [22] proposed a sparse wavelet datastructure to speed up occupancy grid mapping. They exploit the multiscale property of the wavelet expansion resulting in a fast and efficient update procedure for 2D and 3D environments. Similarly, Andert et al. [2] employ a pyramidal approach to reduce computation time. Perrollaz et al. [14] computed an occupancy grid in the disparity space using a stereo setup. Although mapping becomes much faster, retransformation to the Cartesian space is necessary.

Other authors concentrate on memory consumption in 3D mapping and use an octree as underlying datastructure. Fairfield et al. [5] use ancestry trees for efficient octree sharing especially for particle filter based approaches. Recently, Wurm et al. [21] developed a framework for octree based mapping while focusing on memory consumption and efficiency. Ryde et al. [17] instead preferred a 2D representation storing a list of occupied voxels in each grid cell. However, just storing occupancy information, they are not able to differentiate between free and unknown space.

3. Background

Three-dimensional occupancy grids discretize the environment in an equally spaced grid, where each grid cell holds a probability for being occupied. Throughout this section we provide basic information on occupancy grid mapping and sensor modeling given a depth camera.

3.1. Occupancy Grid Mapping

Let m denote a three-dimensional environment map, $z_{1,\dots,t}$ sensor readings up to time t and $x_{1,\dots,t}$ sensor poses. In the context of mapping only, one assumes that sensor locations are known a priori. Occupancy grid mapping aims at calculating the posterior over maps m

$$p(m|z_{1,\dots,t}, x_{1,\dots,t}). \tag{1}$$

According to Moravec at al. [9] Equation 1 can be broken down to estimate the probability of each individual grid cell m_{xyz} independently:

$$p(m|z_{1,\dots,t}, x_{1,\dots,t}) = \prod_{xyz} p(m_{xyz}|z_{1,\dots,t}, x_{1,\dots,t}). \tag{2}$$

Using the more common log odds representation of occupancy and applying Bayes-Theorem, one can easily see that the log odd value l_t of a grid cell at time t can be computed from the previous one as follows [20]:

$$l_t = l_{t-1} + log\left(\frac{p(m_{xyz}|z_t, x_t)}{1 - p(m_{xyz}|z_t, x_t)}\right) - l_0 \tag{3}$$

$$l_0 = log\left(\frac{p(m_{xyz})}{1 - p(m_{xyz})}\right), \tag{4}$$

where l_0 denotes the initial belief (usually set to 0). The desired occupancy value can be recovered from l_t through:

$$p(m_{xyz}|z_{1,\dots,t}, x_{1,\dots,t}) = 1 - \frac{1}{1 + e^{l_t}}. \tag{5}$$

3.2. Probability Calculation Given a Range Image Sensor

The Kinect sensor outputs dense depth and color information at a framerate of 30 Hz at a resolution of 640×480 pixels. To perform occupancy grid mapping, one has to extract measurements for each valid depth pixel. Provided a camera calibration matrix K, the j-th pixel $(u, v)_j$ and its associated depth d_j can be interpreted as a measure z_t^j representing the Euclidean distance from the current camera center to an obstacle:

$$z_t^j(u, v, d) = \left\| \left(K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}_j d_j \right) \right\| \tag{6}$$

Hereby, camera orientation composed of rotation and translation $x_t = [R_t|T_t]$ is assumed to be known for each sensor reading.

Typically, an inverse measurement model is formulated which relates a sensor measurement z_t recorded at location

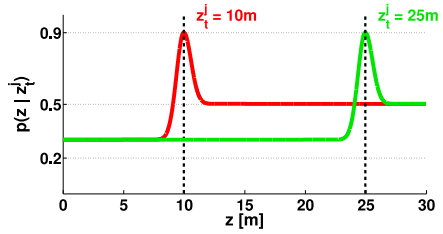


Figure 1. Inverse measurement model. Occupancy values for two distance measures at $z_t^j = \{10m, 25m\}$ using $k = \sigma = 0.6$.

x_t to a probability $p(m_{xyz}|z_t, x_t)$ for a cell being occupied. Since we do not focus on sensor modeling, we make use of a simplified version of the model proposed in [2], which is the combination of a linear function and a Gaussian together with an importance factor k and measurement uncertainty σ:

$$p(z|z_t^j) = \begin{cases} 0.3 + \left(\frac{k}{\sigma\sqrt{2\pi}} + 0.2\right) e^{-\frac{1}{2}\left(\frac{z-z_t^j}{\sigma}\right)^2}, & 0 < z \leq z_t^j \\ 0.5 + \frac{k}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{z-z_t^j}{\sigma}\right)^2}, & z > z_t^j \end{cases} \quad (7)$$

It has to be mentioned that Equation 7 should be regarded as a weighting function instead of a probability, since the sum under the curve does not necessarily sum up to 1. In general, an inverse measurement model is valid as long as the values are within the open interval $(0, 1)$. Resulting probabilities for two measurements are shown in Figure 1 using $k = \sigma = 0.6$.

3.3. Ideal Occupancy Grid Mapping

Since the environment is approximated by a discrete grid, multiple projection rays and their associated measurements may contribute to the occupancy value of a single voxel (compare Figure 2(a)).

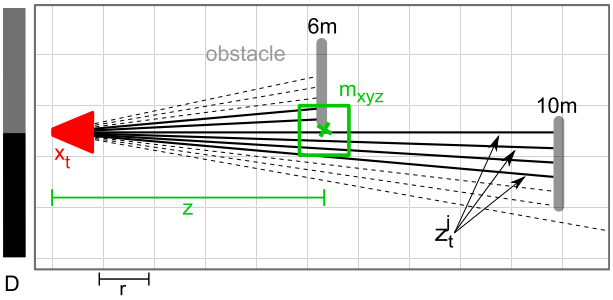
Hence, the log odd value of a voxel m_{xyz} with distance z to the camera center is computed as follows:

$$log\left(\frac{p(m_{xyz}|z_t, x_t)}{1 - p(m_{xyz}|z_t, x_t)}\right) = \sum_{z_t^j \in m_{xyz}} log\left(\frac{p(z|z_t^j)}{1 - p(z|z_t^j)}\right) \cdot \frac{len(m_{xyz} \cap z_t^j)}{\eta}, \quad (8)$$

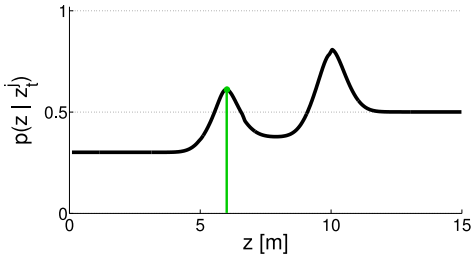
with

$$\eta = \sum_{z_t^j \in m_{xyz}} len(m_{xyz} \cap z_t^j), \quad (9)$$

where z_t^j denotes those measurements belonging to the rays intersecting voxel m_{xyz}. Each log odd is further weighted with the normalized length of the segment passing through the voxel, denoted by $\frac{len(.)}{\eta}$. Hereby, we assume infinite fine rays. A two-dimensional sketch is presented in Figure 2(a). We show the slice of a depth image with obstacles located at $6m$ and $10m$, respectively. The ideal occupancy values computed along the optical axis are shown in Figure 2(b) (using Equations 8, 7 and 5 at a grid resolution of $r = 0.1m$ and model parameters $k = \sigma = 0.6$).



(a)



(b)

Figure 2. Ideal occupancy mapping. (a) A simplified, two-dimensional sketch, where a sensor faces two obstacles at $6m$ and $10m$. A single depth image row is shown on the left. Each voxel at distance z (marked green) can be affected by one or more measurements z_t^j (solid black lines). (b) The resulting occupancy values along the optical axis are shown, using Equations 8 and 5. The occupancy value for the voxel marked green is highlighted.

4. Methodology

Given a depth image, we try to approximate the ideal mapping algorithm described in the previous section by a fast and accurate pyramidal approach. Since we are interested in real-time mapping given ideal camera poses, we propose an inverse modeling algorithm to process each voxel inside the camera view frustum in parallel using a GPU implementation. In order to increase accuracy, we propose a weighted interpolation scheme to compute each voxels occupancy value.

4.1. Pyramid Generation

Since the inverse sensor model (Equation 7) operates on distance measurements instead of depth image readings, we generate image pyramids out of the measurements. Hereby,

each depth image D can be converted to an image M containing the measurements z_t^j at $(u,v)_j$ by evaluating Equation 6. An image pyramid M_i at layer i is created by taking those measurements of M in a $2^i \times 2^i$ neighborhood closest to a given distance threshold t_i:

$$\begin{aligned} M_i(u,v) = &\{M(x,y) \mid min|M(x,y) - t_i|, \\ &x = \{u, ..., u + 2^i - 1\}, \\ &y = \{v, ..., v + 2^i - 1\}\}, \end{aligned} \tag{10}$$

where t_0 denotes the distance between a voxel and the camera center, where the size of the projected voxel in the image plane equals one pixel; and t_i is defined as

$$t_i = \frac{t_0}{2^i}. \tag{11}$$

Here, each pyramid layer is correlated to a specific distance t_i. This guarantees that we maintain the most likely measurement within a specific neighborhood around (u,v) at layer i. In order to keep important information, we approximate each pyramid layer from the full resolution image $M(x,y)$.

4.2. Pyramidal Occupancy Mapping

Since voxels can be interpreted as independent random variables, we iterate over the grid cells inside the cameras view frustum to compute their occupancy values in parallel on the GPU. Therefore, the Euclidean distance z between the voxel centroid and the current camera center is determined. In order to update its log odd value, we have to interpret the measurements a voxel is influenced. Recalling the image pyramid definition from the previous section, a voxel at distance z is affected by two neighboring pyramid layers. The following inequality is used to determine which layers are taken into account:

$$t_i < z \leq t_{i+1}. \tag{12}$$

Hence, higher pyramid layers (lower image resolutions) are used for a voxel near the camera center. This is a valid assumption since the grid cell is influenced by several neighboring pixels and therefore a lower resolution image containing compressed information can be used (compare Figure 2(a)).

We compute the image projection (u,v) of the voxel's centroid by:

$$\begin{bmatrix} u \\ v \end{bmatrix} = (K \begin{bmatrix} R \mid T \end{bmatrix} m_{xyz})_n, \tag{13}$$

where $(.)_n$ denotes the conversion from homogeneous to Euclidean coordinates. Their pyramidal counterparts $(u,v)^i$ and $(u,v)^{i+1}$ are determined through

$$(u,v)^i = \left(\frac{u}{2^i}, \frac{v}{2^i}\right). \tag{14}$$

To compute the log odd value for a grid cell we propose a two-step procedure: First, the log odds l^i and l^{i+1} corresponding to each pyramid layer are determined. Therefore, we perform bilinear interpolation of the log odds evaluated around $(u,v)^i$ and $(u,v)^{i+1}$ taking the measurements stored in M_i and M_{i+1}, respectively:

$$\begin{aligned} l^i = &(1 - <u>)(1 - <v>)\ log^i(\lfloor u \rfloor, \lfloor v \rfloor) + \\ &<u>(1 - <v>)\ log^i(\lfloor u \rfloor + 1, \lfloor v \rfloor) + \\ &(1 - <u>)<v>\ log^i(\lfloor u \rfloor, \lfloor v \rfloor + 1) + \\ &<u><v>\ log^i(\lfloor u \rfloor + 1, \lfloor v \rfloor + 1) \end{aligned} \tag{15}$$

where

$$\begin{aligned} log^i(u,v) &= log\left(\frac{p(z|M_i(u,v))}{1 - p(z|M_i(u,v))}\right) \\ <x> &= x - \lfloor x \rfloor. \end{aligned} \tag{16}$$

Second, the final log odd value $l(z)$ for a voxel at distance z is determined by linear interpolation:

$$l(z) = l^i + (z - t_i)\frac{l^{i+1} - l^i}{t_{i+1} - t_i}. \tag{17}$$

This procedure is summarized in Algorithm 1.

Algorithm 1 Pyramidal Occupancy Mapping

Input: $l_{t-1}, z_t, x_t = \{R_t, T_t\}$
  $C = -R_t' \cdot T_t$
  for all m_{xyz} inside the camera view frustum **do**
    $z = \|m_{xyz} - C\|$
    $t_i < z \leq t_{i+1}$
    compute (u,v) using Equation 13
    compute $(u,v)^i$, $(u,v)^{i+1}$ using Equation 14
    compute $\{l^i, l^{i+1}\}$ using Equations 15, 16
    compute $l(z)$ using Equation 17
    $l_{t,xyz} = l_{t-1,xyz} + l(z) - l_{0,xyz}$
  end for
Output: l_t

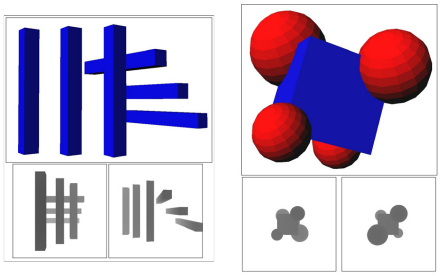
5. Experiments

To evaluate our approach we perform several synthetic and real-world experiments. We also compare our approach to the ideal occupancy values described in section 3.3 and to a state-of-the-art stereo mapping algorithm.

5.1. Synthetic Experiments

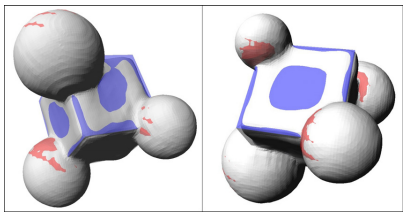
We generated two synthetic scenes named BOX ($10.8 \times 8.3 \times 11.2m^3$) and SPHERES ($2.2 \times 2.1 \times 2m^3$) containing different geometric primitives at variable depth. We captured a total of 181 depth images with a resolution of

400×400 pixels by rotating the camera around the x and y axis at $4°$. Per pixel depth values are generated through ray casting using a standard intrinsic camera calibration (focal length 400, principal point located at $(200, 200)$). Scenery and depth images from different camera views are shown in Figure 3.

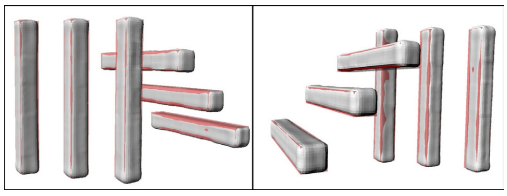


(a) BOX experiment. (b) SPHERES experiment.

Figure 3. Synthetic experiments. (a) Scene containing six objects of $1m$ width at different depths and two example depth images. (b) Synthetic scene containing a box and four spheres of variable size and their depth images.



(a) SPHERES experiment.



(b) BOX experiment.

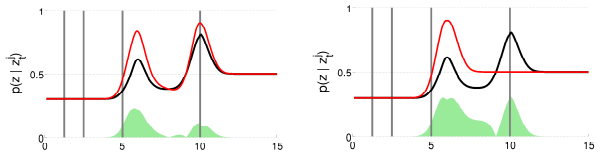
Figure 4. Grid mapping results. The three-dimensional occupancy grids (gray) are shown as iso-surface and compared to groundtruth data (red and blue).

The resulting occupancy grids computed at a resolution of $0.02m$ are shown in Figure 4 with their groundtruth data overlaid. For demonstration purpose occupancy grids are shown as iso-surfaces at level 0.5. The iterative mapping procedure is also visualized in the supplementary material.

5.2. Accuracy Evaluation

Regarding accuracy we compared our approach against a reimplementation of a state-of-the-art stereo mapping procedure proposed by Andert et al. [2]. First, we run both approaches on the synthetic example depicted in Figure 2 and computed the occupancy values along the optical axis, assuming a grid resolution of $0.1m$. The resulting occupancy values (red) together with the absolute errors (green) to the ideal occupancy values (black) are shown in Figure 5.

Furthermore, we do some visual inspection of the occupancy grid itself by processing a single depth image of the BOX scene with both algorithms at a grid resolution of $0.5m$[1]. Two single slices from a side- and a top-view are presented in Figure 6.



(a) Our approximation. (b) Stereo approximation.

Figure 5. Accuracy evaluation. We compared our approach (a) as well as the procedure proposed in [2] (b) to the ideal occupancy values (black) at a grid resolution of $0.1m$. The absolute errors are shown as green area plot, respectively. Pyramid layers are marked as vertical lines.

(a) Side View. (b) Top View.

Figure 6. Algorithm comparison. We show a single slice of the occupancy grid from the side (a) and the top (b) after processing a depth image from the BOX scene. We compare our approach (left image) against the procedure proposed by [2] (right image).

5.3. Real World Experiment

One possible application would be to generate realistic, dense representations of indoor environments. Therefore, we acquired an indoor sequence of a spacious environment consisting of narrow corridors and a large meeting room using a handheld Kinect. In total we traveled a loop of $60m$ resulting in more than 1600 frames. Provided 684 corrected camera poses stemming from a keyframe-based Simultaneous Localization and Mapping pipeline [16] we construct a dense, three-dimensional occupancy grid at a resolution of $0.02m$ per voxel.

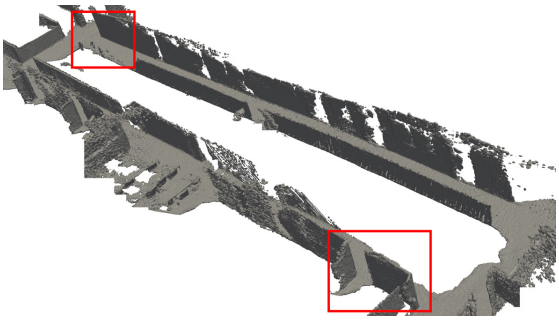
The resulting iso-surface at the zero-level of the log odd values is shown in Figure 7 (a). As shown in the close up views in Figures 7 (b),(c) we are able to construct detailed surface information like doors or picture frames. It takes

[1] example code can be downloaded from http://rvlab.icg.tugraz.at/project_page/project_slam/project_slam.htm

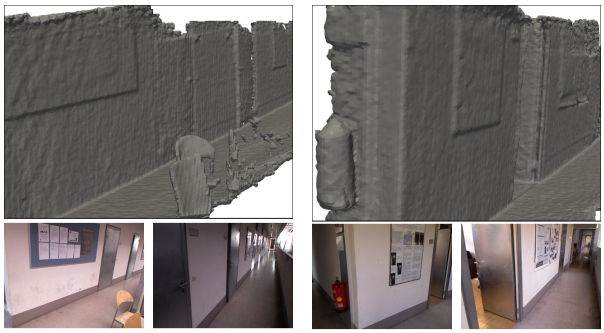
us 6.8 minutes to compute the occupancy grid using 684 frames and the appropriate depth images. Further timings and memory consumption at various resolutions are listed in Table 1. For navigation tasks, the resolution of $0.08m$ is precise enough, resulting in an update rate (framerate) of $14.07fps$.

res [m]	**0.02**	**0.04**	**0.08**	**0.16**
time [min]	6.8	1.51	0.81	0.57
memory [GB]	4.23	0.99	0.30	0.06

Table 1. Time and memory consumption. While processing the indoor sequence at various resolutions, we recorded processing time and memory requirements of the final grid.



(a)



(b) Close up. (c) Close up.

Figure 7. Real world experiment. Provided camera poses from a Simultaneous Localization and Mapping procedure, we take the appropriate depth images as input. (a) The resulting occupancy grid is shown as iso-surface at the zero-level of the log odd values. (b),(c) Close ups of the occupancy grid at the regions marked red with the appropriate RGB images.

6. Discussion and Conclusion

We developed a fast and accurate algorithm constructing dense, volumetric environments using sensor readings from a consumer depth camera. To speed up computation time we used an inverse mapping procedure paired with pyramidal depth image processing both implemented on the GPU. The proposed interpolation scheme between neighboring image pyramids results in more accurate occupancy values. In contrast to a state-of-the-art stereo approach, we presented a different pyramid generation, which is more suitable for the used sensor model. Furthermore, interpolating between both pyramid layers results in occupancy values closer to groundtruth. As shown in Figure 5 our approach takes both measurements into account whereas the stereo algorithm prefers closer obstacles by taking the minimum depth of a single pyramid layer. This is also observed in Figure 6, where our occupancy grid indicates free space instead of unseen. As stated in [2] the resulting grid should also be blurred after the mapping procedure to consider angular sensor uncertainty. Using our approach this is handled implicitly (compare the smoothed edges in Figure 6). The developed mapping procedure will be further used for higher level robotic tasks such as view suggestion within a probabilistic mapping procedure.

7. Acknowledgement

This work was supported by the Austrian Research Promotion Agency (FFG) and Federal Ministry of Economics, Family Affair and Youth (BMWFJ) within the Austrian research Studio Machine Vision Meets Mobility.

References

[1] Microsoft KINECT. www.xbox.com/kinect. 1

[2] F. Andert. Drawing stereo disparity images into occupancy grids: measurement model and fast implementation. In *International Conference on Intelligent Robots and Systems*, pages 5191–5197, 2009. 2, 3, 5, 6

[3] S. Y. Bao and S. Savarese. Semantic structure from motion. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 2011. 1

[4] D. Cole and P. Newman. Using laser range data for 3D SLAM in outdoor environments. In *IEEE International Conference on Robotics and Automation*, pages 1556 –1563, 2006. 2

[5] N. Fairfield, G. A. Kantor, and D. Wettergreen. Real-time SLAM with octree evidence grids for exploration in underwater tunnels. *Journal of Field Robotics*, 2007. 2

[6] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun. Real time motion capture using a single time-of-flight camera. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 755 –762, 2010. 1

[7] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using depth cameras for dense 3d modeling of indoor environments. In *International Symposium on Experimental Robotics*, 2010. 2

[8] S. May, S. Fuchs, D. Droeschel, D. Holz, and A. Nüchter. Robust 3D-Mapping with Time-of-Flight Cameras. In *International Conference on Intelligent Robots and Systems*, pages 1673–1678, 2009. 2

[9] H. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Mag.*, 9:61–74, 1988. 1, 2

[10] H. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 116 – 121, 1985. 2

[11] R. Newcombe and A. Davison. Live dense reconstruction with a single moving camera. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1498 –1505, 2010. 1, 2

[12] A. Nuechter, K. Lingemann, J. Hertzberg, and H. Surmann. 6d slam—3d mapping outdoor environments: Research articles. *J. Field Robot.*, 24(8-9):699–722, 2007. 2

[13] Q. Pan, G. Reitmayr, and T. Drummond. ProForma: Probabilistic feature-based on-line rapid model acquisition. In *British Machine Vision Conference*, 2009. 1, 2

[14] M. Perrollaz, J.-D. Yoder, A. Spalanzani, and C. Laugier. Using the disparity space to compute occupancy grids from stereo-vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2721 –2726, 2010. 2

[15] H. Pfister, M. Zwicker, J. V. Baar, and M. H. Gross. Surfels: surface elements as rendering primitives. In *Annual Conference on Computer Graphics*, pages 335–342, 2000. 2

[16] K. Pirker, M. Rüther, and H. Bischof. GPSlam: Marrying sparse geometric and dense probabilistic visual mappgin. In *British Machine Vision Conference*, To appear. 5

[17] J. Ryde and H. Hu. 3D mapping with multi-resolution occupied voxel lists. *Autonomous Robots*, 28:169–185, 2010. 2

[18] R. Shade and P. Newman. Choosing Where To Go: Complete 3D exploration with stereo. In *IEEE International Conference on Robotics and Automation*, 2011. 1

[19] S. Thrun. Learning occupancy grids with forward models. In *IEEE Conference on Intelligent Robots and Systems*, pages 1676–1681, 2001. 2

[20] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. 1, 2

[21] K. M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In *Proc. of the ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation*, 2010. 2

[22] M. Yguel, C. T. M. Keat, C. Braillon, C. Laugier, and O. Aycard. Dense mapping for range sensors: Efficient algorithms and sparse representations. In *Robotics: Science and Systems*, 2007. 2