GPSlam: Marrying Sparse Geometric and Dense Probabilistic Visual Mapping

Katrin Pirker [1]
kpirker@icg.tugraz.at
Gerald Schweighofer[2]
gerald.schweighofer@joanneum.at
Matthias Rüther[1]
ruether@icg.tugraz.at
Horst Bischof[1]
bischof@icg.tugraz.at

[1] Institute for Computer Graphics and Vision
Graz University of Technology
Graz, Austria

[2] Institute of Digital Image Processing
Joanneum Research GmbH
Graz, Austria

Abstract

We propose a novel, hybrid SLAM system to construct a dense occupancy grid map based on sparse visual features and dense depth information. While previous approaches deemed the occupancy grid usable only in 2D mapping, and in combination with a probabilistic approach, we show that geometric SLAM can produce consistent, robust and dense occupancy information, and maintain it even during erroneous exploration and loop closure. We require only a single hypothesis of the occupancy map and employ a weighted inverse mapping scheme to align it to sparse geometric information. We propose a novel map-update criterion to prevent inconsistencies, and a robust measure to discriminate exploration from localization.

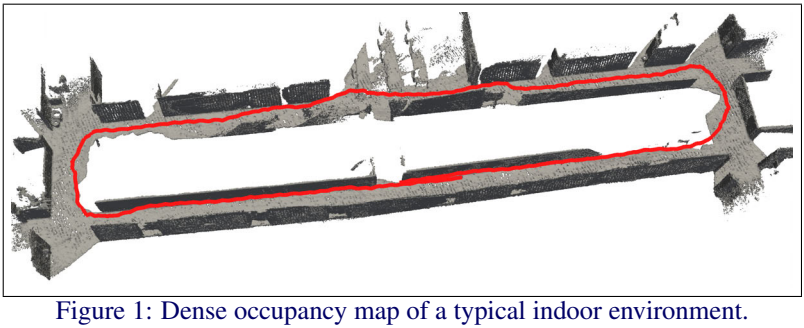


Figure 1: Dense occupancy map of a typical indoor environment.

1 Introduction

The ultimate goal of every visual localization and mapping procedure is building a dense three-dimensional representation of the environment. In contrast to sparse pointcloud maps, it allows to plan high-level robotic tasks such as autonomous navigation, path optimization,

BMVC 2011 http://dx.doi.org/10.5244/C.25.115

object manipulation and obstacle avoidance. When building dense environment maps one has to address the problems of sensor and pose uncertainty, memory consumption and geometric correction of the map.

Currently, most existing algorithms follow either a pure geometric or probabilistic approach to solve the Simultaneous Localization and Mapping (SLAM) problem. Geometric approaches, like Structure from Motion (SfM), outperform probabilistic methods in terms of accuracy and speed [25]. They allow real-time tracking and mapping even in large scale environments as well as fast and robust loop closure detection and correction. Usually, triangular meshes [20] or geometric primitives [21] are fitted to the pointcloud data to construct a dense surface representation. To handle misaligned, noisy or systematically wrong depth measurements some form of registration and regularization is needed.

Probabilistic environment modeling using occupancy grids allow easy integration of noisy range data, and are widely employed in 2D mapping [7, 19] using a Particle Filter for pose estimation. Unfortunately, their accuracy highly depends on the probabilistic model, the sampling scheme (i.e. the number of particles used) and the degree of map discretization. Robust Monte-Carlo based methods are memory and time consuming, because every particle has to hold a hypothesis of its own map. An extension to 3D SLAM and a volumetric map quickly leads to an explosion of the number of particles, memory per particle, and computation time.

Hence, a SLAM system would greatly benefit from combining the positive aspects of geometric and probabilistic approaches. Provided accurate sensor poses, sensor noise, scene dynamics and dense modeling would be implicitly handled by a probabilistic map building routine. Geometric localization within sparse feature maps supplies accuracy at relatively low computational cost. By incorporating dense occlusion information into sparse mapping, one would boost accuracy and speed even further, and also allow to new visual features [29].

In this work we propose a novel SLAM system which fuses geometric and probabilistic methods to create a sparse geometric map, and a dense, volumetric occupancy grid, using an RGBD camera. We allow for fast and accurate pose estimation within a sparse, three-dimensional feature map using state-of-the-art incremental Structure from Motion techniques. Simultaneously we compute a single dense occupancy grid. Using only a single map hypothesis, our method is not as memory consuming as traditional probabilistic approaches. To tackle the problem of geometric correction after loop closure we propose a weighted inverse mapping procedure. Furthermore, sparse geometric information coupled with probabilistic data allows us to robustly discriminate exploration movements (i.e. traveling in unknown terrain and enlarging the map) from localization movements (i.e. traveling in previously seen terrain and enhancing the map). During exploration we prevent self-destruction of the occupancy grid through submapping and a robust criterion for detecting such map conflicts. Successful mapping of two indoor sequences demonstrates that we are able to construct dense three-dimensional environments during the exploration phase and repair the underlying occupancy grid after loop closure. Real-time performance is achieved through implementing occupancy operations on the GPU.

2 Related Work

There exist few approaches to incrementally build dense, three-dimensional maps based on monocular visual information. Pan et al. [21] construct small, textured, three-dimensional models from a Structure from Motion pointcloud by employing probabilistic tetrahedron

carving. Newcombe and Davison [20] combine two approaches to perform monocular dense reconstruction. Taking sparse feature maps and camera poses produced by a real-time SfM algorithm, GPU-based optical flow estimates dense correspondences between keyframes. These are used to build local reconstructions, which are merged afterwards. Both approaches are evaluated on a small desktop environment and currently do not scale to map sizes suitable for robotic applications.

Little work has been presented on SLAM methods using range image devices. Joochim and Roth [13] use a time of flight (TOF) camera to produce a dense pointcloud followed by an iterative closest point (ICP) algorithm to register the range scans. To overcome the limited resolution, a separate CCD camera is used for feature tracking and motion estimation. Mapping larger environments with a single TOF sensor has been proposed by May et al. [18]. Again an ICP algorithm is used to pairwise register consecutive range scans followed by a graph based algorithm [26] to compensate for accumulated registration errors. Similarly, Henry et al. [12] build large scale reconstructions of an indoor environment using a variant of the ICP algorithm together with a loop closing routine. Surfels [22] are used for realistic environment modeling and handling of occlusion information. Sensor uncertainty is not considered during incremental map building, and it is unclear how the underlying map is corrected after loop closure.

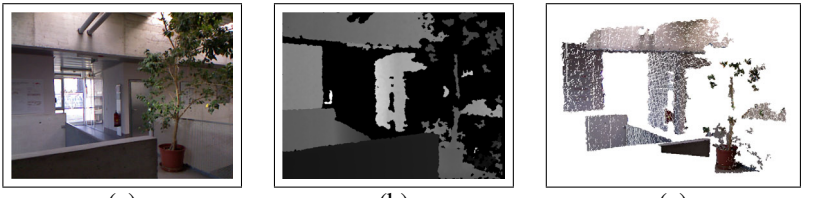
In visual SLAM, few methods employ an occupancy grid as the underlying map representation. To reduce memory consumption in a Particle Filter, Marks et al. [17] diminish the map to 2D occupancy grids holding the height of potential obstacles in each cell. Similarly, Koch et al. [14] generate depth maps from neighboring views of a desktop environment and fuse them in voxel space. Other approaches focus on building highly efficient grid mapping algorithms. To allow for real-time updates, GPU-implementations [16] or multiresolution approaches [2] are proposed. Others focus on reducing memory consumption in 3D mapping, with multi-volume grids [6] or octree representations [30].

Dense visual mapping is more prominent in the field of photogrammetry, where time and computational limitations are not as strict. Furukawa et al. [8] decompose unstructured images into overlapping clusters. A multi-view stereo algorithm reconstructs dense pointclouds which are merged into a single model afterwards. The problem of dense urban scene modeling was addressed by Gallup et al. [9, 10] using street level video. They represent the surface by simplified 2.5D heightmaps and fuse them over multiple views resulting in detailed facade reconstructions. Similarly, Pollefeys et al. [23] build urban scene models out of geo-registered video frames. Given camera projection matrices estimated from a Kalman Filter they perform stereo depth map estimation and fusion followed by the creation of a smooth triangle mesh. Both methods do not address the problem of loop closing within their framework.

3 Geometric Occupancy Map Building

Keyframe-based geometric SLAM is able to perform fast and accurate pose estimation in a previously unknown world, typically up to metric scale. A range imaging device, like an RGBD camera, additionally delivers a metrically correct depth map at video frame rate (Figure 2). If the depth map is used to update occupancy in a static voxel grid, one quickly runs into the problems of self-destruction and geometric correction. The first problem arises during exploration, where the accumulated pose estimation error might cause a false overlay of the current pose and previously explored space. While this problem is easy to solve in

sparse geometric SLAM, where a binding always exists between cameras and observed map points, false updates of occupancy cannot be repaired later on. The second problem arises during loop closure, where the camera trajectory might be mapped to a completely different configuration, and the occupancy grid should follow this mapping.



(a) (b) (c)

Figure 2: Data delivered from the range image device: (a) RGB image. (b) Gray coded depth information. (c) Color coded pointcloud.

In this Section, the naive approach to visual occupancy grid mapping is introduced. Our solutions to the above mentioned problems are presented in Section 4.

3.1 Geometric Pose Estimation

Our environment is represented by three-dimensional feature points $X = [x\ y\ z]$ and camera poses $P = [R|T]$ (denoted keyframes), both located in a global world coordinate frame. To every feature point we attach a SIFT-descriptor [15], which is used for feature extraction and robust data association (when implemented on the GPU real-time performance is achieved [4]).

In every incoming frame keypoints are detected and matched against potentially visible map points, resulting in 2D-3D correspondences. The new camera pose is estimated by robust minimization of the reprojection error. A new keyframe P_t is added, if the camera has moved at least a predefined amount (e.g. 3° or 10cm), or if the number of visible map points falls below a threshold. New map points X are initialized from the depth measurement as

$$X = (K P_t)^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} d, \tag{1}$$

where K is the camera calibration matrix, and $[u\ v]$ denote the image point with its associated depth d.

Sliding-window bundle-adjustment is applied to increase pose estimation accuracy in a local neighborhood of cameras (e.g. the most recent 10 keyframes). The problem of scale-drift, which is inevitably present in monocular, incremental Structure from Motion, is circumvented by incorporating the depth measurements of visual features.

Traditional bundle adjustment [28] estimates camera poses P_i and feature point locations X_j by minimizing the reprojection error

$$err_{re} = \sum_{i,j} \left[\begin{pmatrix} u \\ v \end{pmatrix}_{ij} - (K P_i X_j)_n \right]^2, \tag{2}$$

given a number of image measurements $[u\ v]_{ij}$ (image point of feature X_j in keyframe P_i).

Hereby $(.)_n$ denotes the conversion from homogeneous to Euclidean coordinates. We introduce an additional error measure dependent on the measured depth d_{ij}:

$$err_d = \sum_{i,j} \left[d_{ij} - \left(P_i^{3T} X_j \right)_n \right]^2, \tag{3}$$

where $\left(P_i^{3T} X_j \right)_n$ is the depth of a map point X_j in camera pose P_i with its principal plane P_i^{3T}. Thus the overall error to be minimized is summarized as

$$err = err_{re} + err_d. \tag{4}$$

Since both error measures are in a different range, outliers are handled by employing a robust Huber cost function [11] to both measures separately.

Detection of loop closure is handled by the FAB-MAP algorithm [5], followed by a geometric consistency check, which robustly delivers a single potential loop closing frame P_l. Once a loop has been detected, we establish 2D-3D correspondences between the current frame P_c and map points visible from P_l. The corrected $\tilde{P}_c$ is estimated by minimizing the combined error stated in Equation 4. To close the loop we employ a pose graph optimization algorithm [24], which minimizes the discrepancy between relative constraints of subsequent poses $P_t P_{t-1}^{-1}$ and the loop closure constraint $P_l \tilde{P}_c^{-1}$. Corrected map points $\hat{X}$ are computed from optimized poses $\hat{P}_t$ as

$$\hat{X} = median[\hat{P}_t^{-1} (P_t X)]. \tag{5}$$

3.2 Probabilistic Map Building

A three-dimensional occupancy grid discretizes the environment in equally sized voxels, which hold a probability for occupancy. Each voxel stores its probability p at time t in log-odd form o_t

$$o_t = log\left(\frac{p}{1-p}\right), \tag{6}$$

which allows for fast additive map updates

$$o_t = o_{t-1} + log\left(\frac{p}{1-p}\right) - o_0, \tag{7}$$

where o_0 denotes the initial belief (usually set to 0) [27].

Every depth measurement $d(u,v)$ delivered by the sensor defines a vector originating from the camera center C_t with length z^*:

$$z^* = \| \left((K P_t)^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} d \right)_n - C_t \|. \tag{8}$$

Typically, an inverse sensor model is formulated which relates a measurement z^* to an occupancy value p. Since we do not focus on sensor modeling, we make use of a simplified version of the model proposed in [2], depending on z^* and its associated uncertainty σ_{z^*}:

$$p(z|z^*, \sigma_{z^*}) = \begin{cases} \left[e^{-\left(\frac{z-z^*}{\sqrt{2}\sigma_{z^*}}\right)^2} - \frac{1}{2} \right] \frac{k}{\sigma_{z^*}} + \frac{1}{2}, & 0 < z \leq z^* \\ \left[e^{-\left(\frac{z-z^*}{\sqrt{2}\sigma_{z^*}}\right)^2} \frac{1}{2} \right] \frac{k}{\sigma_{z^*}} + \frac{1}{2}, & z > z^*. \end{cases} \tag{9}$$

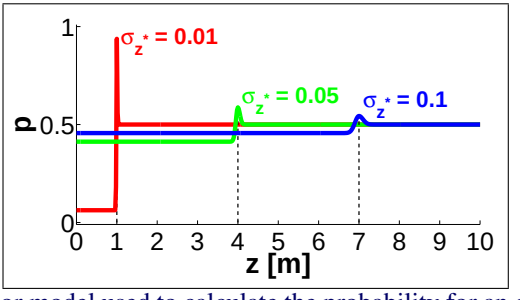


Figure 3: Inverse sensor model used to calculate the probability for an obstacle at distance z. We show occupancy values given three distance measurements $z^* = \{1,4,7\}$ together with their uncertainties $\sigma_{z^*} = \{0.01,0.05,0.1\}$

Figure 3 shows occupancy probabilities for three different distance measurements z^*. Hereby, near distance measurements are considered to be more accurate than far ones and assigned a higher peak probability, and a smaller confusion range. Because voxels can be regarded independent from each other, we implemented a GPU-based ray casting, where all voxels inside the view frustum of the current camera are updated in parallel.

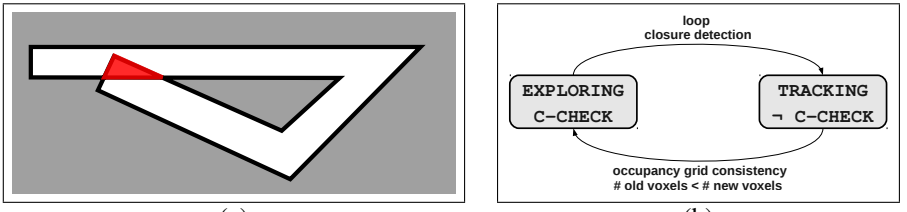
4 Consistency Handling

Sparse localization and dense occupancy grid mapping allow accurate pose estimation and incremental dense surface modeling. Sensor noise is handled by the probabilistic mapping routine, but error propagation in visual odometry is not. Even though we incorporate metric depth information in pose estimation, errors accumulate over time. This results in wrong trajectories and maps which can not be corrected until loop closing (see Figure 7). As a consequence, we may destroy an existing map with incoming sensor data because of wrong map overlaps. Such a conflict is sketched in Figure 4 (red). Further, we need to robustly warp the occupancy grid according to a global change in the camera trajectory, as it occurs in a loop closure.

We first introduce a robust criterion to discriminate exploration from pure localization. Hereby, a camera is meant to be in exploration stage if it travels in previously unseen terrain while expanding the map. In localization mode instead, a sensor moves in already mapped areas and enhances/updates its map. During localization, overlaps with the existing occupancy grid map are natural and depth measurements can be safely used for updating. Map overlaps during exploration instead, are likely to produce map inconsistencies and should be detected as a conflict. On conflict, we split the occupancy grid into local submaps, which are merged again after loop closure.

4.1 Conflict Detection during Map Updates

To discriminate valid map updates of subsequent cameras from conflicting ones, as shown in Figure 4, we add an additional value o_L to each voxel, which stores the traveled distance of the camera it has been most recently updated from. More precisely, we define three points $p_i^t, i = 1,\ldots,3$ in the view frustum of camera P_t. The first one is the camera center, the other two are located at the image border, at maximum depth range.



(a) (b)

Figure 4: Inconsistency handling: (a) Motion estimation errors can lead to the destruction of the underlying occupancy grid. After detecting an inconsistent map update, a new submap is created. (b) A simple state machine describes the transitions between tracking and exploring mode using image based loop closure detection as well as occupancy grid information.

The camera maintains a measure of its overall traveled distance l^t through

$$l^t = l^{t-1} + max(|p_i^{t-1} - p_i^t|). \tag{10}$$

By taking the distance-maximum of three non-collinear points, l^t is strictly monotonically increasing, even under pure rotation. Each time a voxel is considered for update, we check for a map conflict through:

$$l^t - o_L > T, \tag{11}$$

given a user define threshold T. If at least 1% of all voxels inside the view frustum indicate a conflict we start a new submap. Otherwise, we set $o_L = l^t$ and continue. In tracking mode, we do not check for any inconsistent map updates and o_L is only set for previously unseen voxels.

To robustly discriminate tracking from exploration, we employ a simple state-machine. On successful relocalization we start in tracking mode, otherwise in exploration mode. Only a loop closure allows us to enter a previously seen place and changes our state. After a loop correction eventually created submaps have to be aligned with its sparse counterpart and merged, which is described in Section 4.2. On the other hand, if the number of "old" voxels in the current view cone is smaller than the number of "new" ones, we expect to be in exploring mode. Hereby, a voxel is considered to be "old" if Equation 11 is fulfilled. A schematic view describing the behavior of the algorithm is shown in Figure 4(b).

4.2 Occupancy Grid Morphing

Especially after loop closure, occupancy grids have to be aligned to its sparse counterpart. Provided erroneous camera poses P_t (associated with a submap) and a globally corrected trajectory $\hat{P}_t$, we transform each submap into a globally consistent map such that we retain the probabilistic information included in all submaps.

To avoid holes in the resulting occupancy grid we propose an inverse mapping procedure for each voxel lying in the view cones of the corrected camera poses (compare Figure 5). For each voxel $\hat{X}$ in the target grid we determine the cameras $\hat{P}_i$ it is visible in, together with distances z_i of the voxel to the camera centers. Its corresponding location X_i in a submap is calculated as:

$$X_i = P_i^{-1}(\hat{P}_i \hat{X}). \tag{12}$$

A weighted average of the log-odds at X_i results in an occupancy value for $\hat{X}$:

$$\hat{X} = \frac{\sum_i logodd(X_i)\,\frac{1}{z_i}}{\sum_i \frac{1}{z_i}}. \tag{13}$$

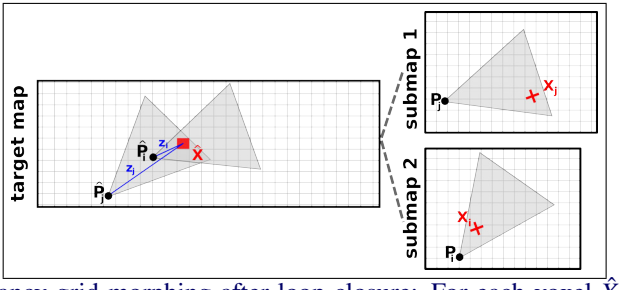


Figure 5: Occupancy grid morphing after loop closure: For each voxel $\hat{X}$ in the resulting grid we determine the cameras $\hat{P}_i$ it is visible in. Using the known camera poses before and after the loop closing, we calculate its potential previous positions. Their log-odds weighted with the distances to the cameras $\hat{P}_i$ defines the new occupancy value.

5 Experiments

To evaluate our approach we acquired two indoor sequences using a calibrated [3], handheld Kinect [1]. The Kinect sensor outputs dense depth and color information at a framerate of $30Hz$ in VGA resolution. The first sequence is a 360° view of a typical office environment consisting of 742 frames. A more spacious environment is covered by the second sequence consisting of narrow corridors and a large meeting room while traveling a loop of $60m$ resulting in 1600 frames. Incremental localization and map building while simultaneously constructing a single, dense occupancy grid as described in Sections 3 and 4 results in 167 and 662 keyframes, respectively. Throughout our experiments we use a grid resolution of $0.02m$. Measurement uncertainty σ_{z^*} used in Equation 9 is calculated according to [2], where k is approximately 90% of the minimum depth uncertainty. To make the effect of the proposed consistency check and grid morphing routine visible, we deliberately added 0.1° to each camera during visual odometry.

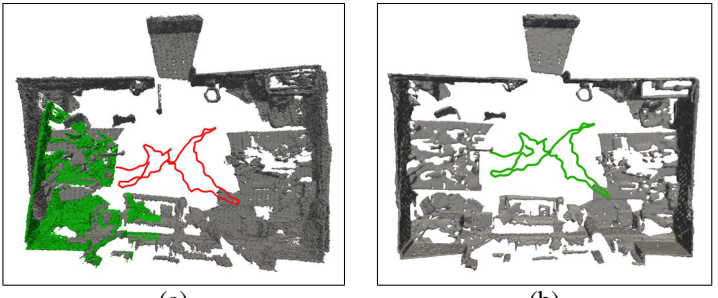
As shown in Figures 6(a) and 7 we are able to detect inconsistent grid updates which would destroy an already existing map. Two and three submaps have been established during the exploration phase. After loop closure detection, submaps are merged successfully into a single occupancy grid. The resulting surface and the corrected trajectories of both sequences are presented in Figures 6(b) and 1. For demonstration purpose, occupancy grids are rendered as isosurfaces at the zero-level of the log-odd values.

Recorded image sequences are tested on a PC featuring an Intel dual-core processor at $3.16GHz$ and an NVidia GeForce 9600 GT series GPU. Per frame operations like keypoint detection, data association and pose refinement are computed in $33ms$, $27ms$ and $23ms$, respectively. Occupancy grid updates and sliding window bundle adjustment are computed every keyframe only and require $500ms$ and $40ms$. For robotic navigation tasks a grid resolution of $0.08m$ would be accurate enough and reduces computation time from $500ms$ to

71*ms*. Further timings for occupancy operations and memory requirements of the final grid of the office sequence at various resolutions are listed in Table 1.

res [m]	0.02	0.04	0.08	0.16
time [fps]	1.68	7.55	14.07	20
memory [GB]	4.23	0.99	0.3	0.06

Table 1: Timings and memory consumption of the final occupancy grid of the office sequence at various grid resolutions.



(a) (b)

Figure 6: Office Sequence: (a) Two submaps (gray and green) are created after detecting an invalid grid update during the exploration phase. (b) Resulting office environment after loop closure detection and correction.

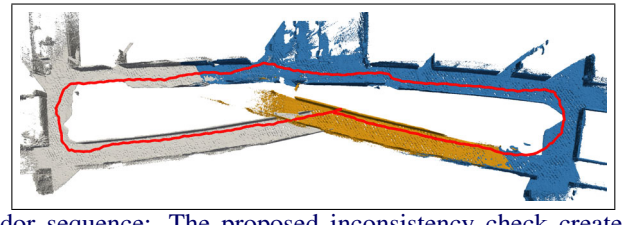


Figure 7: Corridor sequence: The proposed inconsistency check creates three submaps marked gray, blue and orange.

6 Conclusion

In this work we presented a novel visual SLAM system capable of performing accurate pose estimation within a sparse, geometric map while producing dense occupancy information. We maintain a valid occupancy grid even during inaccurate motion estimation by defining rules to detect inconsistent grip updates. Even complex motions resulting in multiple grid intersections can be handled, since conflict detection is just performed in the current submap. To align erroneous submaps to globally corrected camera poses we employ a weighted inverse mapping routine. Hereby we take care to incorporate the probabilistic information of all submaps. Our experiments indicate that we are able to construct and preserve dense, consistent environment information even under wrong motion estimation. Even multiple, nested loops can be handled with the proposed procedure. One could either perform global

morphing over the whole trajectory or align only inconsistent map parts to an already corrected map. When performing continuous localization and map building, dynamic changes like opened or closed doors or moved furniture are implicitly handled by the underlying occupancy grid. Moreover, sparse features falling into voxels with a low probability can be deleted. In future work we will incorporate more sophisticated features employing the three-dimensional information delivered by the occupancy grid allowing a more robust data association.

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. URL www.xbox.com/kinect.

[2] Franz 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. ISBN 978-1-4244-3803-7.

[3] J. Y. Bouguet. Camera calibration toolbox for Matlab, 2008. URL http://www.vision.caltech.edu/bouguetj/calib_doc/.

[4] B. Clipp, Jongwoo Lim, J.-M. Frahm, and M. Pollefeys. Parallel, real-time visual slam. In *EEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3961 –3968, 2010.

[5] Mark Cummins and Paul Newman. FAB-MAP: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research*, 27(6): 647–665, 2008.

[6] I. Dryanovski, W. Morris, and Jizhong Xiao. Multi-volume occupancy grids: An efficient probabilistic 3D mapping model for micro aerial vehicles. In *International Conference on Intelligent Robots and Systems*, pages 1553 –1559, 2010.

[7] Austin Eliazar and Ronald Parr. DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks. *IJCAI*, pages 1135–1142, 2003.

[8] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. Towards internet-scale multi-view stereo. In *Conference on Computer Vision and Pattern Recognitio*, 2010.

[9] D. Gallup, J.M. Frahm, and M. Pollefeys. A heightmap model for efficient 3d reconstruction from street-level video. In *3DPVT10*, 2010.

[10] David Gallup, Jan-Michael Frahm, and Marc Pollefeys. Piecewise planar and non-planar stereo for urban scene reconstruction. In *Computer Vision and Pattern Recognition*, 2010.

[11] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[12] Peter Henry, Michael Krainin, Evan Herbst, Xiaofeng Ren, and Dieter Fox. RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In *International Symposium on Experimental Robotics*, 2010.

[13] C. Joochim and H. Roth. The indoor slam using multiple three dimension sensors integration. In *IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, pages 500 –504, 2009.

[14] Reinhard Koch, Marc Pollefeys, and Luc Van Gool. Robust calibration and 3D geometric modeling from large collections of uncalibrated images. In *DAGM*, pages 413–420, 1999.

[15] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[16] Christian Laugier Manuel Yguel, Olivier Aycard. Efficient GPU-based construction of occupancy grids using several laser range-finders. *International Journal of Vehicle Autonomous Systems 2008*, 6:48–83, 2007.

[17] T. K. Marks, A. Howard, M. Bajracharya, G. W. Cottrell, and L. Matthies. Gamma-SLAM: Using stereo vision and variance grid maps for SLAM in unstructured environments. In *IEEE International Conference on Robotics and Automation*, pages 3717–3724, 2008.

[18] Stefan May, Stefan Fuchs, David Droeschel, Dirk Holz, and Andreas Nüchter. Robust 3D-Mapping with Time-of-Flight Cameras. In *International Conference on Intelligent Robots and Systems*, pages 1673–1678, 2009.

[19] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, 2002.

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

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

[22] Hanspeter Pfister, Matthias Zwicker, Jeroen Van Baar, and Markus H. Gross. Surfels: surface elements as rendering primitives. In *Annual Conference on Computer Graphics*, pages 335–342, 2000.

[23] M. Pollefeys, D. Nistér, J.-M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S.-J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles. Detailed real-time urban 3D reconstruction from video. *International Journal of Computer Vision*, 78:143–167, 2008. ISSN 0920-5691.

[24] H. Strasdat, J. M. M. Montiel, and A. Davison. Scale drift-aware large scale monocular slam. In *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, June 2010.

[25] Hauke Strasdat, J. M. M. Montiel, and Andrew J. Davison. Real-time monocular SLAM: Why filter? In *IEEE International Conference on Robotics and Automation*, pages 2657–2664, 2010.

[26] Sebastian Thrun and Michael Montemerlo. The graph slam algorithm with applications to large-scale mapping of urban structures. *Int. J. Rob. Res.*, 25:403–429, 2006.

[27] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. ISBN 0262201623.

[28] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment – a modern synthesis. In *Proc. Vision Algorithms*, 1999.

[29] Changchang Wu, Brian Clipp, Xiaowei Li, Jan-Michael Frahm, and Marc Pollefeys. 3D model matching with viewpoint-invariant patches (VIP). *Computer Society Conference on Computer Vision and Pattern Recognition*, 0:1–8, 2008.

[30] 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.