2008 IEEE/RSJ International Conference on Intelligent Robots and Systems
Acropolis Convention Center
Nice, France, Sept, 22-26, 2008

A sensor-independent approach to RBPF SLAM - Map Match SLAM applied to Visual Mapping

Christof Schroeter, Horst-Michael Gross

***Abstract*— In this paper, we present the application of our generic, sensor-independent Map Match SLAM framework to visual mapping. In our previous work [14], we have introduced the Map Match SLAM approach for mapping with sonar range readings: Extending the grid-based Rao-Blackwellized Particle Filter SLAM approach, in Map Match SLAM, a local map is maintained by each particle in addition to the global map. The local map is used to represent the most recent observations, and weighting of the particles is done based on the compliance of the local and the global map.**

In this paper, we show how RBPF SLAM can also be applied for mapping and path reconstruction with a stereo camera or a single monocular camera, respectively. By mapping with completely different sensors such as sonar, stereo, or monocular cameras, we prove the wide range applicability of RBPF SLAM and our Map Match SLAM computational framework.

I. INTRODUCTION AND RELATED WORK

Simultaneous Localization and Mapping (SLAM) describes the problem of mapping an unknown environment without any external position reference [1]., e.g. GPS. In that case, the consistency of the map depends on consistent knowledge of the current position, while robust self-localization requires an accurate map of the environment. Therefore, the localization and the mapping problem are coupled. The mutual dependency between pose and map estimates requires to model the state of the robot in a high-dimensional space, consisting of a combination of the pose and map state.

The SLAM problem has drawn a lot of attention from researchers in recent years and a variety of solutions have been proposed, consequently. Algorithmic approaches can be roughly divided according to two main criteria: The kind of model that is used for representing the environment and the algorithm that is employed in order to estimate the model state. Furthermore, the model and method are often selected depending on the sensor that is used and the features of the environment that can be perceived by that sensor.

For the map representation often vectors of point-like feature positions [2] are used. These features correspond to landmarks or salient points/regions in the environment which can be determined from the sensor input by adequate feature-extraction algorithms. The attractiveness of feature/landmark-based representations for SLAM lies in their compactness. However, they rely on *a priori* knowledge about the structure of the environment to identify and distinguish potential landmarks. In the field of visual landmark-based SLAM algorithms, Lowe's SIFT approach [3] is commonly used. In recent publications, SURF features are proposed instead of SIFT as a better compromise between efficiency and accuracy [4]. Other successful approaches [5] utilize different interest operators for detecting salient and recognizable image regions, e.g. those introduced by Shi and Tomasi [7]. Landmarks have also been extracted from laser range scanner data, e.g. detecting tree trunks from local minima in the depth profile [8].

In contrast to landmark representations, gridmaps [9] do not make assumptions about any specific features to be observable in the environment. They can represent arbitrary environment structures with nearly unlimited detail. However, they require a large amount of memory, in particular, the memory cost grows with the desired level of detail.

The estimation algorithms can be roughly distinguished in two main classes: Kalman filters and derivations thereof, and (Rao-Blackwellized) particle filters. The Kalman filter and its non-linear derivation Extended Kalman filter (EKF) as well as similar approaches like the Information filter [10] are best suited for landmark representations. They are able to estimate the full posterior distribution over the pose and map state. However they assume Gaussian distributions in the motion and observation model and become unstable if these assumptions are not met. Furthermore, updateing the full covariance matrix can become very expensive for large environments. Therefore, algorithmic extensions have been proposed for handling large numbers of state variables [6].

An effective means of handling the high-dimensionality in the SLAM problem has been introduced in the form of the Rao-Blackwellized Particle Filter (RBPF) [11]: in this approach the state space is partitioned into the pose and map state. A particle filter approximates the pose belief distribution of the robot, while each particle contains a map which represents the model of the environment, assuming the pose estimation of that specific particle (and its history, which is the estimation of the entire robot path) to be correct. The RBPF approach to SLAM has been successfully used with landmark-based maps [8] as well as with gridmaps [12]. However, gridmap solutions mostly used laser scanners and exploited the high resolution and accuracy in order to reduce state uncertainty and therefore computational cost [13].

In contrast to most of the approaches that have been proposed so far, we aim at developing a SLAM algorithm that is not adapted to any specific feature definitions or

This work has received funding from the State Thuringia (TAB-Grant #2006-FE-0154) and the AiF/BMWI (ALRob-Project Grant #KF0555101DF).

C. Schroeter, and H.-M. Gross are with Neuroinformatics and Cognitive Robotics Lab, Ilmenau University of Technology, 98684 Ilmenau, Germany. christof.schroeter@tu-ilmenau.de.

sensor characteristics. Instead, a widely applicable model and algorithm should be used to represent the robot's observations and to build the global environment map. For this purpose, a gridmap model is not only used for the global map, but also for representation of the current observation(s): In addition to the global map, a local gridmap is built from the current observation or from a sequence of recent observations. While in other implementations the evaluation of state hypotheses (particles) is based on the compliance of observations with the global map through a sensor model, which is specific to a particular sensor, in our approach it is calculated using the compliance of local and global map. With this method, any sensor (even multiple sensors) can easily be integrated in the SLAM framework, as long as a gridmap can be built from the sensor observations. Furthermore, because the local map can incorporate multiple subsequent observations, it is particularly well suited for sensors with low spatial resolution or high noise, where a sequence of measurements (preferably from slightly different positions) will yield significantly richer information than a single measurement.

In [14] we presented the Map Match SLAM approach and showed its successful application to SLAM with low resolution/high uncertainty sonar range sensors. We also proposed a specific map model as a solution to the high memory cost of gridmaps in combination with large particle numbers, as an alternative to the one proposed in [13]. We will give a short overview over the SLAM framework that was used there in section III and will describe the application to binocular (stereo) vision input in section IV and to monocular vision input in section V. Experimental results for both inputs are presented in section VI.

II. RAO-BLACKWELLIZED PARTICLE FILTER

The complexity of the SLAM problem arises from the very high-dimensional state space, consisting of the variables describing the robot pose and the variables describing the environment state. In the case of gridmaps, the map alone usually contains a few thousands up to several million cells, each of which corresponding to a state variable. Obviously, a full posterior over the state is extremely costly to estimate. The idea of the Rao-Blackwellized Particle Filter (RBPF) in application to SLAM is to use a particle filter to estimate the robot trajectory distribution $p(x_{1:t}|z_{1:t}, u_{0:t})$ given the sequence of odometry measurements $u_{0:t}$ and environment observations $z_{1:t}$. This trajectory estimate is then used to estimate the desired distribution over map and trajectory:

$$p(x_{1:t}, m|z_{1:t}, u_{1:t}) = p(m|x_{1:t}, z_{1:t})p(x_{1:t}|z_{1:t}, u_{0:t}) \quad (1)$$

The particle filter works in analogy to Monte-Carlo-Localization [15], except that instead of one given map, each particle contains a separate map. In order to calculate the importance weights for $p(x_{1:t})$, each particle uses its own map. The map, in return, is built from the estimated trajectory of that corresponding particle. The effect is that a number of hypothesis maps are built, each corresponding

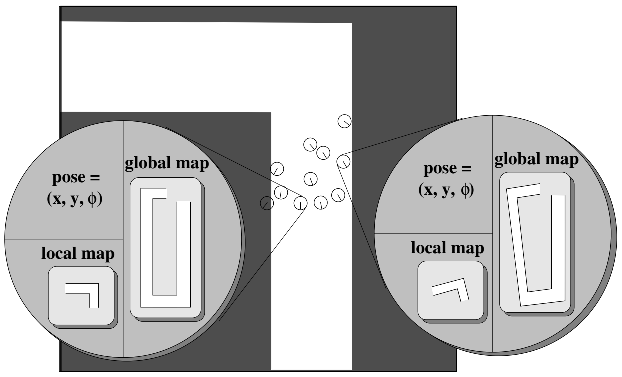


Fig. 1. Data representation overview: The particles model the distribution of the robot pose belief. Each particle contains a full map of the environment, which is a combination of the particle trajectory and the sensor observations. Furthermore, each particle contains a local map, which only contains the most recent measurements and depends on the particle's current pose belief. The situation shown is shortly before a loop closing. Apparently, the left particle is a better approximation of the true pose than the right particle.

to a possible trajectory. Importance weighting is performed with the weight for particle i following

$$w^{(i)} \simeq \frac{p(x_t^{(i)}|z_{1:t}, u_{0:t})}{\pi(x_t^{(i)}|z_{1:t}, u_{0:t})} \quad (2)$$

Here, $\pi(x_t^{(i)})$ denotes the proposal distribution. Typically, the motion model is used to generate the proposal distribution from the last particle generation (again, in analogy to localization), in which case the weight formula simplifies to

$$w^i \simeq p(z_t|x_t^{(i)}, m^{(i)}) \quad (3)$$

By repeatedly calculating importance weights, followed by resampling to adapt the particle distribution to the estimated distribution, particles are preferred which contain maps that match new observations best, therefore the most likely map is selected.

III. MAP MATCH SLAM

The base of our SLAM approach is a particle filter, where each particle contains a pose (x,y, heading ϕ) estimate as well as a map estimate. Without loss of generality, we can assume the robot to start mapping at position (0,0,0). However, it is also possible to initialize the particle filter at any other pose or even with any pose distribution, e.g. to align with a previously learned part of the map. While the robot moves, the particles move as well, according to the odometry readings and the probabilistic odometry motion model, which describes the uncertainty in the actual robot motion. Due to this uncertainty, the motion model contains a stochastic component, which results in the particles spreading out and generating slightly different trajectories. Basically, we use the standard motion model which also can be found in [12]. Additionally, during motion the robot observes the environment by means of an appropriate sensor. A map update is triggered frequently (depending on sensor range and measurement characteristics). In that map update, each particle adds the new environment observation to its own map, at its own estimated current position. Since the position estimates of the particles are slightly different, the maps differ as well (Fig. 1). As a distinctive feature of Map Match SLAM, each particle contains, in addition to the global map, a local map. The local map is built from sensor observations in the same way as the global map, but consists of a limited number of the most recent observations only, at any time.

Furthermore, by delaying updates for the global map until the robot has moved on for several meters, we can ensure that local and global map consist of disjoint sets of observations. For this purpose, observations are stored in a queue together with respective particle positions and global map updates are performed with queue elements after the robot has moved away from the actual observation position. This enables the use of map matching between local and global map for particle weighting, while still making best use of the available information by incorporating all observations in the global map.

Because of the delayed updates, from the start of mapping, as long as the robot moves forward, the global map at the current position is unknown, and there is no overlap between local and global map. Only when the robot returns to a known area (more precisely, when a particle believes to return to a known area), local and global map overlap at the estimated position and can be compared (Fig. 2). For a correct particle position estimate, local and global map of the particle should be compliant with each other (assuming a static environment), while for wrong position estimates, there will be discrepancies between the maps. Therefore, the matching of local and global map is an appropriate measure for evaluating the correctness of the particle's position estimate.

An alternative to using one local per particle would be to build one common local map (relative to a robot centered reference frame) and transform that local map into world coordinates for the matching, according to each respective particle's position. Unfortunately, the discrete nature of grid maps causes aliasing effects under translational and rotational transformations, therefore this solution would come at the cost of reduced pose resolution.

The calculation of the match value between the local and global map is simple: For each cell in the local map the occupancy value of the corresponding cell in the global map is tested. If the cell is occupied in both maps, it contributes with a value of $+1$. If the cell values of local and global map disagree, the cell contributes with a value of -1. That way, the match value is positive if local and global map are very similar, and it is negative if many objects exist in the local map where there is free space in the global map. Discrimination between free, occupied and unknown cells is done based on thresholds, but there is little sensitivity of the results with respect to the actual threshold values, because cell values mostly converge towards $0/1$ after few observations and the map itself is nearly binary. Therefore, the following, simple binary decision is applied for calculating the match value:

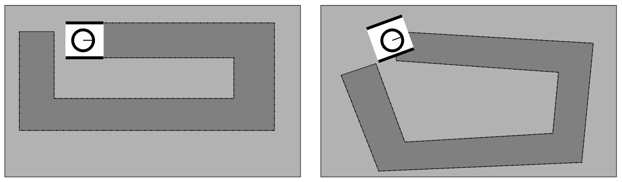


Fig. 2. Map matching: For the left particle, representing a correct position belief, the local map (white) is aligned to the global map (dark gray) very well, while for the right particle, which does not represent a position belief consistent with the environment, the local map conflicts with the global map (many of the occupied wall cells in the local map correspond to free cells in the global map). This situation would result in a higher weight for the left particle, supporting the position hypothesis which generates a consistent map.

$$match^{(i)} = \sum_{cell \in map} \begin{cases} 1 & if \quad (local(cell) = occ) \wedge \\ & \quad (global(cell) = occ) \\ -1 & if \quad (local(cell) = occ) \wedge \\ & \quad (global(cell) = free) \\ -1 & if \quad (local(cell) = free) \wedge \\ & \quad (global(cell) = occ) \\ 0 & \quad otherwise \end{cases} \tag{4}$$

Note that there is no rating for matching free cells. Because the number of free cells is typically dominating occupied cells, a "reward" for matching free cells would often favor a maximum *area overlap* of local and global map, disregarding the accurate match of occupied cells. In the experiments, in many cases this would prevent a correct loop closure, as those particles that approached a loop closure position earlier would be weighted higher, regardless of correct position estimate. Furthermore, unknown cells do not contribute to the match value (eq. 4), because an unknown cell does not allow any assumption about the "correct" cell value - it is not even justified to assume unknown cells in the local map must correspond to unknown cells in the global map, as a slightly different robot path may yield observations of a cell that was occluded previously (or vice versa).

To obtain the actual particle weight $w^{(i)}$, an exponential function is applied as follows:

$$w^{(i)} = e^{c \cdot match^{(i)}} \tag{5}$$

with c being a free parameter to influence the range of the particle weights and, therefore, the speed of convergence.

IV. GRIDMAPS FROM DISPARITY IMAGES

In our previous work [14], we introduced Map Match SLAM for sonar range sensors. There, we found that map matching works better for particle weighting in RBPF SLAM than single observations when working with sonar sensors, which have a low resolution and high characteristic sensor noise. In order to prove the easy adaptation to various sensors (even with a completely different measurement principle), we will show here how to use Map Match SLAM for building gridmaps from disparity images that are provided by a stereo camera. Only minor modifications are necessary for using a

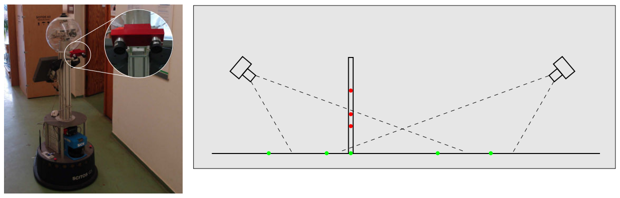


Fig. 3. Left: The stereo camera mounted on our experimental platform SCITOS-G5 from MetraLabs GmbH, Ilmenau. Right: An obstacle seen from two different observation positions: Due to the camera's limited field of view, only the lower part of the obstacle is visible if it is too far away from the robot. Consequently, that position will be wrongly estimated as free space from that observation (see text).

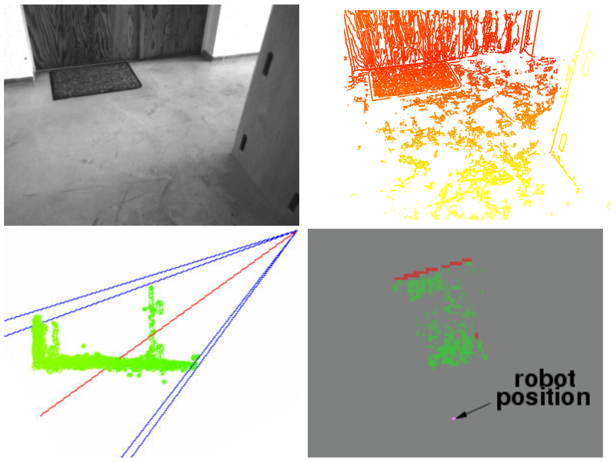


Fig. 4. Occupancy estimation using a stereo camera: One input image of the stereo image pair (top left) and the corresponding disparity image (top right, yellow = low, orange = medium, red = high disparity). The disparity image is not dense because disparity can only be calculated for image regions with distinctive structure. The bottom images show the 3D point cloud generated from the disparity image (left, side view into the scene, + camera viewing frustum) and the respective cell occupancy estimations (right, with red = occupied cells, green = free cells) for the visible area: the opposite door and wall as well as the edge of the box to the right are correctly seen as obstacles, while the floor is free space (incidentally, this is a view out of the door that is visible in the background in Fig. 3)

new sensor: In fact, only the map update needs to be modified to be able to update the local as well as the global map from the sensor data. All other parts of the particle filter (motion update, particle weighting through map matching, resampling etc.) remain unchanged.

For image aquisition, we use a Videre Design Stereo-On-Chip (STOC) camera (Fig. 3 left), which calculates disparity images from two calibrated camera images in hardware (Fig. 4). Furthermore, the camera library also provides methods to transform the pixels into a 3D depth point cloud in the camera coordinate system. This is possible because view direction is correctly known for each pixel from the camera's intrinsic parameters (determined as part of stereo calibration), and the depth is known through disparity.

In order to convert the 3D points into robot coordinates (to get the relative position as well as the height above ground), we need to know the camera mount parameters height and pitch. This is estimated in a simple calibration run where the camera must see (nearly) only the floor in front of the robot, preferably with good surface structure. With the assumption that all points in the generated point cloud are on the ground plane, the best fitting height and pitch parameters are calculated using a RANSAC algorithm [16].

With known camera mount parameters, points can be transformed from the camera coordinate system into the robot coordinate system. With the robot position estimated by a particular particle, a further transformation yields a global coordinate system position for each point.

The occupancy estimate is done in the following way: first, the world coordinate points calculated from the disparity images are assigned to the respective cells of the occupancy gridmap. Every point with height above an obstacle threshold $h > h_{obst}$ indicates that the cell is occupied. A point below a free space threshold $h < h_{free}$ indicates a free cell. Here, we do not make any implicit assumptions about free space, (like, e.g. space between the robot and the next obstacle is free), because they do not hold if an obstacle is not perceived, e.g. due to insuffient surface structure. Instead, cells are only considered to be free in the current observation if the floor is actually visible (and disparity for it can be determined).

Note that there may be points of different height within one cell. In order to eliminate outliers (erroneous obstacle detections), a cell is estimated as occupied only if a certain number of points (e.g. 5) above the obstacle threshold are located within the cell. If there are points indicating free space, but no obstacle points, the cell is considered to be free. These estimates are then used to update the stored occupancy for all cells $P(occ_{ij})$, using standard Bayesian occupancy update [9]:

$$P(occ_{ij}|o_{1:n}) = 1.0 - \frac{1.0}{1.0 + \frac{P(occ_{ij}|o_n)}{1-P(occ_{ij}|o_n)} \cdot \frac{P(occ_{ij}|o_{1:n-1})}{1-P(occ_{ij}|o_{n-1})}} \tag{6}$$

where the intial occupancy is 0.5. If a cell holds no or not a sufficient number of depth points in the current observation, it remains unchanged.

One specific exception arises from the limited field of view of the camera: Since the camera is mounted in such a way that it is looking towards the floor in front of the robot, the visible height decreases with the distance from the robot (Fig. 3 right). Due to that limitation, obstacles become partly or fully invisible if they are too far away from the robot. When the lower part is still visible but not the upper part, the respective position will wrongly be estimated to be free from that observation. Through the cell update, this could lead to fluctuation of the cell occupancy estimates or even removal of previously correctly perceived obstacles from the map. Note that although the viewing frustum is known, the 2D occupancy grid map does not store the height of obstacles and hence it is impossible to decide whether the object that occupied the cell in an earlier observation has disappeared or is just out of view. Eventually, this limitation could only be overcome by using a representation which preserves the distribution of obstacles also in the height dimension, e.g. similar to [18]. Our intention here is to show that map matching using a simple 2D model will yield good results in many scenarios. To this purpose, we resort to a simple heuristic method: occupancy updates with "free" observations are omitted if a cell has an occupancy above a certain threshold, i.e. obstacles have been seen in that cell in a certain number of observations before. This way, an obstacle will remain in the map even if it is out of the camera's field of view in later observations. There is a drawback of course: This method is not suited for environments with dynamic objects, as an obstacle that was seen in a certain place will remain in the map - even if it disappears, and the position is observed to be free later.

V. GRIDMAPS FROM MONOCULAR CAMERA IMAGES

In order to further prove the wide range capabilities of the Map Match SLAM approach, we also demonstrate SLAM from monocular camera input. Again, only the map update needs to be adapted. Since the hardware does not provide depth information in that case, we need an extended pre-processing. To reconstruct 3-dimensional information from the camera images, a stereo-from-motion approach was used [17]. The basic components of the depth reconstruction algorithm are:

1) Detection of interest points in the camera images applying the FAST feature detector [19].

2) Tracking of features over a sequence of images through optimal linking of the detected points in consecutive frames. The robot's odometry is used for finding epipolar lines and initial 3D position estimation. Furthermore, image window correlation is considered in the linking process.

3) Utilizing a multi-baseline stereo approach for initial depth estimation of tracked points: a tracked feature's 3D position is calculated from the feature's image positions in a sequence of frames. Odometry is used as an estimate of the respective camera observation positions. Since errors in the odometry are small enough over a short distance (features are usually tracked for not more than a few meters), it is sufficient for that purpose.

4) Kalman filtering for refinement of the 3D position: With the initial estimate, one Kalman filter is initialized for each feature. Tracking of features continues until they are lost (usually because they leave the camera's field of view). Each new observation contributes to the position estimate by a Kalman filter observation update. Since feature positions are estimated in global coordinates, a motion update is not required.

In contrast to a binocular stereo camera setup, in monocular stereo-from-motion the observation positions are not calibrated and scanlines for detection of matching points are not exactly known. Therefore, uniqueness/recognizability requirements for feature points are much higher. Besides the computation cost, this is the main reason that the resulting depth information is very sparse compared to a stereo camera (Fig. 5). This also affects the occupancy information that can

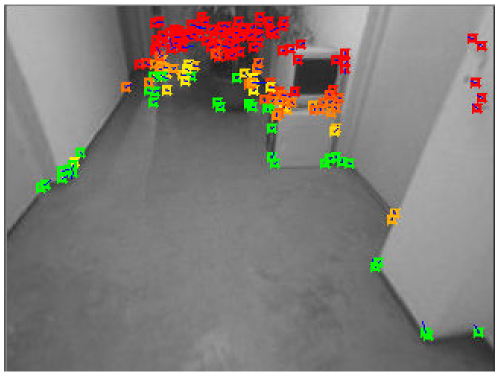


Fig. 5. Stereo-from-motion: 3D points estimated from the monocular image sequence (only the most recent image shown here). The feature's positions in the current image are marked, the color illustrates the estimated height over ground (green = low, orange = medium, red = high).

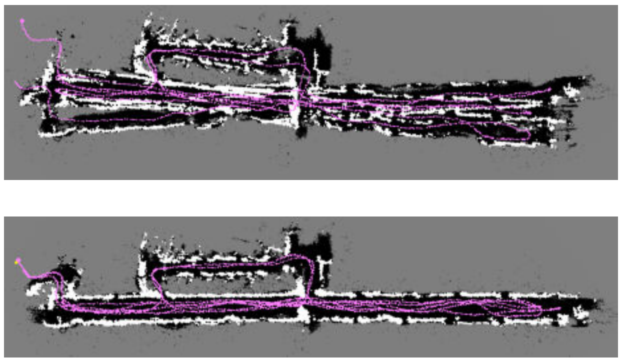


Fig. 6. SLAM using a stereo camera: The top image shows the path and the resulting occupancy grid map according to the robot's odometry. The length of the hallway is about 40m and the entire pathlength in this experiment is 300m. Although odometry has a relatively good accuracy on the SCITOS robot in comparison to other platforms, the position errors accumulate up to a few meters and in effect the map is very inconsistent. In the bottom image, the corrected path and map from our Map Match SLAM approach with a visual depth sensor is shown. All the position errors are corrected and the obstacles and free space in the map are clearly defined, despite the limitations of the sensor and its field of view.

be extracted. Due to the sparse depth information, detection of free space is not feasible. The map building is restricted to estimation of occupied cells from detected feature points with a height greater than an obstacle threshold (Fig. 8). For the map matching, eq. 4 is used again, but since no free cells are detected, only matching occupied cells contribute to the sum value. Of course, for path planning and obstacle avoidance this kind of map can only serve as an additional cue and needs to be combined with information from other sensors like laser or sonar. For e.g. self-localization, though, it is directly applicable. The approach is mainly presented here to show that Map Match SLAM is capable to estimate the correct path even with such sparse sensor information, although we are aware of the fact that occupancy grids may not be the best suited spatial model for this kind of sensor.

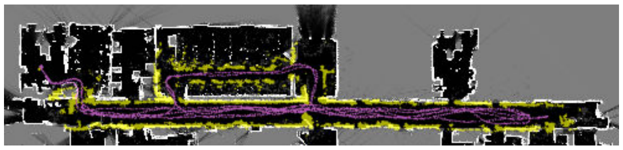


Fig. 7. Overlay of the stereo SLAM map (colored yellow for better visibility) with a reference laser map. The most noticeable differences are caused by objects which are only partially or not at all seen by the laser scanner (as the observations are restricted to one horizontal plane), e.g. tables, chairs, plants.

VI. EXPERIMENTAL RESULTS

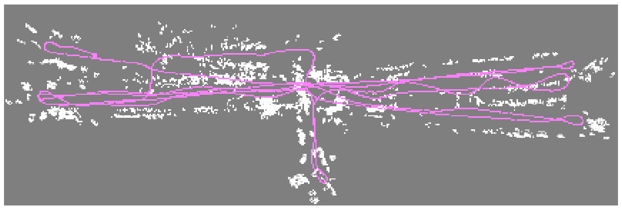
First, for the stereo mapping a proof-of-concept experiment is presented, in which a partial map of our institute's building was built (Fig. 6). The cellsize was chosen as $(0.1m)^2$, which yields a good resolution but also ensures a sufficient number of pixels/depth points per cell to avoid excessive noise in the occupancy estimation from disparity outliers. The robot was manually driven several hundred meters through the central hallway and some adjacent rooms, and SLAM was performed using only the odometry and disparity images as inputs, with a fixed number of 200 particles.

A similar experiment was done with monocular input (Fig. 8). Due to the sparse depth information, relatively few cells in the map are adapted at all. As explained in section V, free space is not detected. Despite the sparse map, the consistent reconstruction of the true path also succeeds with the combination of monocular images and odometry only.

VII. CONCLUSIONS

In extension to our previous work on Map Match SLAM, in this paper we demonstrated a novel way of Simultaneous Localization And Mapping based on visual input. We presented methods to estimate gridmap occupancies from stereo disparity images as well as from monocular camera images and integrated those map building approaches into our Map Match SLAM framework, enabling the use of camera images in SLAM without any further modeling requirements. Experimental demonstration of indoor mapping confirms the functional capability of the proposed visual SLAM approach.

The demonstration of SLAM using such different sensors as a sonar range array, a stereo camera, or a single camera within the same computational framework supports the wide range applicability of our Map Match SLAM system, which is not dependent on specific sensor characteristics or predefined features of the environment.



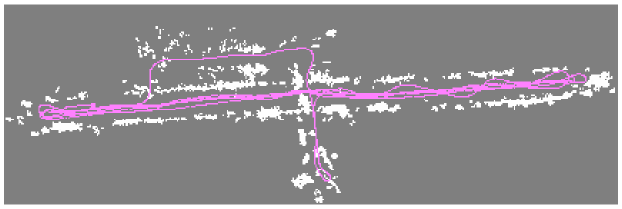


Fig. 8. SLAM with a single monocular camera: The top image shows the robot path and resulting occupancy gridmap from the odometry. The area is the same as in Fig. 6. The bottom image shows the result of Map Match SLAM with the single camera. Although the map is very sparse, the path estimation is correct.

REFERENCES

[1] S. Thrun, W. Burgard, D. Fox, *Probabilistic Robotics*, The MIT Press, 2005

[2] R. Smith, M. Self, P. Cheeseman, "A stochastic map for uncertain spatial relationships", in *4th Intl. Symposium on Robotics Research*, pp. 467-474, 1988

[3] S. Se, D. Lowe, J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks", in *International Journal of Robotics Research*, 21(8), pp. 735-758, 2002

[4] Ch. Valgren, A. Lilienthal, "SIFT, SURF and Seasons: Long-term Outdoor Localization Using Local Features", in *Proc. 3rd European Conf. on Mobile Robots*, pp. 253-258, 2007

[5] A. Davison, "Real-time simultaneous localisation and mapping with a single camera", in *Proc. Intl. Conf. on Computer Vision (ICCV'03)*, pp. 1403-1410, 2003

[6] J. Leonard, P. Newman, "Consistent, Convergent, and Constant-Time SLAM", in *Proc. Intl. Joint Conf. on Artificial Intelligence*, pp. 1143-1150, 2003

[7] J. Shi, C. Tomasi, "Good features to track", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 593-600, 1994

[8] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem", in *Proc. AAAI-2002.*, pp. 593-598

[9] Moravec, H., "Sensor Fusion in Certainty Grids for Mobile Robots", in *AI Magazine*, 9(2), pp. 61-77, 1988

[10] S. Thrun, D. Koller, Z. Ghahramani, H. Durrant-Whyte, A.Y. Ng, "Simultaneous mapping and localization with sparse extended information filters", in *Intl. Journal Robotics Research*, 23(7-8), pp. 693-716, 2004

[11] K. P. Murphy, "Bayesian map learning in dynamic environments", in *Proc. NIPS'99*, pp. 1015-1021, 1999

[12] D. Hähnel, W. Burgard, D. Fox, S. Thrun, "An Efficient FastSLAM Algorithm for Generating Maps of Large-Scale Cyclic Environments from Raw Laser Range Measurements", in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS'03)*, pp. 206-211, 2003

[13] A. I. Eliazar, R. Parr, "DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks", in *Proc. Intl. Joint Conf. on Artificial Intelligence (IJCAI'03)*, pp. 1135-1141, 2003

[14] C. Schröter, H.-J. Böhme, H.-M. Gross, "Memory-Efficient Gridmaps in Rao-Blackwellized Particle Filters for SLAM using Sonar Range Sensors", in *Proc. 3rd European Conference on Mobile Robots*, pp. 138-143, 2007

[15] D. Fox, W. Burgard, F. Dellaert, S. Thrun, "Monte Carlo Localization: Efficient Position Estimation for Mobile Robots", in *Proc. AAAI Natl. Conf. on Artifical Intelligence*, pp. 5398-5403, 1999

[16] M.A. Fischler and R.C. Bolles, "Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography", in *Communications of the ACM*, 24(6), pp. 381-395, 1981

[17] E. Einhorn, C. Schröter, H.-J. Böhme, H.-M. Gross, "A Hybrid Kalman Filter Based Algorithm for Real-time Visual Obstacle Detection", in *Proc. 3rd European Conference on Mobile Robots*, pp. 156-161, 2007

[18] Tim K. Marks, Andrew Howard, Max Bajracharya, Garrison W. Cottrell, and Larry Matthies, "Gamma-SLAM: Using Stereo Vision and Variance Grid Maps for SLAM in Unstructured Environments", in *Proc. IEEE Intl. Conf. on Robotics and Automation*, pp. 3717-3724, 2008

[19] E. Rosten, T. Drummond, "Machine learning for high-speed corner detection", in *Proc. European Conf. on Computer Vision*, pp. 430 - 443, 2006.