Map Building with Mobile Robots in Populated Environments

Dirk Hähnel[†] Dirk Schulz[‡] Wolfram Burgard[†]

[†]*University of Freiburg, Department of Computer Science, Germany*
[‡]*University of Bonn, Department of Computer Science, Germany*

Abstract

The problem of generating maps with mobile robots has received considerable attention over the past years. However, most of the approaches assume that the environment is static during the data-acquisition phase. In this paper we consider the problem of creating maps with mobile robots in populated environments. Our approach uses a probabilistic method to track multiple people and to incorporate the results of the tracking technique into the mapping process. The resulting maps are more accurate since corrupted readings are treated accordingly during the matching phase and since the number of spurious objects in the resulting maps is reduced. Our approach has been implemented and tested on real robot systems in indoor and outdoor scenarios. We present several experiments illustrating the capabilities of our approach to generate accurate 2d and 3d maps.

1 Introduction

Learning maps with mobile robots has received considerable attention over the last two decades. This is because maps often are inherently necessary for mobile robots to perform their tasks. When mapping an environment, a mobile robot generally has to cope with different kinds of noise: noise in the odometry and noise in the sensor data. Therefore, the map learning problem is a chicken-and-egg problem. If the pose (we use the term *pose* to refer to a robot's x-y location and its heading direction θ) of the robot was always known during mapping, building maps is relatively easy. On the other hand, if a map was available, determining the robot's poses can be done efficiently. In the literature, the mobile robot mapping problem is often referred to as the *simultaneous localization and mapping problem (SLAM)* [2, 4, 7].

Approaches to concurrent mapping and localization can roughly be classified according to the kind of sensor data processed and the matching algorithms used. For example, the approaches described in [12, 2, 4, 7] extract landmarks out of the data and match these landmarks to localize the robot in the map being learned. The other set of approaches such as [8, 6, 13] use raw sensor data and perform a dense matching of the scans. Although all approaches possess the ability to cope with a certain amount of noise in the sensor data, they assume that the environment is almost static during the mapping process. Especially in populated environments, additional noise is introduced to the sensor data which increases the risk of localization errors. Additionally, people in the vicinity of the robots appear as objects in the resulting maps and therefore make the maps not usable for path planning etc. Recently [15] presented a heuristic and feature-based approach to identify dynamic objects in range scans. The corresponding measurements are then filtered out during 2d scan registration.

In this paper we present a probabilistic approach to filtering people out of sensor data and techniques to incorporate the results of the filtering into the mapping process. Our approach has several desirable properties. First, by incorporating the results of the people tracker, the alignment of the scans becomes more robust. Additionally, the resulting maps are more accurate, since measurements corrupted by people walking by are filtered out. Compared to [15] our approach uses a tracking technique and therefore is able to predict the positions of the person's even in situations in which the corresponding features are temporarily missing. Empirical results, described in this paper, illustrate that our approach succeeds in learning accurate large-scale 2d and 3d maps of populated environments with range scanners even if several persons are in the vicinity of the robot.

This paper is organized as follows. In the next section we briefly present our approach to tracking multiple people in range scans. The third and forth section describes our mapping technique and how the results of the people tracker are integrated into the mapping process. The fifth section contains several experiments describing the advantages of our approach to learning 2d and 3d maps with range scanners.

2 Sample-based Joint Probabilistic Data Association Filters (SJPDAFs)

To detect people and track people in the vicinity of the robot, our system applies a sample-based variant of Probabilistic Data Association Filters (JPDAFs) [3]. Suppose

there are K persons and let $\mathbf{X}^t = \{\mathbf{x}_1^t, \ldots, \mathbf{x}_K^t\}$ be the states of these persons at time t. Note that each $\mathbf{x}_i^t$ is a random variable ranging over the state space of a single person. Furthermore, let $\mathbf{Z}(t) = \{\mathbf{z}_1(t), \ldots, \mathbf{z}_{m_t}(t)\}$ denote a feature set observed at time t, where $\mathbf{z}_j(t)$ is one feature of such a set. $\mathbf{Z}^t$ is the sequence of all feature sets up to time t. The key question when tracking multiple persons is how to assign the observed features to the individual objects.

In the JPDAF framework, a joint association event θ is a set of pairs $(j, i) \in \{0, \ldots, m_t\} \times \{1, \ldots, K\}$. Each θ uniquely determines which feature is assigned to which object. Please note, that in the JPDAF framework, the feature $\mathbf{z}_0(t)$ is used to model situations in which an object has not been detected, i.e. no feature has been found for object i. Let Θ_{ji} denote the set of all valid joint association events which assign feature j to the object i. At time t, the JPDAF considers the posterior probability that feature j is caused by object i:

$$\beta_{ji} \quad = \quad \sum_{\theta \in \Theta_{ji}} P(\theta \mid \mathbf{Z}^t). \tag{1}$$

According to [11], we can compute the β_{ji} as

$$\beta_{ji} \quad = \quad \sum_{\theta \in \Theta_{ji}} \alpha \, \gamma^{(m_t - |\theta|)} \prod_{(j,i) \in \theta} p(\mathbf{z}_j(t) \mid \mathbf{x}_i^t). \tag{2}$$

It remains to describe, how the beliefs $p(\mathbf{x}_i^t)$ about the states of the individual objects are represented and updated. In our approach [11], we use sample-based representations of the individual beliefs. The key idea underlying all particle filters is to represent the density $p(\mathbf{x}_i^t \mid \mathbf{Z}^t)$ by a set $\mathbf{S}_i^t$ of N weighted, random samples or *particles* $s_{i,n}^t (n = 1 \ldots N)$. A sample set constitutes a discrete approximation of a probability distribution. Each sample is a tuple $(x_{i,n}^t, w_{i,n}^t)$ consisting of state $x_{i,n}^t$ and an importance factor $w_{i,n}^t$. The *prediction* step is realized by drawing samples from the set computed in the previous iteration and by updating their state according to the prediction model $p(\mathbf{x}_i^t \mid \mathbf{x}_i^{t-1}, \delta t)$. In the *correction* step, a feature set $\mathbf{Z}(t)$ is integrated into the samples obtained in the prediction step. Thereby we consider the assignment probabilities β_{ji}. In the sample-based variant, these quantities are obtained by integrating over all samples:

$$p(\mathbf{z}_j(t) \mid \mathbf{x}_i^t) \quad = \quad \frac{1}{N} \sum_{n=1}^{N} p(\mathbf{z}_j(t) \mid x_{i,n}^t). \tag{3}$$

Given the assignment probabilities we now can compute the weights of the samples

$$w_{i,n}^t = \alpha \sum_{j=0}^{m_t} \beta_{ji} p(\mathbf{z}_j(t) \mid x_{i,n}^t), \tag{4}$$

where α is a normalizer ensuring that the weights sum up to one over all samples. Finally, we obtain N new samples from the current samples by bootstrap resampling.

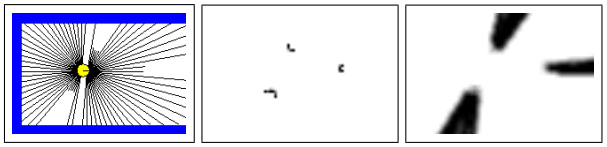

Figure 1: *Typical laser range finder scan. Two of the local minima are caused by people walking by the robot (left image). Features extracted from the scan, the grey-level represents the probability that a person's legs are at the position (center). Occlusion grid, the grey-level represents the probability that the position is occluded (right image).*

Figure 2: *From left to right, top-down: the occupancy map for the current scan, the occupancy map for the previous scan, the resulting difference map, and the fusion of the difference map with the feature maps for the scan depicted in Figure 1*

For this purpose we select every sample $x_{i,n}^t$ with probability $w_{i,n}^t$.

In our system we apply the SJPDAF to estimate the trajectories of persons in range scans. Since the laser range scanners mounted on our platforms are at a height of approx. 40 cm, the beams are reflected by the legs of the people which typically appear as local minima in the scans. These local minima are used as the features of the SJPDAF. See left and middle part of Figure 1. Unfortunately, there are other objects which produce patterns similar to people. To distinguish these static objects from moving people our system additionally considers the differences between occupancy probability grids built from consecutive scans. Static features are filtered out. This is illustrated in Figure 2.

Finally, we have to deal with possible occlusions. We therefore compute a so-called "occlusion map" containing for each position in the vicinity of the robot the probability that the corresponding position is not visible given the current range scan. See right part of Figure 1.

3 Computing Consistent Maps

Our current system is able to learn 2d and 3d maps using range scans recorded with a mobile robot. In both cases, the approach is incremental. Mathematically, we calculate a sequence of poses $\hat{l}_1, \hat{l}_2, \ldots$ and corresponding maps by maximizing the marginal likelihood of the

t-th pose and map relative to the $(t-1)$-th pose and map:

$$\hat{l}_t = \operatorname*{argmax}_{l_t}\{p(s_t \mid l_t, \hat{m}(\hat{l}^{t-1}, s^{t-1})) \cdot p(l_t \mid u_{t-1}, \hat{l}_{t-1})\} \quad (5)$$

In this equation the term $p(s_t \mid l_t, \hat{m}(\hat{l}^{t-1}, s^{t-1}))$ is the probability of the most recent measurement s_t given the pose l_t and the map $\hat{m}(\hat{l}^{t-1}, s^{t-1})$ constructed so far. The term $p(l_t \mid u_{t-1}, \hat{l}_{t-1})$ represents the probability that the robot is at location l_t given the robot was previously at position $\hat{l}_{t-1}$ and has carried out (or measured) the motion u_{t-1}. The resulting pose $\hat{l}_t$ is then used to generate a new map $\hat{m}$ via the standard incremental map-updating function presented in [9]:

$$\hat{m}(\hat{l}^t, s^t) = \operatorname*{argmax}_m p(m \mid \hat{l}^t, s^t) \quad (6)$$

One disadvantage of the approach described above lies in the fact, that the complexity of a single maximization step is in $O(t)$, since every measurement is compared with all previous measurements. In our system, we therefore use a map

$$\hat{m}(\hat{l}^{t-1,\Delta t}, s^{t-1,\Delta t}) = \hat{m}(\hat{l}_{t-1}, \ldots, \hat{l}_{t-\Delta t}, s_{t-1}, \ldots, s_{t-\Delta t}) \quad (7)$$

that is constructed based of the Δt most recent measurements only. This is motivated by two observations. First, proximity sensors have only a limited range so that the system generally cannot cover the whole environment with a single scan. Additionally, objects in the environment lead to occlusions so that many aspects of a given area are invisible from other positions. Therefore, measurements obtained at distant places often provide no information to maximize (5).

The overall approach can be summarized as follows: At any point $t-1$ in time the robot is given an estimate of its pose $\hat{l}_{t-1}$ and a map $\hat{m}(\hat{l}^{t-1,\Delta t}, s^{t-1,\Delta t})$. After the robot moved further on and after taking a new measurement s_t, the robot determines the most likely new pose $\hat{l}_t$. It does this by trading off the consistency of the measurement with the map (first term on the right-hand side in (5)) and the consistency of the new pose with the control action and the previous pose (second term on the right-hand side in (5)). The map is then extended by the new measurement s_t, using the pose $\hat{l}_t$ as the pose at which this measurement was taken.

It remains to describe how we actually maximize Equation (5). Our system applies two different approaches depending on whether the underlying scans are 2d or 3d scans.

3.1 Two-dimensional Scan Alignment

Our algorithm to 2d scan matching is an extension of the approach presented in [14]. To align a scan relative to the

Figure 3: *Two-dimensional scan alignment. Map created out of the Δt most recent scans (left image), measurement s_t obtained at time t (center image) and resulting alignment (right image).*

Δt most recent scans, we first construct a local grid map $\hat{m}(\hat{l}^{t-1,\Delta t}, s^{t-1,\Delta t})$ out of the Δt most recent scans. Additionally to [14] we integrate over small Gaussian errors in the robot pose when computing the maps. This avoids that many cells remain unknown especially if Δt is small, increases the smoothness of the likelihood function to be optimized and thus results in better alignments. The left image of Figure 3 shows a typical map constructed out of 100 scans. The darker a location, the more likely it is that the corresponding place in the environment is covered by an obstacle. Please note that the map appears slightly blurred according to the integration over small pose errors. To maximize the likelihood of a scan with respect to this map, we apply a hill climbing strategy. A typical scan is shown in the center of Figure 3. The optimal alignment of this scan with respect to the map is shown in the right image of Figure 3. As can be seen from the figure, the alignment is quite accurate.

3.2 Aligning Three-dimensional Range Scans

Unfortunately, a three-dimensional variant of the maps used for the 2d scan alignment would consume too much memory in the case of three dimensions. Therefore this approach is not applicable to 3d scan alignment. Instead, we represent the 3d maps as triangle meshes constructed from the individual scans. We create a triangle for three neighboring scan points, if the maximum length of an edge does not exceed a certain threshold which depends on the length of the beams.

To compute the most likely position of a new 3d scan with respect to the current 3d model, we apply an approximative physical model of the range scanning process. Obviously, an ideal sensor would always measure the correct distance to the closest obstacle in the sensing direction. However, sensors and models generated out of range scanners are noisy. Therefore, our systems incorporates measurement noise and random noise to deal with errors typically found in 3d range scans. First, we generally have normally distributed measurement errors around the distance "expected" according to the current position of the scanner and the given model of the environment. Additionally, we observe randomly distributed measurements because of errors in the model and because of deviations in the angles between corresponding beams

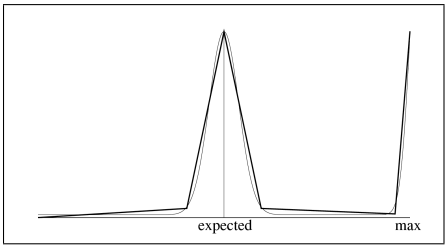

Figure 4: *The probabilistic measurement model given as a mixture of a Gaussian and a uniform distribution and its approximation by piecewise linear functions.*

in consecutive scans. Therefore, our model consists of a mixture of a Gaussian with a uniform distribution. The mode of the Gaussian corresponds to the expected distance given the current map. Additionally, we use a uniform distribution to deal with maximum range readings. To save computation time, we approximate the resulting distribution by a mixture of triangular distributions.

Whereas this approach saves memory, it requires more computation time than the technique for 2d scan alignment. However, in practical experiments we found out that this technique has two major advantages over the Iterative Closest Point (ICP) algorithm [1, 5] and other scan-matching techniques which are crucial in the context of 3d mapping. First, it exploits the fact that each laser beam is a ray that does not go through surfaces and therefore does not require special heuristics for dealing with occlusions. Second, our approach also exploits the information provided by maximum range readings since beams going through surfaces in the map reduce the likelihood of an alignment.

To compute the likelihood of a beam b given the current map $\hat{m}(\hat{l}^{t-1,\Delta t}, s^{t-1,\Delta t})$, we first determine the expected distance $e(b, \hat{m}(\hat{l}^{t-1,\Delta t}, s^{t-1,\Delta t}))$ to the closest obstacle in the measurement direction. This is efficiently carried out using ray-tracing techniques based on a spatial tiling and indexing [10] of the current map.

Then we compute the likelihood of the measured distance given the expected distance, i.e. we determine the quantity $p(b \mid e(b, \hat{m}(\hat{l}^{t-1,\Delta t}, s^{t-1,\Delta t})))$ using the mixture computed for $e(b, \hat{m}(\hat{l}^{t-1,\Delta t}, s^{t-1,\Delta t}))$. Assuming that the beams contained in s_t are independent, we compute the likelihood of the whole scan as

$$p(s_t \mid l_t, \hat{m}(\hat{l}^{t-1,\Delta t}, s^{t-1,\Delta t})) = \prod_{b \in s_t} p(b \mid e(b, \hat{m}(\hat{l}^{t-1,\Delta t}, s^{t-1,\Delta t}))). \quad (8)$$

To maximize Equation 5 we again apply a hill climbing technique.

4 Integrating People Tracking Results into the Map Building Process

The goal of integrating the results of the people tracker into a mapping process can be divided in two subjects:

1. to improve the alignment between the scans and
2. to filter out corrupted measurements originating from people walking in the vicinity of the robot.

To consider the estimated states of the persons during the scan alignment, we need to know the probability $P(\text{hit}_{x,y}|\mathbf{X}^t)$ that a beam ending at position $\langle x, y\rangle$ is reflected by a person. In our current implementation, we consider the individual persons independently:

$$P(\text{hit}_{x,y}|\mathbf{X}^t) = 1 - \prod_{i=1}^{K} \left(1 - P(\text{hit}_{x,y}|\mathbf{x}_i^t)\right) \quad (9)$$

where $P(\text{hit}_{x,y}|\mathbf{x}_i^t)$ is the likelihood that a beam ending at position $\langle x, y\rangle$ is reflected by a person, given the state $\mathbf{x}_i^t$. To compute this quantity, we construct a two-dimensional histogram by counting how many samples of $\mathbf{S}_i^t$ representing the belief of $\mathbf{x}_i^t$ fall into each bin. We then normalize the histogram and compute the probability $P(\text{hit}_b|\mathbf{x}_i^t)$ that a beam b is reflected by a particular person $\mathbf{x}_i^t$ as the probability contained in the histogram bin $\langle x_b, y_b\rangle$ in which the beam ends. Accordingly, we have

$$P(\text{hit}_b|\mathbf{X}^t) = P(\text{hit}_{x_b,y_b}|\mathbf{X}^t). \quad (10)$$

During the scan alignment we then weight each beam by the probability $1 - P(\text{hit}_b|\mathbf{X}^t)$.

The second task is to filter out beams reflected by persons to avoid spurious objects in the resulting maps. In our current system we compute a bounding box for each sample set $\mathbf{S}_i^t$ and integrate only those beams whose endpoint does not lie in any of the bounding boxes. To cope with the possible time delay of the trackers, we also ignore corresponding beams of several previous and next scans before and after the person was detected. Please note, that one generally can be more conservative during the map generation process, because the robot generally scans every part of the environment quite often. However, during scan alignment, a too conservative strategy results in too few remaining beams which reduces the accuracy of the estimated positions.

5 Experiments

The approach described above has been implemented and tested on different robotic platforms and based on extensive off-line experiments carried out with recorded data. The goal of the experiments described in this section is to illustrate that the integration of people detection techniques into the mapping process leads to better maps since the resulting alignments are more accurate and since beams reflected by persons are filtered out

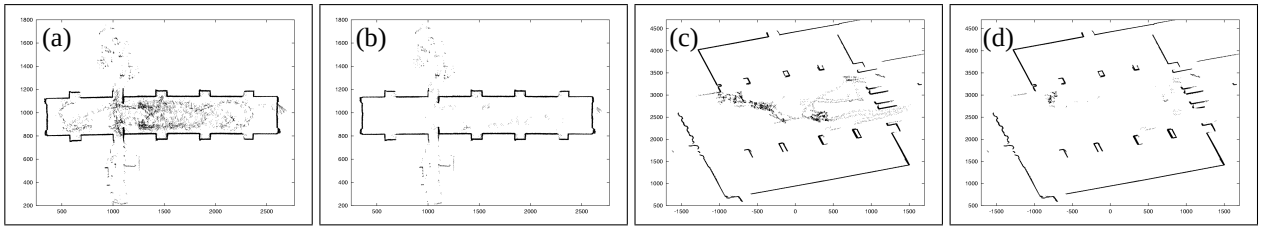

Figure 5: *Maps created without (a and c) and with (b and d) people filtering.*

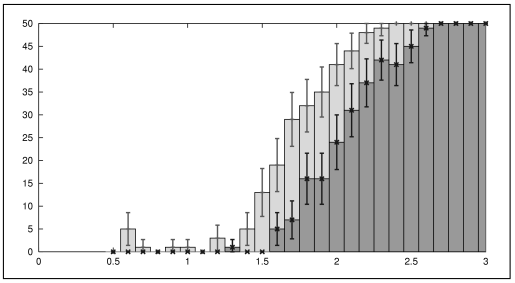

Figure 6: *Number of maps with translational error larger than 200cm computed without people filtering (light grey) and using people filtering (dark grey) for different additional noises.*

which reduces the number of spurious objects. Our implementation can detect people in real-time, so that the time to map an environment is not influenced by using this information.

5.1 Learning 2d Maps

The first experiments were carried out using an RWI B21 and a Pioneer I robot in a 25m x 4m large corridor environment and in the 30m x 45m large hallway of a museum. In both experiments several people were walking through the environment leading to a huge amount of corrupted readings. Figures 5(a) and 5(c) depict the maps obtained by integration all the input data. As can be seen from the figures, there are a lot of beams that were reflected by the persons. The maps resulting from our system after filtering out people are illustrated in Figures 5 (b) and 5 (d). Apparently, the number of corrupted readings is reduced considerably. Please note that there were 15 people walking through the museum hallway while the robot was mapping it. Some of the people were forming an almost static crowd so that they could not be detected by our people tracker. Additionally, the columns in this hallway produce similar features as humans do. Nevertheless, our approach could seriously reduce the number of readings corrupted by people.

5.2 Improved Robustness

Besides the fact that the resulting maps are better, filtering people increases the robustness of the mapping process. To demonstrate this we have carried out a series of experiments in which we added random noise to the poses in the input data and compared the performance of our

Figure 7: *Pioneer 2 AT robot used for 3d mapping (left) and typical situation in which people walk through the scene during mapping (right).*

Figure 8: *Spurious objects caused by people.*

mapping strategy with and without people filtering. We performed 50 experiments for each noise level. Figure 6 shows the numbers of maps containing a translational error larger than 200cm for the different noise values. As can bee seen by this statistic, using the information of the people tracker significantly increases the accuracy of the position estimation during the mapping process.

5.3 Learning 3d Maps

The last experiment was carried out to analyze the performance of our system when learning three-dimensional maps. For this experiment we used the Pioneer 2 AT platform (see left image of Figure 7) equipped with two laser range-scanners. Whereas the first scanner, that is mounted in front of the robot, is used for tracking people, the second scanner, that is mounted on an AMTEC wrist module, is used to scan the 3d structure of the environment. The right image of Figure 7 shows a typical scenario during this experiment performed on our university campus. Here, several people were walking through the scene while the robot was scanning it. Figure 9 (left) depicts the model obtained after aligning two scans of the same environment. In this model, the people appear as three-dimensional curves. Figure 8 contains a mag-

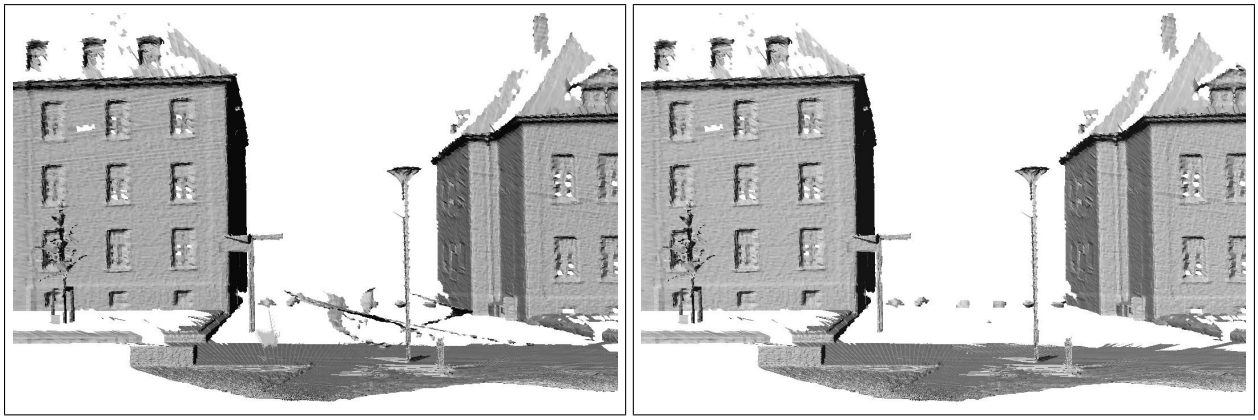

Figure 9: *Three-dimensional map of a building (left) and people filtered (right).*

nified view of the corresponding portion of the map. If we integrate the information obtained from the people tracker, however, these spurious objects are completely removed (see right image of Figure 9). The number of triangles in these models are 416.800 without filtering and 412.500 with filtering. Please note, that this experiment also illustrates the advantage of using a tracking system over a pure feature-based approach. Due to the displacement of the scanners, people are not always visible in both scanners. Accordingly, a purely feature-based approach like [15] will add objects to the 3d model whenever they are not detected by the first scanner. Our system, however, can predict positions of persons in the case of occlusions and thus can filter out the corresponding readings even if the features are missing.

6 Conclusions

In this paper we presented a probabilistic approach to mapping in populated environments. The key idea of this technique is to use a joint probabilistic data association filter to track people in the data obtained with the sensors of the robot. The results of the people tracking are integrated into the scan alignment process and into the map generation process. This leads to two different improvements. First, the resulting pose estimates are better and second, the resulting maps contain less spurious objects than the maps created without filtering people.

Our technique has been implemented and tested on different robotic platforms and for generating 2d and 3d maps. The experiments demonstrate that our approach can seriously reduce the number of beams corrupted by people walking through the environment. Additionally, extensive simulation experiments illustrate that the pose estimates are significantly better if the results of the tracking system are incorporated during the pose estimation.

Acknowledgements

This work has partly been supported by the EC under contract number IST-2000-29456.

References

[1] P. Besl and N. McKay. A method for registration of 3d shapes. *Trans. Patt. Anal. Mach. Intell. 14(2)*, pages 239–256, 1992.

[2] J.A. Castellanos, J.M.M. Montiel, J. Neira, and J.D. Tardós. The SPmap: A probabilistic framework for simultaneous localization and map building. *IEEE Transactions on Robotics and Automation*, 15(5):948–953, 1999.

[3] I.J. Cox. A review of statistical data association techniques for motion correspondence. *International Journal of Computer Vision*, 10(1):53–66, 1993.

[4] G. Dissanayake, H. Durrant-Whyte, and T. Bailey. A computationally efficient solution to the simultaneous localization and map building (SLAM) problem. In *ICRA'2000 Workshop on Mobile Robot Navigation and Mapping*, 2000.

[5] M.A. Greenspan and G. Godin. A nearest neighbor method for efficient ICP. In *Proc. of the 3rd Int. Conf. on 3-D Digital Imaging and Modeling (3DIM01)*, 2001.

[6] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proc. of the IEEE Int. Symp. on Computational Intelligence in Robotics and Automation (CIRA)*, 1999.

[7] J.J. Leonard and H.J.S. Feder. A computationally efficient method for large-scale concurrent mapping and localization. In *Proc. of the Ninth Int. Symp. on Robotics Research (ISRR)*, 1999.

[8] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.

[9] H.P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, pages 61–74, Summer 1988.

[10] H. Samet. *Applications of Spatial Data Structures*. Addison-Wesley Publishing Company, 1990.

[11] D. Schulz, W. Burgard, D. Fox, and A.B. Cremers. Tracking multiple moving objects with a mobile robot. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[12] H. Shatkay. *Learning Models for Robot Navigation*. PhD thesis, Computer Science Department, Brown University, Providence, RI, 1998.

[13] S. Thrun. A probabilistic online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research*, 20(5):335–363, 2001.

[14] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*, 2000.

[15] C.-C. Wang and C. Thorpe. Simultaneous localization and mapping with detection and tracking of moving objects. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*, 2002.