Operator-Assistive Mapping in Harsh Environments

Alexander Kleiner and C. Dornhege

Linköping University Post Print

N.B.: When citing this work, cite the original article.

Operator-Assistive Mapping in Harsh Environments

Alexander Kleiner
University of Freiburg
Georges-Köhler-Allee 52
79110 Freiburg, Germany
kleiner@informatik.uni-freiburg.de

Christian Dornhege
University of Freiburg
Georges-Köhler-Allee 52
79110 Freiburg, Germany
dornhege@informatik.uni-freiburg.de

Abstract — Teleoperation is a difficult task, particularly when controlling robots from an isolated operator station. In general, the operator has to solve "nearly blindly" the problems of mission planning, target identification, robot navigation, and robot control at the same time. The goal of the proposed system is to support teleoperated navigation with real-time mapping. We present a novel scan matching technique that reconsiders data associations during the search, enabling robust pose estimation even under varying roll and pitch angle of the robot enabling mapping on rough terrain.

The approach has been implemented as an embedded system and extensively tested on robot platforms designed for teleoperation in critical situations, such as bomb disposal. Furthermore, the system has been evaluated in a test maze by first responders during the Disaster City event in Texas 2008. Finally, experiments conducted within different environments show that the system yields comparably accurate maps in real-time when compared to higher sophisticated offline methods, such as Rao-Blackwellized SLAM.

Keywords: *SLAM, Mapping, HRI, Teleoperation, Operator Assistance*

I. INTRODUCTION

Teleoperation is a difficult task, particularly when controlling robots from an isolated operator station. This is particularly the case when the target area is hazardous to human beings and therefore robots can only be controlled from a safety zone. In general, the operator has to solve in an unknown environment the problems of mission planning, target identification, robot navigation, and robot control, at the same time. For untrained operators, control and target identification are already challenging on their own. The goal of the proposed system is to support teleoperated navigation with real-time mapping, and by this, leaving more freedom to operators for performing any other task.

During the last decades a rich set of solutions for building maps from 2D laser range data have been proposed. Lu and Milios [1997] presented the IDC algorithm that can be applied in non-polygonal environments. Cox [1990] proposed a method particularly suited for polygonal environments for matching range readings with a priori given line mode, and Gutmann [2000] presented a method combining it with IDC. A robust grid-based method has been presented by Hähnel [2005] that aligns scans on a grid map successively build over time.

In contrast to scan matching methods, higher sophisticated methods, such as *FastSlam* [Montemerlo *et al.*,

Fig. 1. Test setting during DisasterCity, Texas, 2008: (a) Test maze inclined by 15° and additionally covered with rolls and ramps (obscured during experiments), (b) Responders teleoperating the robot with assistive mapping. (c) Map generated online during teleoperation.

2002], and *GMapping* [Grisetti *et al.*, 2005], have been introduced. These methods are capable of correcting the entire map at once when loop-closures, i.e., re-visits of places, have been detected. However, in most cases they have to be applied offline on recorded sensor readings. Olson *et al.* [2006] presented an optimization approach that applies stochastic gradient descent for resolving relations in a network efficiently. Extensions of this work have been presented by Grisetti *et al.* [2007b; 2007a] Most approaches to graph-based SLAM such as the work of Olson *et al.*, Grisetti *et al.*, and others, assume that the relations are given.

Although existing methods are capable of dealing with sensor noise, they do require reasonable pose estimates, e.g., such as from wheel odometry, as an initial guess for the mapping system. However, wheel odometry has shown to be unreliable given an unpredictable amount of wheel slip, which is particularly the case when navigating robots on rough terrain. Furthermore, methods performing loop-

closures are mostly not applicable in real-time since their computational needs can unpredictably increase within unknown environments.

The system introduced in this paper aims on the application scenario of realistic teleoperation. Under certain constraints, such as low visibility and rough terrain, first responder teleoperation leads to very noisy and unusual data. For example, due to environmental structures and failures in control, laser scans are frequently taken under varying roll and pitch angle, making it difficult to reliably find correspondences from successive measurements. In contrast to many "artificially" generated data logs, logs from teleoperation only seldom contain loops.

Most existing methods work after the principle of minimizing the squared sum of error distances between successive scans by searching over scan transformations, i.e., rotations and translations. Scan point correspondences are decided once before the search starts based on the Euclidean distance. In contrast to other methods, the proposed approach re-considers data associations during the search, which remarkably increases the robustness of scan matching on rough terrain. The algorithm processes data from laser range finder and gyroscope only, making it easily applicable on different robot platforms, and more importantly, independent of faulty readings from wheel odometry.

The mapping system has been implemented as a fan-less embedded system which can easily be attached to different robot types. The system has been extensively tested on robot platforms designed for teleoperation in critical situations, such as bomb disposal. Furthermore, the system has been evaluated in a test maze by first responders during the Disaster City event in Texas 2008. Experiments conducted within different environments show that the system yields comparably accurate maps in real- time when compared to higher sophisticated offline methods, such as Rao-Blackwellized SLAM.

The remainder of this paper is structured as follows. In Section II the mapping algorithm, and in Section III the implementation of the mapping system are described. Results from experiments with different robots and environments are shown in Section IV. In Section V the conclusion is presented.

II. VOTING-BASED SCAN MATCHING

In this section a *two-step processing* of laser and gyro data for incrementally tracking the robot's trajectory is described. Given a sequence of scans $\{S_t, S_{t-1}, S_{t-2}, ...\}$, where each scan is defined by a set of cartesian points $S : \left\{ s_i = (x_i, y_i)^T \mid i = 0...n - 1 \right\}$ relative to the robot center, and a sequence of gyro readings $\{\psi_t, \psi_{t-1}, \psi_{t-2}, ...\}$, the relative transformation $\tau = (\Delta x, \Delta y, \Delta \theta)^T$ of the robot pose within the last time interval Δt is computed. Note that the concatenation of all computed transformations yields an estimate of the true robot trajectory which in turn can be used to integrate scans on a grid map yielding the result presented in Figure 1 (c).

By the first step an initial guess of τ is computed by incrementally aligning successive scans. By the second step this initial guess is utilized for finding the final transformation of the current scan with respect to the history of scan observations. By considering the history of observations, local misalignments that occurred during the first step can be corrected.

A. Incremental scan alignment

Scans are preprocessed by a scan reduction filter in order to minimize both sensor noise and computation time of the algorithm. This is carried out by clustering scan points that are within a certain Euclidean distance δ from a common cluster center. In our implementation a cluster radius of $\delta = 5cm$ has been selected. Note that in contrast to simply reducing the resolution of the scanner, cluster-based filtering preserves relevant structure of the environment since all measurements are taken into account for computing cluster centers.

Scan alignment is taking place as shown by Algorithm 1. The procedure takes as inputs the current scan observation and the change of the gyro angle $\Delta \psi$ observed between the current and previous scan (line 1). The current scan is rotated backwards with respect to the gyro change, and then further transformed according to the result of a voting-based search procedure.

The search procedure considers the set of transformations T generated by the function $genTrans(\Delta t)$ (line 4). This function randomly samples a transformation from a set of discretized transformations. The selection prefers transformations, i.e. selects them with higher probability, which are close to transformations that have been selected in the past. This is carried out by maintaining a short-term history of transformations, which is updated online after each estimation step. The idea is motivated from the fact that within a fixed time frame (depending on the data rate of the laser scanner) only a limited amount of motion change can be achieved due to inertia forces. For example, in case of a continuous fast forward motion of the robot, backward transformations are unlikely to be selected by this function.

The function $matchRadius(.)$ returns the Euclidean distance within which scan points are considered as being matched. According to the sensor model of the scanner, the returned distance depends on the spot size of the beam, and by this, on the beam's range. For example, longer point distances are returned for further ranges since they are accompanied with a larger beam diameter when hitting an object.

Although implementing three nested loops, the procedure turned out to be remarkably performant in practice. We determined experimentally that due to the break conditions in average less than 20% of all beams are considered by the algorithm. Note that the search over

Algorithm 1: Voting-based scan matching

Input: Scan S, gyro change $\Delta\psi$
Output: Transformation $\tau = (\Delta x, \Delta y, \Delta\theta)^T$
Data: Reference Scan $R \leftarrow \emptyset$

```
   // Rotate S by gyro change Δψ:
1  foreach s_i ∈ S do
2       s_i ← s_i ( cos−Δψ   −sin−Δψ )
                  ( sin−Δψ    cos−Δψ );
3  end
   // Generate set of transformations T:
4  T ← genTrans(time(S) − time(R)) ;
   // Find best transformation τ_best:
5  bestVote ← 0 ;
6  τ_best ← ∅ ;
7  foreach τ_j ∈ T do
8       foreach s_i ∈ S do
9            s_i ← s_i ( cos Δθ   −sin Δθ ) + (Δx,Δy)^T;
                       ( sin Δθ    cos Δθ )
10           foreach r_i ∈ R do
11                if distance(s_i, r_i) < matchRadius(range(s_i))
                   then
12                     vote_j ← vote_j + 1 ;
13                     break ;
14                end
15           end
16           if vote_j + remaining(S) < bestVote then
17                break ;
18           end
19      end
20      if vote_j > bestVote then
21           bestVote ← vote_j ;
22           τ_best ← τ_j;
23      end
24      R ← S ;
25 end
   // Return best transformation found
26 return τ_best;
```

---

possible data associations increases the robustness of the matching significantly.

B. Grid-based scan matching

The scan alignment described above provides a good initial guess of the robot transformation. However, more robustness is achieved by matching scans against a history of observations. This is carried out by grid-based scan matching, as described in Hähnel [2005]. The technique determines from a history of scan observations $S_t, S_{t-1}, ..., S_{t-n}$ and transformations $\tau_t, \tau_{t-1}, ..., \tau_{t-n}$, computed by the first step, the estimated robot pose $\hat{x}_t$ relative to the start location.

At each sensor observation S_t, a local grid map $\hat{m}(\hat{x}_{t-n:t-1}, S_{t-n:t-1})$ is constructed from the previous n scans and pose estimates. The map is then convoluted with a Gaussian kernel in order to accelerate the scan alignment search procedure.

Given current transformation τ_t and scan observation S_t, the new pose estimate x_t is then computed by maximizing the scan alignment on the grid map:

$$\hat{x}_t = \underset{x_t}{argmax} \\ \{p(S_t|x_t, \hat{m}(\hat{x}_{t-n:t-1}, S_{t-n:t-1})) \cdot p(x_t|\tau_t, x_{t-1})\}, \quad (1)$$

where $p(x_t|\tau_t, x_{t-1})$ denotes the probability that the robot is located at x_t given the transformation computed by step 1. Finally, new pose x_t and scan S_t are added to the history buffer.

III. Implementation

The goal of the proposed mapping system is to enable online mapping on commercial robot platforms during first responder teleoperation. For achieving this goal, the mapping system has to be embeddable on these platforms, wherefore three requirements have to be fulfilled: First, the device has to be small enough to fit onto the platform. Second, the device has to be waterproof, i.e. based on a fan-less CPU. Third, the device has to communicate via the existing communication link of the robot, which is typically an analog video transmitter. The mapping
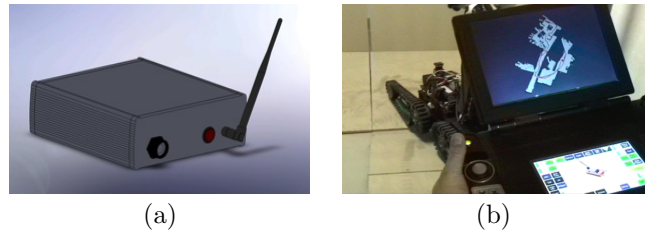

(a)   (b)

Fig. 2. Implementation of the mapping system as an embedded system: The black-box "SensorHead". (b) The box integrated on the Telemax robot

system has therefore been implemented as an embedded system (see Figure 2 (a)). The black box, which we named „SensorHead", processes sensor readings from an internal IMU and an externally attached laser range finder (LRF). The unit is directly powered from the power supply of the robot (either 24 V or 48 V). Note that no other connections, e.g. from wheel encoders, are required.

Robots designed for teleoperation are typically forwarding video signals from cameras either via a radio link or tethering. Thus, the main output of the box is a video signal that can directly be fed into the on-board video system of the robot and then being transmitted to the operator console. Figure 2 (b) depicts the integration of the SensorHead on a Telemax robot with operator console displaying the generated map and current position.

The SensorHead contains a 3.5" *Wafer-Atom* board equipped with a 1.6 GHz Intel Atom CPU, and 2 GB memory, a 64 GB solid state disk, a *Xsens MTi* Inertial Measurement Unit (IMU), and a video converter. Online computed maps can additionally be received via Wireless-LAN.

IV. Experiments

The mapping system has been evaluated on several robot platforms, e.g., *Telemax* (Telerob GmbH), *Talon Responder* (Foster-Miller), *Matilda* (Mesa Robotics), *Pioneer AT* (ActiveRobots), and a Kyosho Twin Force R/C car. In this section results from experiments conducted on these platforms will be presented.

A. First Responder Evaluation at DisasterCity

The first responder evaluation during DisasterCity was organized by the National Institute of Standards and Technology (NIST). Within time slots of 15 minutes multiple teams consisting of two first responders had to localize and report hazmat symbols that were deployed in a maze-like structure beforehand (see Figure 1 (a)). On the one hand there were teams exploring the maze by manual mapping, and on the other hand, teams that utilized the output of our mapping system at the operator console, as shown in Figure 1 (b).

To navigate robots through the maze was a challenging task due to an overall inclination of 15° and additional rolls and ramps (either 10° or 15° inclined) that covered the maze entirely. Furthermore, the maze was obscured. Thus, hazmat symbols and the structure of the maze had to be recognized by the responders via the robot's on-board cameras and lights illuminating the scene. Due to this extraordinarily harsh conditions, some responders even failed to simultaneously detecting targets, and controlling the robot. Consequently, they had major difficulties to localize within the maze just by observing the video stream transmitted from the robot, leading frequently to situations where regions had been explored twice. In contrast, some responders demonstrated efficient maze navigation, e.g., for searching hazmat symbols, or quickly exiting the maze from any given location, by using the proposed mapping system.

Finally, our system repeatedly mapped the maze (see Figure 1 (c)) under extremely harsh conditions: Unexperienced responders drove the robot at maximal velocity over rolls and ramps causing jumps and heavy collisions. Although robust robot platforms have been used for the evaluation (e.g. Talon and Matilda), experiments had to be restarted at least five times because the teleoperated robot had been turned over or major malfunctions occurred.

B. Quantitative Evaluation

The second experiment consist of a quantitative evaluation of our approach (*DCMapping*) compared to rao-blackwellized particle filtering (RBPF) utilizing loop closures for improving the map [Grisetti *et al.*, 2005]. More specifically, we run the *GMapping* implementation which is freely available on the Internet [Stachniss *et al.*, 2007]. *GMapping* requires data from wheel odometry in order to compute the map. Since our log files do not contain wheel odometry, *GMapping* has been run with the output of the first level of the scan processing described in Section II. We will first describe our evaluation methodology and then provide results from various different experiments.

1) Metric Evaluation: We based our evaluation on the estimated robot trajectory $x_{1:T}$, where x_i is the robot pose at timestep i from 1 to T. Let $x^*_{1:T}$ be the reference poses of the robot, ideally the true locations. We use a measure based on the *relative* displacement $\delta_{i,j}$ between poses to perform comparisons. We define $\delta_{i,j} = x_i \ominus x_j$, where

$\oplus$ is the standard motion composition operator and $\ominus$ its inverse. Instead of comparing x to x^* (in the global reference frame), we do the operation based on δ and δ^* as

$$\varepsilon(\delta) \quad = \quad \sum_{i,j} (\delta_{i,j} \ominus \delta^*_{i,j})^2. \qquad (2)$$

A more detailed description can be found in our previous work [Burgard *et al.*, 2009].
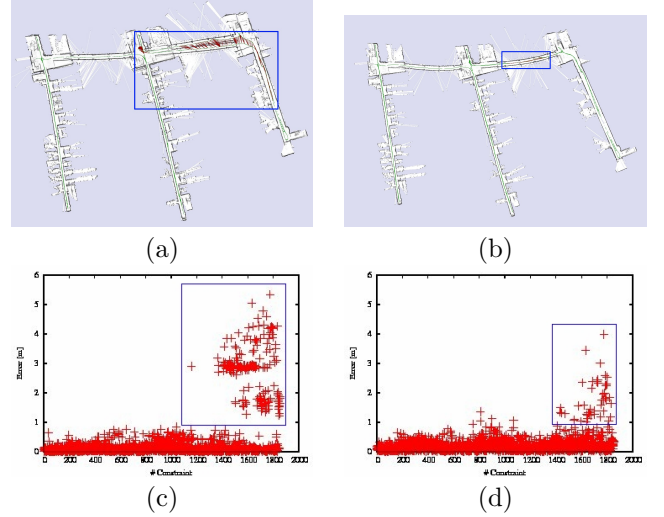

Fig. 3. Comparison of results on the Imtek dataset. The left side shows the map generated by scan matching (a) and the translational error plotted by relation (c). On the right side the map by RBPF (b) is shown with the corresponding error plot (d). The maps are overlaid with the graph formed by the given relations. Both error plots show a remarkable cluster towards the end. Rectangles mark the corresponding relations in the plot and the map. All marked relations are also drawn in red in the map.

Although we do not need true locations x^* anymore, we still need the true displacements δ^*. We use a two step approach to derive those in an assisted manner: First, we run a SLAM algorithm under manual supervision on the raw data to estimate a trajectory, that is globally consistent, and derive initial displacement candidates $\delta^i_{i,j}$ from that. Next, every candidate $\delta^i_{i,j} = x_i \ominus^i x_j$ is manually verified. In this step a human expert will be presented the two laserscans at timestep i and j displaced by $\ominus^i$. The expert can *accept* or *reject* the displacement for the final displacements $\delta^*_{i,j}$ and in the case of an *accept* it is possible to manually adjust $\ominus^i$ to reflect $\ominus^*$. Although work intensive, we believe, that without manual intervention or other external sources it is impossible to generate reliable results.

One advantage of this metric is that it allows us to compare algorithms in individual situations. An example of a difficult environment with long hallways and glass walls is shown in Figure 3. The plots show the error plotted by relation and large clusters are visible for both the RBPF and DCMapping. We identified the relations in those clusters and marked the corresponding relations in the resulting maps. The clusters clearly identify the weak

TABLE I

QUANTITATIVE RESULTS OF DIFFERENT APPROACHES/DATASETS ON THE TRANSLATIONAL ERROR.

Translational error m (abs) / m^2 (sqr)	DCMapping	RBPF (50 part.)
082er		
abs. errors	**0.072 ± 0.066**	0.115 ± 0.122
squared errors	**0.01 ± 0.024**	0.028 ± 0.074
Maximum abs. error	**0.73**	1.06
aces		
abs. errors	0.121 ± 0.335	**0.068 ± 0.078**
squared errors	0.127 ± 0.719	**0.011 ± 0.035**
Maximum abs. error	2.803	**0.646**
tu-darmstadt		
abs. errors	0.228 ± 0.643	**0.122 ± 0.146**
squared errors	0.465 ± 2.513	**0.036 ± 0.188**
Maximum abs. error	5.942	**1.921**
tu-graz		
abs. errors	**0.054 ± 0.044**	0.112 ± 0.186
squared errors	**0.005 ± 0.009**	0.047 ± 0.312
Maximum abs. error	**0.318**	2.515
imtek		
abs. errors	0.42 ± 0.942	**0.25 ± 0.416**
squared errors	1.063 ± 4.152	**0.235 ± 2.073**
Maximum abs. error	8.463	**6.998**
intel-lab		
abs. errors	0.136 ± 0.132	**0.07 ± 0.082**
squared errors	0.036 ± 0.068	**0.012 ± 0.033**
Maximum abs. error	0.8	**0.687**
mit-killian		
abs. errors	**3.71 ± 12.046**	7.505 ± 26.137
squared errors	**158.832 ± 639.92**	739.314 ± 3112.13
Maximum abs. error	**60.17**	153.087
telemax_hardcore		
abs. errors	**0.108 ± 0.136**	0.274 ± 0.276
squared errors	**0.03 ± 0.109**	0.152 ± 0.314
Maximum abs. error	**1.334**	1.694
hangar		
abs. errors	**0.207 ± 0.443**	0.291 ± 0.527
squared errors	**0.239 ± 1.215**	0.362 ± 1.648
Maximum abs. error	**3.044**	3.646
dc-maze		
abs. errors	**0.173 ± 0.199**	1.490 ± 2.230
squared errors	**0.070 ± 0.164**	7.179 ± 13.600
Maximum abs. error	**0.967**	7.180

TABLE II

QUANTITATIVE RESULTS OF DIFFERENT APPROACHES/DATASETS ON THE ROTATIONAL ERROR.

Rotational error deg (abs) / deg^2 (sqr)	DCMapping	RBPF (50 part.)
082er		
abs. errors	**1.334 ± 1.571**	1.563 ± 1.877
squared errors	**4.245 ± 9.923**	5.964 ± 13.904
Maximum abs. error	**10.558**	12.501
aces		
abs. errors	2.518 ± 3.368	**1.675 ± 2.133**
squared errors	17.678 ± 72.465	**7.351 ± 19.952**
Maximum abs. error	45.015	**14.842**
tu-darmstadt		
abs. errors	0.667 ± 0.886	**0.558 ± 0.674**
squared errors	1.231 ± 3.875	**0.765 ± 2.477**
Maximum abs. error	7.628	**6.794**
tu-graz		
abs. errors	**1.13 ± 1.205**	1.354 ± 1.44
squared errors	**2.727 ± 6.551**	3.906 ± 10.847
Maximum abs. error	**9.09**	15.092
imtek		
abs. errors	**0.899 ± 1.28**	1.041 ± 1.476
squared errors	**2.445 ± 7.737**	3.26 ± 10.363
Maximum abs. error	**9.352**	10.457
intel-lab		
abs. errors	3.661 ± 6.048	**2.494 ± 3.63**
squared errors	49.968 ± 182.194	**19.395 ± 95.383**
Maximum abs. error	**47.267**	50.908
mit-killian		
abs. errors	**2.043 ± 3.781**	4.592 ± 14.08
squared errors	**18.463 ± 79.961**	219.299 ± 875.622
Maximum abs. error	**38.678**	71.114
telemax_hardcore		
abs. errors	**1.38 ± 1.395**	2.339 ± 2.791
squared errors	**3.849 ± 8.708**	13.251 ± 30.709
Maximum abs. error	**9.129**	14.778
hangar		
abs. errors	3.67 ± 4.096	**2.434 ± 2.696**
squared errors	30.222 ± 93.017	**13.183 ± 42.161**
Maximum abs. error	40.089	**21.584**
dc-maze		
abs. errors	**4.568 ± 3.649**	17.306 ± 20.294
squared errors	**34.148 ± 55.194**	710.284 ± 1051.333
Maximum abs. error	**20.300**	59.889

point in the map that originates from the robot returning after driving a long hallway without features. DCMapping is here unable to close the loop fully and this shows in a shearing effect. The RBPF can close the loop. There are less relations with a lower magnitude in its cluster as they only originate from length errors and the slight bend of the corridor.

2) Results: Experiments have been carried out on two types of log files and have been executed on a *Intel Core2Duo 2.6GHz*. On the one hand we evaluated logs that have been recorded during teleoperated exploration of building structures. These are the *082er* log, recorded in a cellar, the *tu-darmstadt* log, recorded in a long hallway of the Darmstadt university, the *tu-graz* log, recorded in a cluttered office environment of the Graz university, the *imtek* log, recorded in three interconnected buildings of the Freiburg university, the *telemax-hardcore* log, recorded in a cellar while driving over ramps and rolls, the *hangar* log, recorded in a hangar like structure, and the *dc-maze* log, recorded at DisasterCity in Texas. On the other hand we evaluated logs that are typically used by the SLAM community and online available. These are *aces*, *intel-lab*, and *mit-killian*. Note that the latter logs contain comparably many situations where the robot re-entered previously explored regions, i.e., enabling loop closures by the algorithm. Figure 4 depicts the result of DCMapping for some of the maps.

The evaluation presented in Table I and Table II shows that DCMapping and RBPF are yielding in average comparably equal good results. This is surprising since RBPF needs much more time to compute the corrected map than DCMapping, which provides results in real-time. For example, we measured for RBPF on the embedded system described in Section III a run time of 330 minutes when processing the 82er log. In contrast, DCMapping took 6 minutes (real-time), which is about 55 times faster. Note, that particularly when closing larger loops, the computation time of RBPF increases drastically. Differences can be found, on the one hand, on maps containing larger loops, such as *aces*, *intel-lab*, and *imtek*, where RBPF shows its strength in closing loops, and on the other hand, *telemax-hardcore* and *maze* where DCMapping clearly shows robustness against extremely faulty laser range readings from rolls and ramps.
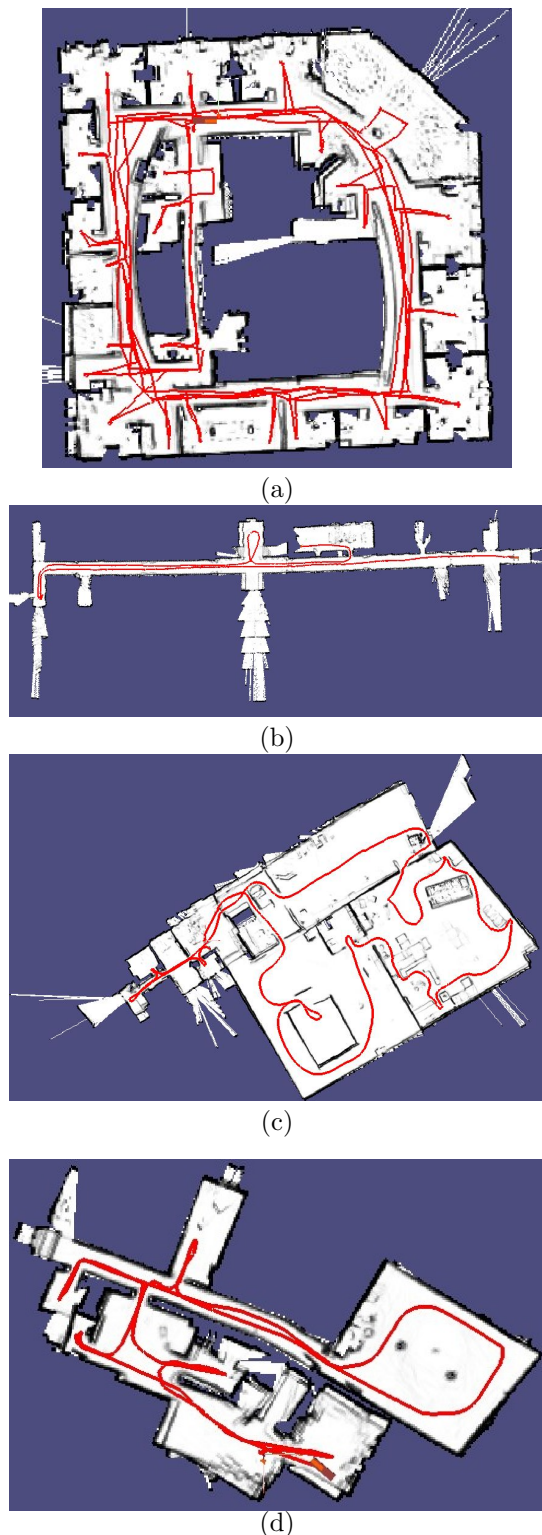
(a)

(b)

(c)

(d)

Fig. 4. Maps generated with DCMapping: (a) *aces*, (b) *tu-darmstadt*, (c) *hangar*, and (d) *082er*. The red line indicates the path taken by the robot.

V. Conclusion

We introduced a mapping system for assisting navigation tasks of first responders teleoperating a robot on difficult terrain. The system has been intensively evaluated in diverse environments, and has also been tested by first responders. The presented results show that the quality of generated maps is close to that generated by computational costive algorithms. Moreover, the system has been considered as advantageous for teleoperation by most first responders testing it in the field.

We showed that DCMapping yields map accuracy comparable to RBPF, however, by requiring remarkably less CPU resources. We assume that combining both RBPF and DCMapping will lead to the best performance. For example, data preprocessed online by DCMapping for navigation can further be processed by RBPF for generating accurate maps. By this, the advantages of both, e.g., the robustness regarding faulty measurements of DCMapping, and the loop closure capabilities of RBPF, can effectively be combined. In future work we will consider this idea.

References

[Burgard *et al.*, 2009] W. Burgard, C. Stachniss, G. Grisetti, B. Steder, R. Kümmerle, C. Dornhege, M. Ruhnke, A. Kleiner, and J.D. Tardós. Trajectory-based comparison of slam algorithms. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, 2009. To appear.

[Cox, 1990] I. J. Cox. Blanche: position estimation for an autonomous robot vehicle. pages 221–228, 1990.

[Grisetti *et al.*, 2005] G. Grisetti, Stachniss C., and Burgard W. Improving grid-based SLAM with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 667–672, Barcelona, Spain, 2005.

[Grisetti *et al.*, 2007a] G. Grisetti, S. Grzonka, C. Stachniss, P. Pfaff, and W. Burgard. Efficient estimation of accurate maximum likelihood maps in 3D. In *Proc. of the Int. Conf. on Intelligent Robots and Systems (IROS)*, San Diego, CA, USA, 2007.

[Grisetti *et al.*, 2007b] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Proc. of Robotics: Science and Systems (RSS)*, Atlanta, GA, USA, 2007.

[Gutmann, 2000] J.-S. Gutmann. *Robuste Navigation autonomer mobiler Systeme*. PhD thesis, Albert-Ludwigs-Universit"at Freiburg, 2000. ISBN 3-89838-241-9.

[Hähnel, 2005] D. Hähnel. *Mapping with Mobile Robots*. PhD thesis, Universität Freiburg, Freiburg, Deutschland, 2005.

[Lu and Milios, 1997] Feng Lu and Evangelos Milios. Robot pose estimation in unknown environments by matching 2d range scans. *J. Intell. Robotics Syst.*, 18(3):249–275, 1997.

[Montemerlo *et al.*, 2002] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: a factored solution to the simultaneous localization and mapping problem. In *Eighteenth national conference on Artificial intelligence*, pages 593–598, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[Olson *et al.*, 2006] E. Olson, J. Leonard, and S. Teller. Fast iterative optimization of pose graphs with poor initial estimates. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 2262–2269, 2006.

[Stachniss *et al.*, 2007] C. Stachniss, U. Frese, and G. Grisetti. OpenSLAM.org – give your algorithm to the community. http://www.openslam.org, 2007.