ELSEVIER

Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

Robotics and Autonomous Systems 51 (2005) 275–296

Robotics and Autonomous Systems

www.elsevier.com/locate/robot

# High-speed laser localization for mobile robots

Kai Lingemann[a,*], Andreas Nüchter[a], Joachim Hertzberg[a], Hartmut Surmann[b]

[a] *University of Osnabrück, Institute of Computer Science, Knowledge-Based Systems Research Group, Albrechtstr. 28, D-49069 Osnabrück, Germany*

[b] *Fraunhofer Institute for Autonomous Intelligent Systems (AIS), Schloss Birlinghoven, D-53754 Sankt Augustin, Germany*

Received 24 April 2004; received in revised form 21 December 2004; accepted 2 February 2005
Available online 7 April 2005

**Abstract**

This paper describes a novel, laser-based approach for tracking the pose of a high-speed mobile robot. The algorithm is outstanding in terms of accuracy and computation time. The efficiency is achieved by a closed-form solution for the matching of two laser scans, the use of natural scan features and fast linear filters. The implemented algorithm is evaluated with the high-speed robot Kurt3D (4 m/s), and compared to standard scan matching methods in indoor and outdoor environments.

*Keywords:* Localization; Pose tracking; Autonomous mobile robots; Scan matching; High-speed robotics

## 1. Introduction

In order to fulfill user specified tasks, autonomous mobile robots have to be able to determine their pose (position and orientation) while moving — a precise and stable self-localization is one of the most important requirements to act purposefully in any environment. In earlier publications we proposed methods for solving 6D-SLAM (*simultaneous localization and mapping*) based on 3D laser range data [1,2]. These algorithms depend on a sufficiently correct pose estimation of the vehicle. Under certain restrictions, e.g., driving with low velocity and on non-slippery ground like carpet, the robot odometry provides such an estimation. However, in general situations, especially when driving with high speed, a more reliable and accurate pose tracking technique is needed. Furthermore, high computation speed of a tracking method is not only essential when driving with high velocities. Regardless of the velocity, it also enables the further application of computationally expensive algorithms while driving, minimizing the resources needed for the localization task.

[*] Corresponding author. Tel.: +49 541 969 2558; fax: +49 541 969 2799.

*E-mail addresses:* lingemann@informatik.uni-osnabrueck.de (K. Lingemann), nuechter@informatik.uni-osnabrueck.de (A. Nüchter), hertzberg@informatik.uni-osnabrueck.de (J. Hertzberg), surmann@ais.fraunhofer.de (H. Surmann).

0921-8890/$ – see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.robot.2005.02.004

For an application on flat ground, the pose $\boldsymbol{p}$ of a robot is denoted by a triple $\boldsymbol{p} = (x, y, \theta)$, with position $(x, y)$ and orientation $\theta$. Standard robotic approaches often use dead reckoning, e.g., odometry (counting wheel rotations), to estimate their position. But wheels slip, especially when driving with high speed, and miscalculation occurs.[1] These errors accumulate over time, and the robot's position estimate becomes increasingly inaccurate. One method to improve localization is to track the pose with an on-board laser range finder. There are, in principle, two different approaches to determine the robot's pose:

- *Relative*: Given an arbitrary initial pose, e.g., $\boldsymbol{p}_0 = (0, 0, 0)$, the robot's current pose relative to $\boldsymbol{p}_0$ is incrementally updated when the robot is moving. For this purpose, consecutively acquired sensor data, e.g., laser scan data, are compared. Let $\mathcal{S}$ be the *current* scan and $\mathcal{R}$ the *reference* scan, e.g., a scan acquired previous to $\mathcal{S}$. If both scans have been acquired from different robot poses, a transformation that maps $\mathcal{S}$ onto $\mathcal{R}$ is calculated that corresponds to the movement of the robot between the scans.
- *Absolute*: Instead of tracking the pose incrementally, the position and orientation of the robot within an a priori given, assumable exact map or a known area with distinguishable active or passive landmarks is calculated.

In the following, we mainly consider relative localization techniques, i.e., so-called *pose tracking* algorithms, since they do not impose any limitation to already mapped environments. Nevertheless, the paper concludes with a demonstration of the algorithm's ability to be used for absolute localization as well. The aim of relative pose tracking is not directly to build an accurate 2D map, but rather to ensure a stable and fast localization, regardless of the robot's speed and without any restrictions of the covered distance. This aim reflects the underlying idea that localization, the preliminary stage of SLAM, is not an end in itself, but the fundament of further algorithms, e.g., the building of three-dimensional environmental maps with a 3D sensor and solving 6D-SLAM [1,2]. The localization algorithm has to be computationally efficient and should not restrict the operational area or the speed of the mobile robot. Hence, subsequent algorithms for improving the calculated map, like cycle detection and correction [3] or global relaxation [4], are not considered here.

The implemented algorithm is evaluated with the high-speed robot Kurt3D (Fig. 1), equipped with the AIS 3D laser scanner that is adjusted horizontally while driving and used for localization purpose [2]. Kurt3D is a mobile robot platform with a size of 45 cm × 33 cm × 26 cm (length × width × height) and a weight of 15.6 kg. Equipped with the 3D laser range finder the height increases to 47 cm and weight increases to 22.6 kg. Kurt3D's maximum velocity is 5.2 m/s (autonomously controlled 4.0 m/s), with a maximal acceleration of 1.9 m/s$^2$. Two 90 W motors are used to power the six wheels, whereas the front and rear wheels have no tread pattern to enhance rotating. Kurt3D operates for about 4 h with one battery (28 NiMH cells, capacity: 4500 mAh) charge. The core of the robot is a Pentium-III-800 MHz with 384 MB RAM, running Linux. An embedded 16-bit CMOS microcontroller is used to control the motor The implemented robot control that enables the robot to drive safely so fast will be presented in [5]. This paper is organized as follows. After presenting the state of the art in robot localization, we continue a description of the implemented tracking algorithm. The next section demonstrates results of real world experiments in (static and dynamic) indoor and outdoor scenarios. Section 5 concludes.

### 1.1. State of the art

Relative localization algorithms based on proximity sensors are distinguishable by the manner of computing the pose change between two sensor inputs acquired at two different poses. We concentrate on methods using a laser range finder, since it has significant, critical advantages in accuracy and processing speed compared to sonar [6], stereo-camera [7] and omnidirectional vision [8]. State of the art are algorithms comparing two scans directly (point-wise scan matching) [9] or utilizing special representations of the scanned data, e.g., histograms [10,11],

[1] A video is available at: http://www.ais.fhg.de/ARC/kurt3D/HAYAI/videos/localization_skidding.divx.avi.

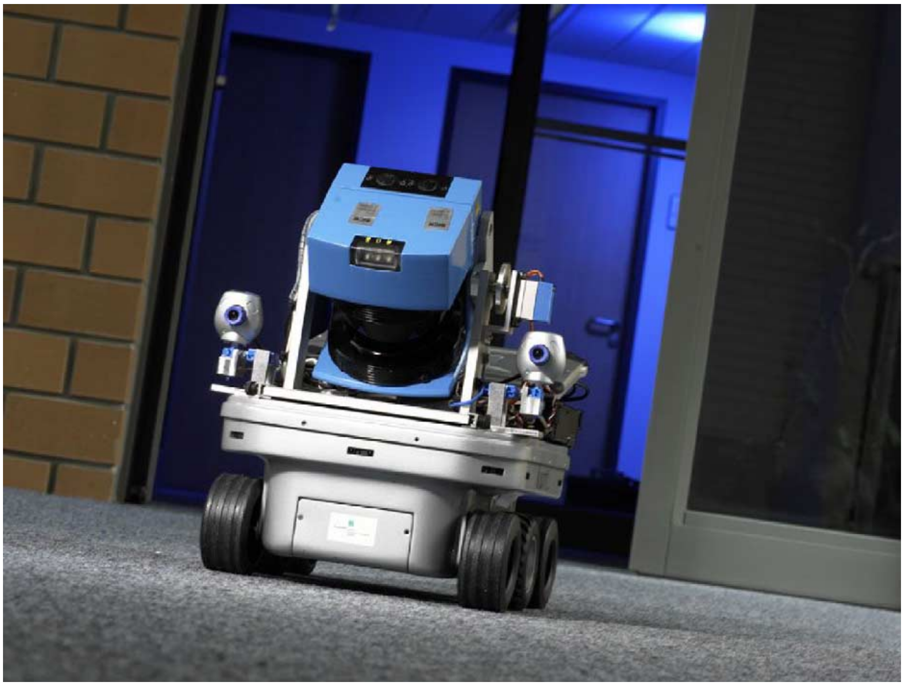

Fig. 1. The mobile robot Kurt3D, equipped with AIS 3D laser scanner.

features like extracted lines [12] or landmarks [13,14]. The latter ones are further distinguished by the type of landmarks, i.e., natural or artificial. In contrast to natural ones, artificial landmarks are, by definition, placed in the environment for simplifying the location task, being either active or passive.

The main drawbacks of current approaches are precision and computation cost, both in practical implementations as well as from a theoretical point of view, preventing a stable self-localization of a robot driving with high-speed (e.g., 4 m/s), especially when operating in non-restricted domains.[2] Problems arise from the iterative way of calculating an optimal match between two scans, preventing theoretically efficient computations, and from algorithms with complexities exponential in $\mathcal{O}(n^k), k > 1$, with $n$ the number of distance values of a scan. To match two scans, usually an error function is defined, implementing a metric on scan matches as an evaluation of the matching. Minimizing the error is a typical optimization problem with well-known problems. Generally, gradient descent methods are implemented, transforming the second scan such that the distance is reduced. Iteration leads to a local minimum of the error function. There are several algorithms that, in addition, compute the transformation within each epoch in an iterative way, employing a second level of iterations.

In this paper, we describe a method for computing the optimal transformation in a *closed-form* manner, eliminating any iteration. For evaluating the proposed algorithm, comparisons to three main scan matching methods, in the implementations of Gutmann [15], are done:

- IDC: Matching of raw data points of both scans. Instead of an elaborated feature detection, points are assigned according to their distance or angle [16].

[2] A video is available at: http://www.ais.fhg.de/ARC/kurt3D/HAYAI/videos/kurt3d_corridor.divx.avi.

- Cox: Data points from one scan are matched against lines, prior extracted from the other scan [17].
- Hist: Angle- and $xy$-histograms are calculated from both scans and compared based on a cross correlation function [18].

## 2. Scan matching with HAYAI

This section describes the newly developed algorithm HAYAI (*high-speed and yet accurate indoor/outdoor tracking*). The matching algorithm, used for pose tracking of the robot, is based on the following scheme:

(1) Detect features within scan $\mathcal{R}$, yielding feature set $M$ (*model set*). Likewise, extract feature set $D$ (*data set*) from scan $\mathcal{S}$.
(2) Search for pairwise corresponding features from both sets, resulting in two subsets $\check{M} \subseteq M$ and $\check{D} \subseteq D$.
(3) Compute the pose shift $\Delta \boldsymbol{p} = (\Delta x, \Delta y, \Delta\theta)^{\mathrm{T}}$ as the optimal transformation for mapping $\check{D}$ onto $\check{M}$.
(4) Update the robot's pose $\boldsymbol{p}_n \underset{\Delta \boldsymbol{p}}{\rightarrow} \boldsymbol{p}_{n+1}$ according to formula (1).
(5) Save the current scan as new reference scan $\mathcal{R} \leftarrow \mathcal{S}$.

Within this relatively conventional scheme, the contributions of HAYAI lie in fast filtering and extraction of natural features in laser scans (step 1) and a closed-form solution for computing the pose shift (step 3).

Given a pose $\boldsymbol{p}_n = (x_n, y_n, \theta_n)$ and a transformation $\Delta \boldsymbol{p} = (\Delta x, \Delta y, \Delta\theta)$, the transition $\boldsymbol{p}_n \underset{\Delta \boldsymbol{p}}{\rightarrow} \boldsymbol{p}_{n+1}$ is calculated as follows:

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \\ \theta_{n+1} \end{pmatrix} = \begin{pmatrix} x_n \\ y_n \\ \theta_n \end{pmatrix} + \begin{pmatrix} \cos\theta_n & \sin\theta_n & 0 \\ -\sin\theta_n & \cos\theta_n & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{pmatrix}. \tag{1}$$

The following subsections describe the implemented data filtering techniques, extraction and matching of the used features as well as the mathematical background and practical matters of calculating the optimal transformation between two scans.

### 2.1. Data filtering

Scanning is noisy and small errors may occur, namely Gaussian noise and so-called salt and pepper noise. The latter one arises for example at edges where the laser beam of the scanner hits two surfaces, resulting in a mean and erroneous data value. Furthermore reflections, e.g., at glass surfaces, lead to suspicious data. We propose two fast filtering methods to modify the data in order to enhance the quality of each scan, typically containing 181 data points. The data reduction, used for reducing Gaussian noise, works as follows. The scanner emits the laser beams in a spherical way, such that the data points close to the source are more dense. Multiple data points located close together are joined into one point. The number of these so-called *reduced points* is one order of magnitude smaller than the original one. A median filter removes the outliers by replacing a data point with the median value of the $n$ surrounding points (here: $n = 7$, covering an angle of $6^\circ$) [19]. The neighbor points are determined according to their number within the scan, since the laser scanner provides the data sorted in a counter-clockwise direction. The median value is calculated with regard to the Euclidean distance of the data points to the point of origin. In order to remove salt and pepper noise but leave the remaining data untouched, the filtering algorithm replaces a data point with the corresponding median value if and only if the Euclidean distance between both is larger than a fixed threshold $\delta$ (see Fig. 2, with $\delta = 200$ cm).

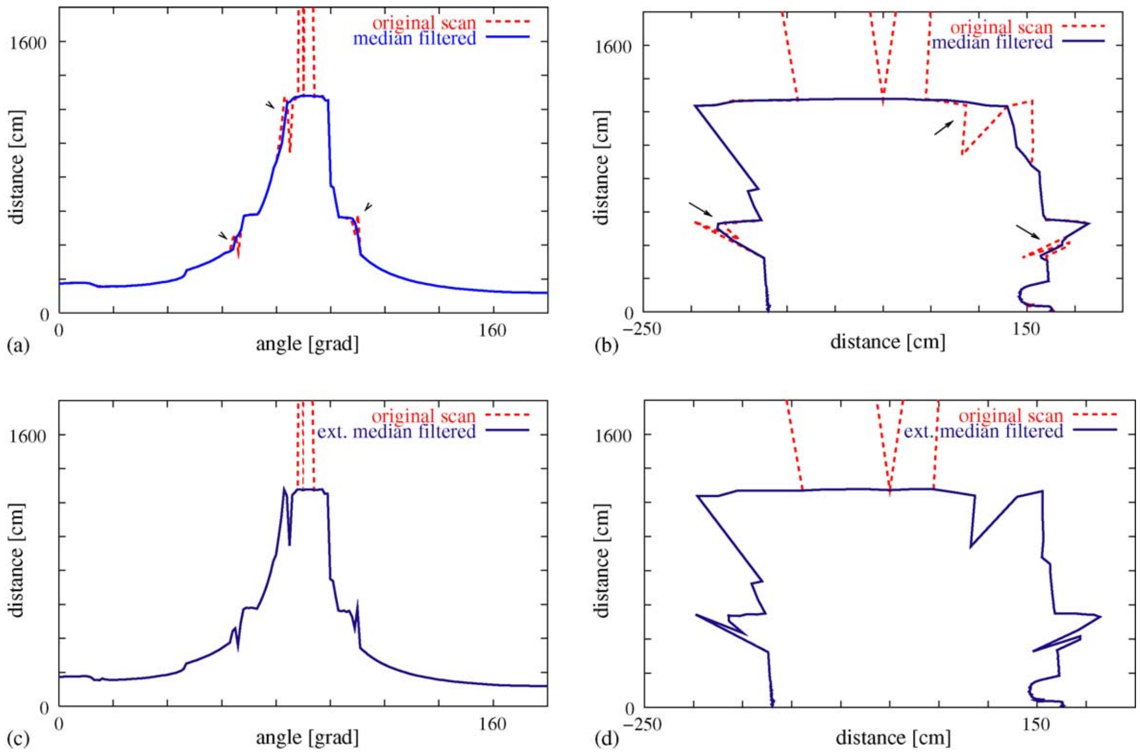

Fig. 2. Median filter (left polar, right Cartesian coordinates). Top: application of a standard median filter eliminates outliers, but also modifies the correct signal. Bottom: result of the extended median filter.

### 2.2. Extraction and matching of features

As described above, the scan matching algorithm computes a transformation $\Delta \boldsymbol{p}$ such that a *set of features*, extracted from the first scan, is mapped optimally to a feature set of the second scan. In order to be usable for a pose tracking algorithm, features have to fulfill two requirements:

- They have to be *invariant* with respect to rotation and translation.
- They have to be *efficiently* computable in order to satisfy real-time constraints.

Using the inherent order of the scan data allows the application of linear filters to be applied for a fast and reliable feature detection. The described algorithm chooses *extreme values* in the polar representation of a scan as natural landmarks. These extreme values correlate to corners and jump edges in Cartesian space. The usage of polar coordinates implicates a reduction by one dimension, since all operations deployed for feature extraction are fast linear filters, operating on the sequence of distance values $(r_i)_{i\in\mathbb{N}}$ of a scan $\mathcal{S} = ((\phi_i, r_i))_{i=1,\ldots,N}$.

Given a one-dimensional filter $\Psi = [\psi_{-1}, \psi_0, \psi_{+1}]$, the filtered value $r_i^{\Psi}$ of a scan point $r_i$ $(i = 1, \ldots, N-1)$ is defined as $r_i^{\Psi} = \sum_{k=-1}^{1} \psi_k \cdot r_{i+k}$. For feature detection, the scan signal is filtered as follows. Fig. 3 illustrates the effects of the used filters.

(1) Sharpen the data in order to emphasize the significant parts of the scan, i.e., the extreme values, without modifying the residual scan, by applying a sharpen filter of the form $\Psi_1 = [-1, 4, -1]$.

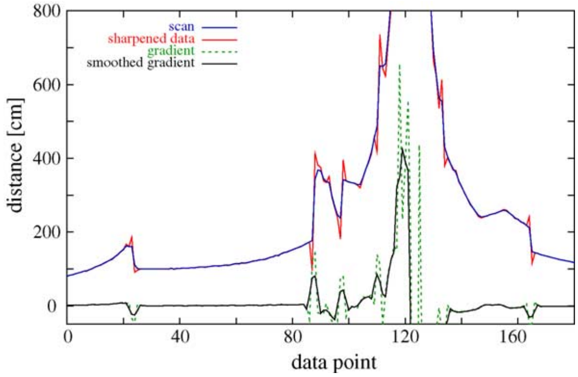

Fig. 3. Result of different filters used for feature extraction.

(2) Compute the derivation signal by using a gradient filter $\Psi_2 = [-1/2, 0, 1/2]$.
(3) Smooth the gradient signal to simplify the detection of zero crossings with a soften filter $\Psi_3 = [1, 1, 1]$.
(4) Feature points correspond to zero-crossings in the gradient signal iff their respective distances $r_i$ are smaller than a threshold $r_{\max}$ (here 800 cm), since the scanner's precision decreases with increasing distance.

After generating the sets of features $M$, $D$ from both scans, a matching between both sets has to be calculated. Instead of solving the hard optimization problem of searching for an optimal match, we use a heuristic approach, utilizing inherent knowledge about the problem of matching features, e.g., the fact that the features' topology cannot change fundamentally from one scan to the following. The basic aim is to build a matrix of possible matching pairs, based on an error function defining the distance between two points $\boldsymbol{m}_i, \boldsymbol{d}_j$, with $\boldsymbol{m}_i = (m_i^x, m_i^y)^{\mathrm{T}}$ in Cartesian, respectively, $(m_i^\phi, m_i^r)^{\mathrm{T}}$ in polar coordinates, $\boldsymbol{d}_j$ analogously:

$$\mathrm{dist}(\boldsymbol{m}_i, \boldsymbol{d}_j) = \sqrt{\omega_1(m_i^\phi - d_j^\phi)^2 + \omega_2(m_i^r - d_j^r)^2} + \sqrt{\omega_3(m_i^x - d_j^x)^2 + \omega_4(m_i^y - d_j^y)^2} + \Theta(\boldsymbol{m}_i, \boldsymbol{d}_j)$$

with constants $(\omega_k)_{k\in\{1,\ldots,4\}}$, implementing a weighting between the polar and Cartesian distances. The function $\Theta$ inhibits matchings between two features of different types:

$$\Theta(\boldsymbol{m}_i, \boldsymbol{d}_j) = \begin{cases} 0, & \Gamma(\boldsymbol{m}_i) = \Gamma(\boldsymbol{d}_j), \\ \infty & \text{else} \end{cases}$$

with $\Gamma$ a classification function, $\Gamma : (M \cup D) \mapsto \{\text{maximum, minimum, inflection point}\}$.

The resulting matrix denoting corresponding features is simplified until the matching is non-ambiguous. See Fig. 4 for a matching of two scans.

### 2.3. Pose calculation

Given two sets of features $M = \{\boldsymbol{m}_i | \boldsymbol{m}_i \in \mathbb{R}^2, i = 1, \ldots, N_m\}$ and $D = \{\boldsymbol{d}_i | \boldsymbol{d}_i \in \mathbb{R}^2, i = 1, \ldots, N_d\}$, the calculation of the optimal transformation for mapping $D$ onto $M$ is a generic optimization problem. Intuitively, we are searching for a rotation $\boldsymbol{R}_{\Delta\theta}$ by the angle $\Delta\theta$ and a translation $\Delta\boldsymbol{t} = (\Delta x, \Delta y)^{\mathrm{T}}$ such that the differences between points from the reference and the corresponding points from the current scan, transformed with $\Delta\boldsymbol{p} = (\Delta x, \Delta y, \Delta\theta)$,

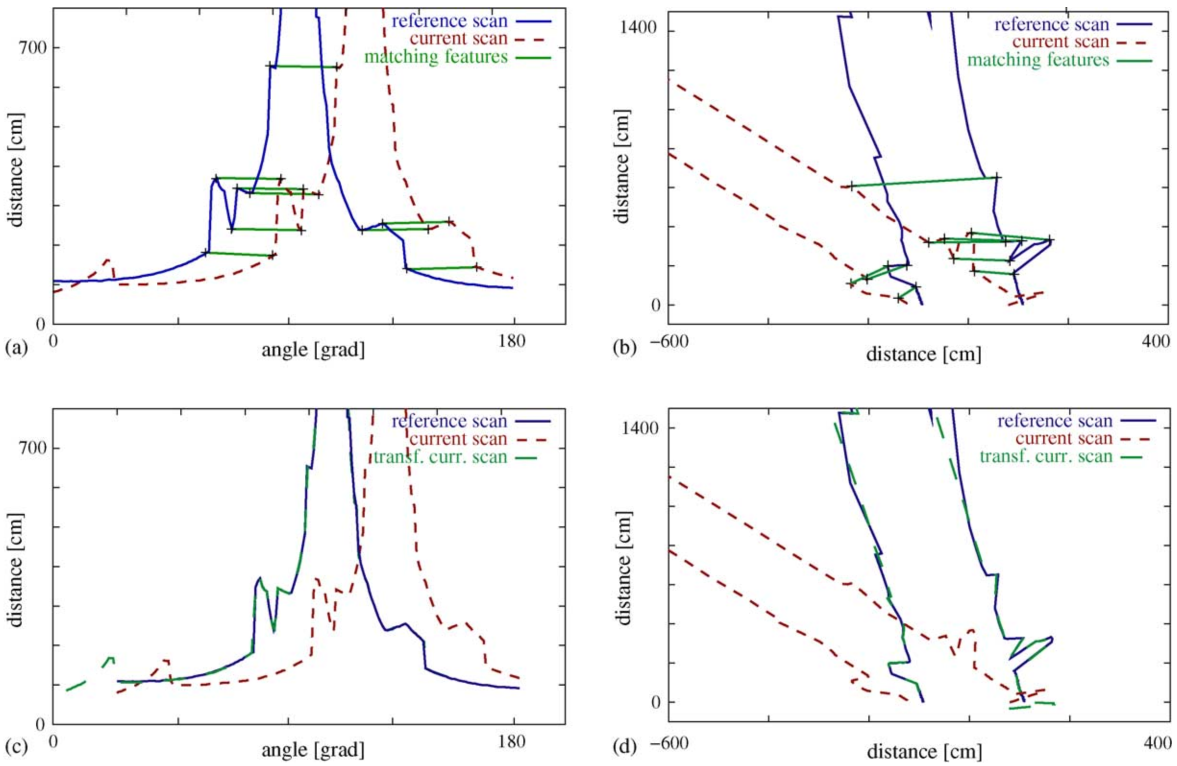

Fig. 4. Matching of features (left polar, right Cartesian coordinates). Top: detected features in both scans. Matched features are connected by a line. Bottom: the second scan is transformed based on the calculated rotation and translation.

are minimized. This results in a minimization of the error function

$$E(\boldsymbol{R}_{\Delta\theta}, \Delta\boldsymbol{t}) = \sum_{i=1}^{N_m}\sum_{j=1}^{N_d} w_{i,j}\|\boldsymbol{m}_i - (\boldsymbol{R}_{\Delta\theta}\boldsymbol{d}_j + \Delta\boldsymbol{t})\|^2 \tag{2}$$

with weights $w_{i,j} = 1$ iff feature $\boldsymbol{m}_i$ corresponds to $\boldsymbol{d}_j$, and 0 otherwise.

Let $\check{M} = \{\check{\boldsymbol{m}}_i\}_{i=1,\dots,N} \subseteq M$ and $\check{D} = \{\check{\boldsymbol{d}}_i\}_{i=1,\dots,N} \subseteq D$ be sets of *matched* features, with $\check{\boldsymbol{m}}_i \sim \check{\boldsymbol{d}}_i\ \forall i \in 1, \dots, N$. Correspondence between two points, $\check{\boldsymbol{m}}_i \sim \check{\boldsymbol{d}}_i$ means that both data points denote the same point in the real scene. Therewith, the error function $E$ simplifies to

$$E^{(1)}(\boldsymbol{R}_{\Delta\theta}, \Delta\boldsymbol{t}) = \sum_{i=1}^{N} \|\check{\boldsymbol{m}}_i - (\boldsymbol{R}_{\Delta\theta}\check{\boldsymbol{d}}_i + \Delta\boldsymbol{t})\|^2. \tag{3}$$

Let $\hat{M}$, $\hat{D}$ be variants of $\check{M}$, $\check{D}$ so that each point is shifted by the centroid of the respective set:

$$\hat{M} = \{\hat{\boldsymbol{m}}_i = \check{\boldsymbol{m}}_i - \check{\boldsymbol{c}}_m | i = 1, \dots, N\}, \qquad \hat{D} = \{\hat{\boldsymbol{d}}_i = \check{\boldsymbol{d}}_i - \check{\boldsymbol{c}}_d | i = 1, \dots, N\}$$

with the centroids defined as

$$\check{\boldsymbol{c}}_m = \frac{1}{N}\sum_{i=1}^{N}\check{\boldsymbol{m}}_i, \qquad \check{\boldsymbol{c}}_d = \frac{1}{N}\sum_{i=1}^{N}\check{\boldsymbol{d}}_i.$$

Substitution into formula (3) leads to

$$E^{(1)}(\boldsymbol{R}_{\Delta\theta}, \Delta\boldsymbol{t}) = \sum_{i=1}^{N}\|\hat{\boldsymbol{m}}_i - \boldsymbol{R}_{\Delta\theta}\hat{\boldsymbol{d}}_i - \underbrace{(\Delta\boldsymbol{t} - \check{\boldsymbol{c}}_m + \boldsymbol{R}_{\Delta\theta}\check{\boldsymbol{c}}_d)}_{=\Delta\tilde{\boldsymbol{t}}}\|^2$$

$$= \sum_{i=1}^{N}\|\hat{\boldsymbol{m}}_i - \boldsymbol{R}_{\Delta\theta}\hat{\boldsymbol{d}}_i\|^2 \tag{4a}$$

$$- 2\Delta\tilde{\boldsymbol{t}} \cdot \sum_{i=1}^{N}(\hat{\boldsymbol{m}}_i - \boldsymbol{R}_{\Delta\theta}\hat{\boldsymbol{d}}_i) \tag{4b}$$

$$+ \sum_{i=1}^{N}\|\Delta\tilde{\boldsymbol{t}}\|^2. \tag{4c}$$

The second term (4a) is zero since the measurements refer to the centroids:

$$\begin{aligned}\sum_{i=1}^{N}(\hat{\boldsymbol{m}}_i - \boldsymbol{R}_{\Delta\theta}\hat{\boldsymbol{d}}_i) &= \sum_{i=1}^{N}(\check{\boldsymbol{m}}_i - \check{\boldsymbol{c}}_m - \boldsymbol{R}_{\Delta\theta}\check{\boldsymbol{d}}_i + \boldsymbol{R}_{\Delta\theta}\check{\boldsymbol{c}}_d) = \sum_{i=1}^{N}\left(\check{\boldsymbol{m}}_i - \frac{1}{N}\sum_{k=1}^{N}\check{\boldsymbol{m}}_k - \boldsymbol{R}_{\Delta\theta}\check{\boldsymbol{d}}_i + \frac{1}{N}\sum_{k=1}^{N}\boldsymbol{R}_{\Delta\theta}\check{\boldsymbol{d}}_k\right)\\ &= \sum_{i=1}^{N}\check{\boldsymbol{m}}_i - \underbrace{\frac{1}{N}\sum_{k=1}^{N}\sum_{i=1}^{N}\check{\boldsymbol{m}}_k}_{=\sum_{i=1}^{N}\check{\boldsymbol{m}}_i} - \sum_{i=1}^{N}\boldsymbol{R}_{\Delta\theta}\check{\boldsymbol{d}}_i + \underbrace{\frac{1}{N}\sum_{k=1}^{N}\sum_{i=1}^{N}\boldsymbol{R}_{\Delta\theta}\check{\boldsymbol{d}}_k}_{=\sum_{i=1}^{N}\boldsymbol{R}_{\Delta\theta}\check{\boldsymbol{d}}_i} = 0.\end{aligned}$$

The third term (4c) is minimal at $\Delta\tilde{\boldsymbol{t}} = 0$, i.e., $\Delta\boldsymbol{t} = \check{\boldsymbol{c}}_d - \boldsymbol{R}_{\Delta\theta}\check{\boldsymbol{c}}_d$. Therefore, $E^{(1)}$ is minimal iff $E^{(2)}$ is, with $E^{(2)}$ depending only on the rotation [20]:

$$E^{(2)}(\boldsymbol{R}_{\Delta\theta}, \Delta\boldsymbol{t}) = \sum_{i=1}^{N}\|\hat{\boldsymbol{m}}_i - (\boldsymbol{R}_{\Delta\theta}\hat{\boldsymbol{d}}_i)\|^2 = \sum_{i=1}^{N}\|\boldsymbol{m}_i\|^2 - 2\underbrace{\sum_{i=1}^{N}\hat{\boldsymbol{m}}_i \cdot \boldsymbol{R}_{\Delta\theta}\hat{\boldsymbol{d}}_i}_{\text{rotation term}} + \underbrace{\sum_{i=1}^{N}\|\boldsymbol{R}_{\Delta\theta}\hat{\boldsymbol{d}}_i\|^2}_{=\sum_{i=1}^{N}\|\hat{\boldsymbol{d}}_i\|^2}. \tag{5}$$

Since rotations are length preserving, in particular $\|\boldsymbol{R}_{\Delta\theta}\hat{\boldsymbol{d}}_j\|^2 \equiv \|\hat{\boldsymbol{d}}_j\|^2$, to minimize (5) it is sufficient to calculate the maximum of

$$E^{(3)}(\boldsymbol{R}_{\Delta\theta}) = \sum_{i=1}^{N}\hat{\boldsymbol{m}}_i \cdot \boldsymbol{R}_{\Delta\theta}\hat{\boldsymbol{d}}_i.$$

In order to maximize $E^{(3)}$, we compute a solution for $(\partial/\partial\Delta\theta)E^{(3)} = 0$:

$$\sum_{i=1}^{N}(-\sin\Delta\theta(\hat{m}_i^x \cdot \hat{d}_i^x + \hat{m}_i^y \cdot \hat{d}_i^y) - \cos\Delta\theta(\hat{m}_i^x \cdot \hat{d}_i^y + \hat{m}_i^y \cdot \hat{d}_i^x)) = 0.$$

The rotation is calculated as

$$\Delta\theta = \arctan\left(\frac{\sum_{i=1}^{N}(\hat{m}_i^x \cdot \hat{d}_i^x + \hat{m}_i^y \cdot \hat{d}_i^y)}{\sum_{i=1}^{N}(\hat{m}_i^y \cdot \hat{d}_i^x - \hat{m}_i^x \cdot \hat{d}_i^y)}\right). \tag{6}$$

With given rotation $\Delta\theta$, the translation is calculated as follows:

$$\underbrace{\begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix}}_{=\Delta \boldsymbol{t}} = \begin{pmatrix}\check{c}_m^x\\ \check{c}_m^y\end{pmatrix} - \underbrace{\begin{pmatrix}\cos\Delta\theta & \sin\Delta\theta\\ -\sin\Delta\theta & \cos\Delta\theta\end{pmatrix}}_{=\boldsymbol{R}_{\Delta\theta}} \cdot \begin{pmatrix}\check{c}_d^x\\ \check{c}_d^y\end{pmatrix}. \tag{7}$$

### *2.4. Dealing with featureless environments*

Driving in environments with no or few features is a challenge for every feature-based tracking algorithm. For calculating a transformation $\Delta\boldsymbol{p}$ between two scans as described, at least *two* features $\in \mathbb{R}^2$ are needed, since the four coordinates are sufficient to determine the three degrees of freedom (translation in *x*- and *y*-direction, rotation $\theta$); this theoretical result stays valid in practical experiments. However, the following situations where it is impossible to find two corresponding features may occur:

(1) *The two scans are too diverse to be matched*. Naturally, every relative localization algorithm is depending on an adequate overlap of two successive sensor inputs. In practice, however, this theoretical problem proved to be non-existent for the proposed algorithm. Since the mobile robot platform Kurt3D that was used to evaluate HAYAI drives with a maximum speed of 4.0 m/s, the scanner's frequency is 75 Hz, and the implemented algorithm reaches an average frequency of 2300 Hz on the robot's on-board computer (Pentium-III-800 MHz), it is guaranteed that *every* acquired scan is processed. Hence, the maximal difference between two scans is $\Delta t = 6.43$ cm (forward translation), respectively, $\Delta\theta = 25.58^\circ$ (rotation on the spot with maximal angular velocity). Various experiments in miscellaneous environments showed that the algorithm is capable of matching reliably much larger differences.

(2) *It is not possible to detect a sufficient number of matchable features in the scans*. This general problem of every feature-based scan matching technique may occur either if there are too few objects within the scanner's range, or the scenery is unsuitable to detect features. The first case is very unlikely, since the range of laser scanners is generally sufficiently high (in case of the model deployed here, 32 m), and the features used in this algorithm do exist frequently in a real world application. An example of the second case is an office environment, when the robot is turning at the end of a corridor and is directly facing a corner, detecting this corner as a sole feature. The probability of encountering such a situation is very low due to the type of detected features, but has to be considered in order to guarantee a stable localization technique. The following sections describe ways to cope with this exceptional case.

#### *2.4.1. Rotation estimation with angle histograms*

As a general solution for coping with featureless environments, we calculate a rotation estimation between two scans $\mathcal{S}$, $\mathcal{R}$ based on angle histograms [18,11] if the number of features is smaller than 2. Regarding the *N* single data points of a scan as vectors makes it possible to associate each data point with an angle, i.e., $\alpha(i)$ of point $i = (x_i, y_i)$ is defined as the angle between $x$-axis and the vector from $i$ to $i + 1$, with

$$\alpha(i) = \arctan\left(\frac{x_{i+1} - x_i}{y_{i+1} - y_i}\right).$$

Discretizing and plotting the angles against their frequency leads to a histogram as a characteristic of the scan that is invariant with regards to rotation, that is, a turning of the robot results in a translational displacement of the circular angle histogram (Fig. 5, right). Likewise, the shift between two histograms extracted from two different scans are used to calculate the rotation $\Delta\theta$ of the robot between taking these scans. More precisely, given two histograms $\mathcal{H}, \mathcal{G}$, the angle $\Delta\theta$ is computed by $\Delta\theta = \arg\max_i K_i(\mathcal{H}, \mathcal{G})$. The error function $K_i$ is used as a measurement of the similarity of the two scans. Weiß et al. suggest the use of a cross correlation function [18],

$$K_i(\mathcal{H}, \mathcal{G}) = \sum_{j=1}^{N} \mathcal{H}(j) \cdot \mathcal{G}((j + i) \bmod (n + 1)).$$

In practice, Gaussian noise of the sensor leads to small variances of the scanned data points that in turn imposes significant differences of the calculated angles. Therefore, we apply the reduction filter as described in Section 2.1 first, resulting in a much clearer histogram. Fig. 5 shows the results of both methods, as well as a histogram calculation based on lines extracted from the scan. For a description of the real-time capable line filter, see [21].

### *2.4.2. Dead reckoning information*

Fig. 6 shows the result of tracking the robot in a corridor, purely based on the scan matching algorithms. Though being considerably more precise than the reference methods, a problem occurs at the left end of the corridor, when the robot enters and turns in an area with very few features. While turning, the corner is detected as a sole feature, and the rotation is calculated by angle histograms as described above. Since every single histogram based scan matching induces a small error due to the implicit discretization, these errors can accumulate to a noticeable rotational error.

Minimizing the erroneous impact of the discretized angles is done by fusing the calculated pose change with odometry data by an extended Kalman filter (EKF) [22]. Due to the fact that both inputs are stochastically independent, the fusion leads to an improved trajectory (Fig. 6). In contrast to the empirically determined and fixed odometry covariance matrix, the corresponding HAYAI covariance matrix is computed for every scan matching individually. The calculation is done analogously to the heuristic formula proposed by Lu for evaluating an ICP based method and takes into account the number of data points used for the matching [23]. Thereby, the trajectory computed by HAYAI is unaltered in general situations, the odometry data is merely used for improving situations with few features. In fact, most of our experiments have been done without any fusion with odometry data.

Let $\check{M} = \{\check{\boldsymbol{m}}_i | i = 1, \ldots, N\}$, $\check{D} = \{\check{\boldsymbol{d}}_i | i = 1, \ldots, N\}$ be two sets of matched feature points and $\Delta\boldsymbol{p} = (\Delta\boldsymbol{t} = (\Delta t^x, \Delta t^y), \Delta\theta)$ the computed transformation. The scan matching's covariance matrix $\boldsymbol{\Sigma}_H$ can be estimated as $\boldsymbol{\Sigma}_H \approx s^2(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}$, with

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_1 \\ \vdots \\ \boldsymbol{A}_N \end{pmatrix}, \quad \boldsymbol{A}_i = \begin{pmatrix} 1 & 0 & -y_i \\ 0 & 1 & x_i \end{pmatrix}, \quad \begin{pmatrix} x_i \\ y_i \end{pmatrix} = \frac{1}{2}(\check{\boldsymbol{m}}_i - (\boldsymbol{R}_{\Delta\theta}(\check{\boldsymbol{d}}_i) + \Delta\boldsymbol{t})), \quad C = (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B},$$

$$\boldsymbol{B} = \begin{pmatrix} \check{\boldsymbol{m}}_1 - (\boldsymbol{R}_{\Delta\theta}(\check{\boldsymbol{d}}_1) + \Delta\boldsymbol{t}) \\ \vdots \\ \check{\boldsymbol{m}}_N - (\boldsymbol{R}_{\Delta\theta}(\check{\boldsymbol{d}}_N) + \Delta\boldsymbol{t}) \end{pmatrix}, \quad s^2 = \frac{(\boldsymbol{B} - \boldsymbol{A}\boldsymbol{C})^{\mathrm{T}}(\boldsymbol{B} - \boldsymbol{A}\boldsymbol{C})}{2N - 3}.$$

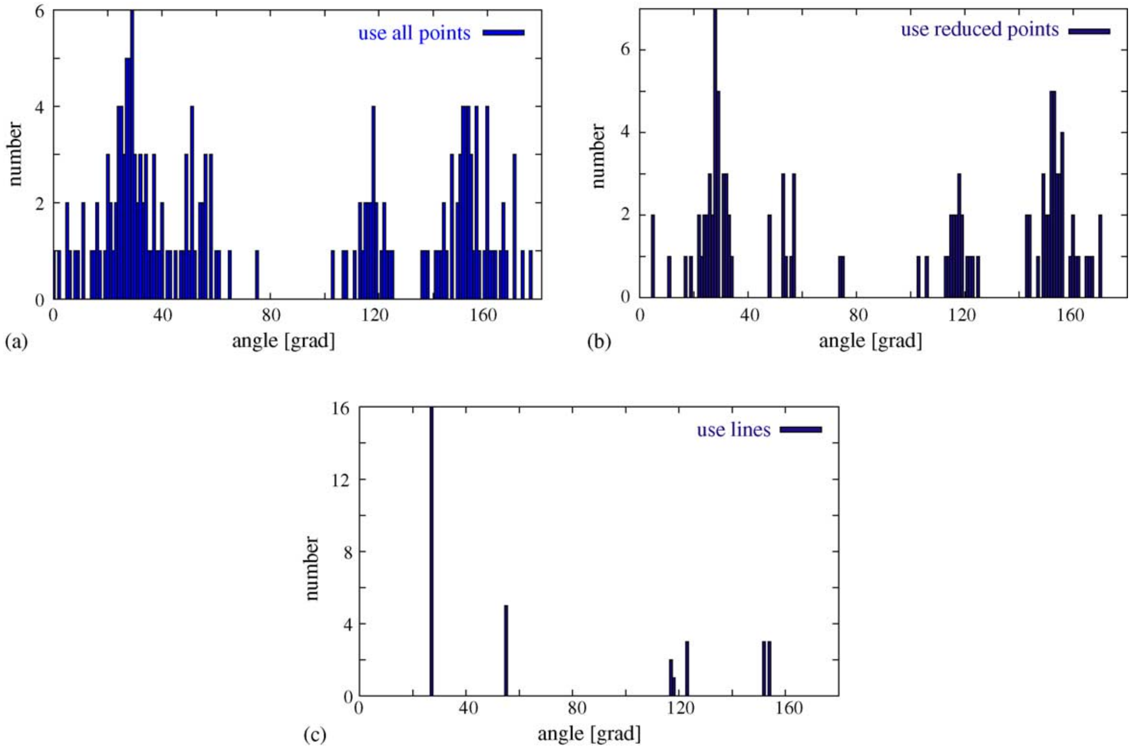

Fig. 5. Angle histograms. Left: due to data noise, the histogram degenerates. Middle: application of a reduction filter leads to clearer histograms. Right: using a line extraction filter results in clear, singular peaks.

### *2.4.3. Mastering corners*

Using angle histograms for a rotation estimation solves primary problems with missing features in general settings, as long as there are some planar surfaces within the scanner's range in order to detect significant peaks in surface orientations in the environment. However, in practical applications, scans with only one feature normally occur when maneuvering near a corner, e.g., at the end of a corridor, facing the corner directly. Such situations can be mastered by a different and computationally cheaper approach.

Let $\mathcal{S} = (\boldsymbol{s}_i)_{i\in\mathbb{N}}$ be the current scan, $\boldsymbol{s}_j \in \mathcal{S}$ the detected single feature point. In this given situation, the application of a line filter yields at least two line segments $L = (\boldsymbol{L}_S, \boldsymbol{L}_E)$ of sufficient length, adjacent to $\boldsymbol{s}_j$. Let $\boldsymbol{L}_S$ be the point nearer to $\boldsymbol{s}_j$. Defining a further feature point that lies on $L$ within a given distance $\mu$ from $\boldsymbol{s}_j$, i.e., $\boldsymbol{s}_k = \boldsymbol{s}_j + \mu(\boldsymbol{L}_E - \boldsymbol{L}_S)/\|\boldsymbol{L}_E - \boldsymbol{L}_S\|$ results in an extension of the feature set (Fig. 7). Applying the same method to the reference scan $\mathcal{R}$ yields a set $M$ of matched features with size $|M| \geq 2$ such that the unaltered application of HAYAI is possible, without the need of a prior rotation estimation. Fig. 8 shows the result of a pose tracking utilizing the described procedure. Without, the algorithm computes the trajectory presented in Fig. 6, inducing a rotational error when turning within the featureless area on the left side; see the next section for further explanations.

This principle of feature point generation is extensible for dealing with other kinds of featureless environments by defining further virtual landmarks in well-defined distances to previously extracted features, utilizing knowledge about the shape of the demanding environment (e.g., circular arcs).

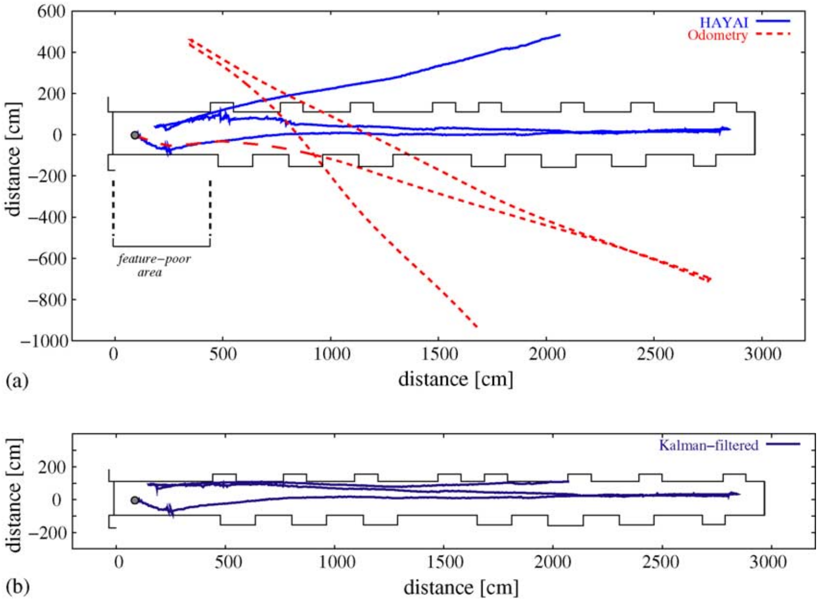

Fig. 6. Fusion of scan matching and odometry data. Top: comparison between odometry and HAYAI without the extension described in Section 2.4.3. Notice the rotational error induced by angle histograms that are occasionally utilized within the *feature-poor area*. Bottom: Kalman filtering of both input signals.

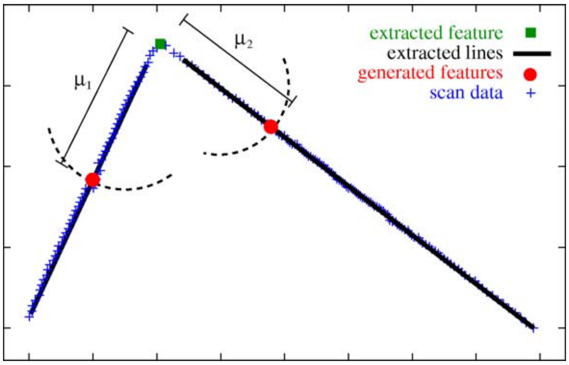

Fig. 7. Generation of feature points, lying on extracted lines within defined distances from the extracted feature.

## 3. Results

The described tracking algorithm has been tested in various indoor and outdoor environments. This paragraph demonstrates results of localization in different static and dynamic scenarios.

### 3.1. Indoor tracking

Fig. 9 shows the result of a 160 m cyclic drive in the office corridor of the Fraunhofer Institute AIS. The map was generated from 12,727 scans, gaged during the drive, and is the direct result of the scan matching algorithm, *without*

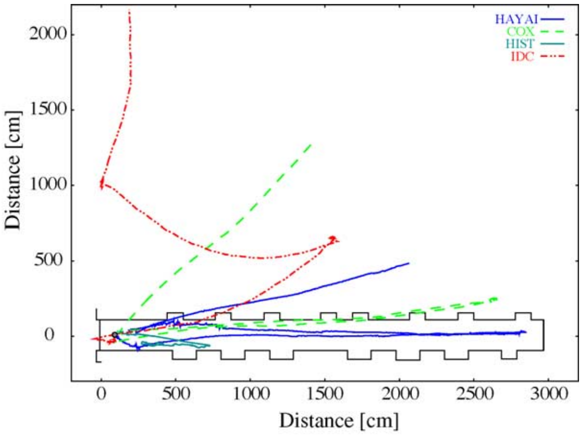

Fig. 8. Result of the pose tracking while driving in a 30 m long corridor (path: 3 m ×28 m, with 2° × 180° rotation at the respective end of the corridor, the starting position is marked with a circle). Plotted are the different trajectories computed by the described method as well as by the three reference scan matching algorithms (compare Fig. 14 for semi-ground-truth).

fusion with odometry, and without additional corrective algorithms like global relaxation or explicit correction of cycles, in order to illustrate the quality of the localization. In Fig. 9(bottom) the calculated trajectory is presented, as well as the results of other standard scan matching algorithms and the robot's odometry. Apparently, the reference algorithms are incapable of processing each incoming scan, since small errors in the calculations accumulate in such a way that the resulting trajectories are useless. Artificial off-line reduction of the scanner's frequency to 10 Hz, i.e., taking about every 7th scan only, improves the result of the comparative algorithms, but is no general solution, since this procedure fails at higher speed of the robot: With increasing velocity, the distance between two acquired scans increases as well, and soon becomes too large for a reliable matching.

Besides accuracy, another important criterion to evaluate the quality of a localization method is computation time. A slow algorithm imposes a hard restriction on the robot's velocity, or leads to periodical loss of acquired scans if they cannot be processed in real time. Skipping scans means to deal with larger pose changes between successive scans — if the robot drives fast enough, the scan matching will become unreliable due to a too small overlap, the pose will be highly incorrect or lost completely. Furthermore, even slower moving robots profit from a fast, efficient algorithm, since it enables the on-board computer to carry out other, computationally expensive tasks, besides the robot localization. Table 1 compares the times needed to calculate the trajectories from the indoor experiment described above, measured on a Pentium-III-800 MHz.

### 3.2. Outdoor tracking

Outdoor localization is particularly demanding for an algorithm, since it has to deal with highly irregular and non-static environments, where even small changes in the pose lead to significant changes in the corresponding scans [24,25]. For example, scans of plants result in varying output due to many jump edges and reflections on

Table 1
Speed comparison of scan matching algorithms: time (s) needed to calculate the trajectories from Fig. 9(bottom left) (160 m, 12,727 scans)

| | |
|---|---:|
| HAYAI | 5.41 |
| Cox | 30.86 |
| Hist | 213.26 |
| IDC | 47.09 |

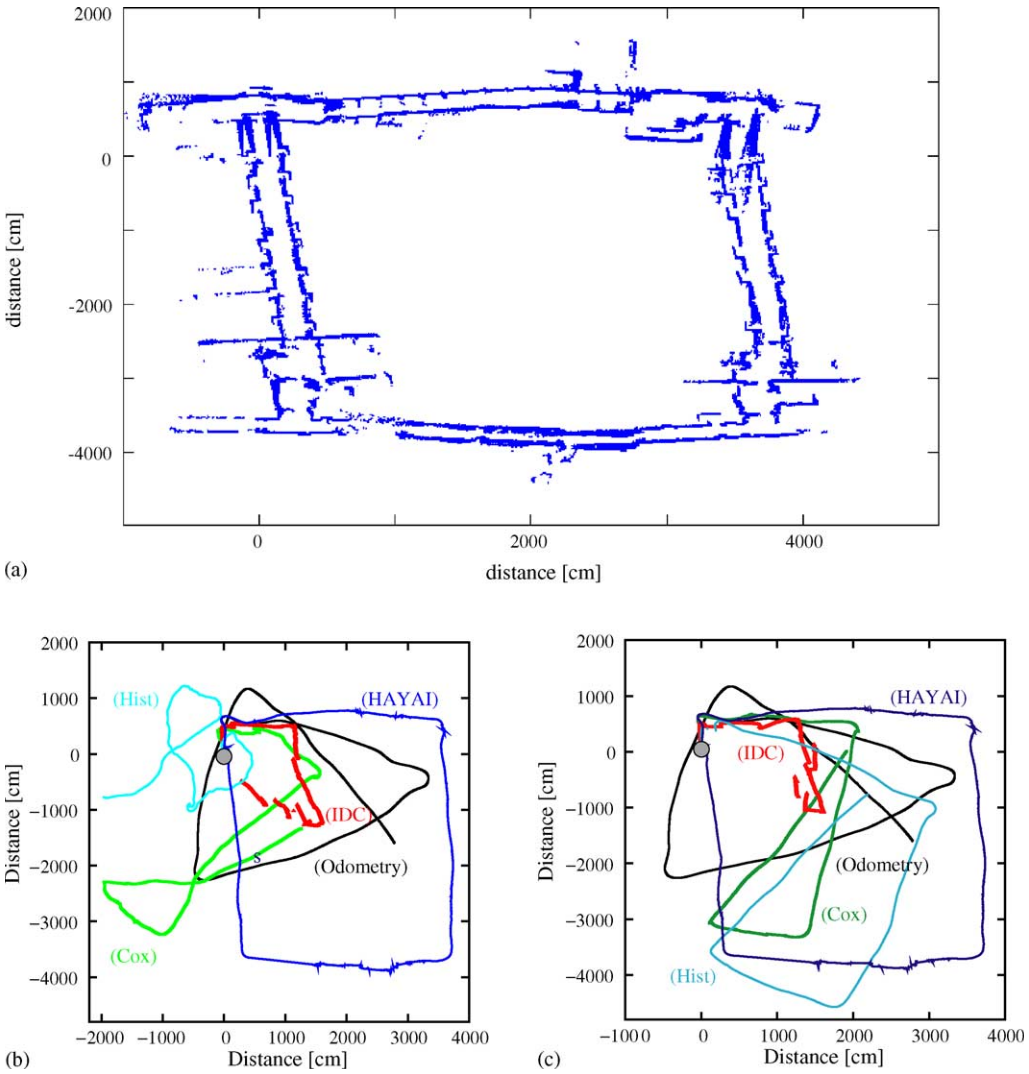

Fig. 9. Result of a 160 m closed-loop drive, inside an office building. Top: the resulting map, based on the scan matching algorithm only, *without* fusion with odometry or application of additional corrective algorithms. Bottom left: trajectories, calculated by different algorithms, based on all 12,727 scans (75 scans per second). Bottom right: artificial off-line reduction to 10 Hz enables the reference algorithms to improve their calculation.

moving leaves and branches. Other outdoor capable localization methods typically depend on some kind of absolute information, like an a priori map of the environment, active beacons or GPS data. Contrary to the reference scan matching algorithms, HAYAI proved to be able to localize the robot even in a clearly unstructured environment (Fig. 10), as long as the ground itself is roughly even so that there are no significant changes of the robot's pitch and roll angles. Fig. 10 does naturally not include ground-truth information, but provides a visual criterion to evaluate the trajectory's quality instead.

As an example for a practical application of the described localization method in combination with an overlying algorithm, Fig. 11 demonstrates a typical result of an autonomous three-dimensional model generation, suitable for

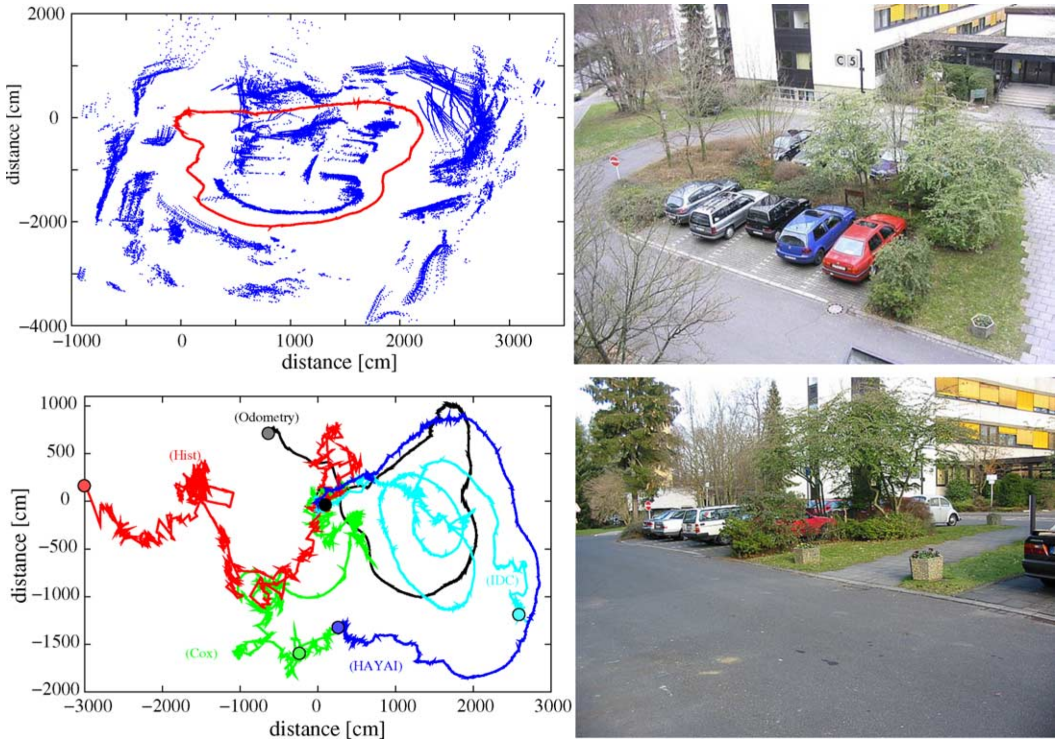

Fig. 10. Localization during a 84 m cyclic outdoor drive. Top left: map generated from 6903 scans, with calculated, exact, final trajectory (Kalman-filtered with odometry). Bottom left: calculated trajectories from HAYAI, other scan matching algorithms, and odometry. Right: photos of the scene.

indoor and outdoor environments. The robot is localized with HAYAI while driving between scan points. In discrete distances, the robot stops and acquires a 3D scan (duration: 6.8 s for 361 × 256 data points). These 3D scans are registered into one consistent model with the 6D-SLAM algorithm described in [1,2].[3]

### 3.3. Dynamic environments

In practice, the described algorithm proved to be robust even in dynamic environments, like scenarios with people walking by. Due to the scanner's high clock rate of 75 Hz, the environment does not change significantly between two scans. In addition, the use of abstracted features instead of whole scans and an accurate calculation of corresponding features yields an inherent robustness, since data points that appear suddenly and do not match correctly with the previous scan features are ignored. Fig. 12 shows a typical example of such an experiment. A comparison between the ground-truth map and the plotted scans shows that the accuracy of the localization is still very high and more precise than the reference algorithms' results even in strictly static scenarios (compare Fig. 8). An explicit detection of dynamic objects as described in [26,27] was not necessary for a stable tracking.

[3] A video of a virtual flight through the scene can be found at: http://www.ais.fhg.de/ARC/kurt3D/HAYAI/videos/localization_outdoor.divx.avi.

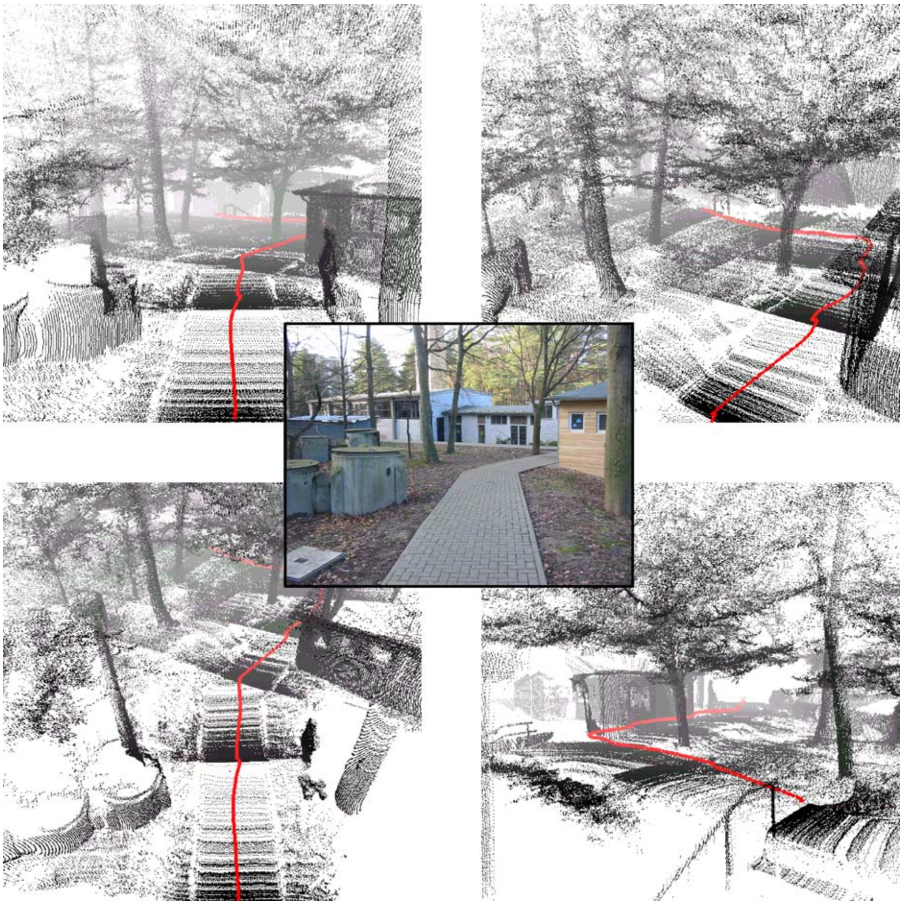

Fig. 11. Outdoor 3D model generation. Localization with HAYAI while driving serves as an essential initial pose estimation for the scan registration, carried out with a 6D-SLAM algorithm. The figure shows four different views of the final 3D model (the calculated trajectory is plotted as a line), as well as a photo of the real scene.

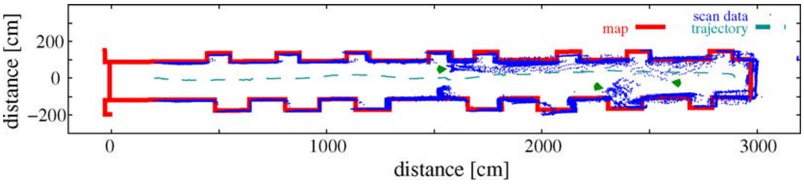

Fig. 12. Left: pose tracking in dynamic environments. The red plotted map serves as ground-truth, the calculated trajectory is based on scan matching only, without fusion with odometry. Green arrows mark scan data of dynamic objects (people walking around). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

## 4. Absolute localization

Designed as a method for *relative* localization, the described algorithm is inherently independent of an a priori given map. However, if global, absolute information in the form of a model of the environment is available, HAYAI is capable of utilizing it successfully for re-localizing the robot within the map. This way, relative localization's intrinsic problem of error accumulation is solved. The described scan matching algorithm can be used without any changes. This section describes the explicit procedure for absolute localization and concludes with a comparison between results computed by HAYAI as well as the three reference algorithms mentioned above, when utilized in the same manner. Consequentially, comparison with algorithms that inherently depend on a mandatory prior map [28,29] is omitted.

The pose is tracked as described above. In addition, each calculated pose change is simultaneously applied to a virtual pose of the robot within the map. That is, given an initial absolute pose, the robot's trajectory is traced both locally, relative to the starting pose, and globally, inside the map. At discrete steps, depending on the distance, a scan is simulated at the virtual pose. Scan simulation is done by generating virtual rays at discrete angles (here: 1° steps for an apex angle of 180°), coming from a source at the specified pose of the scanner within a map given as line model. The first intersection between such a ray and a map line is returned as a scan data point iff the Euclidean distance between source and point is smaller than a given threshold (e.g., 8 m).

Differences between the real and simulated scans are due to accumulated errors and sensor noise. This comparison is done with HAYAI as well, the virtual pose is corrected by the calculated transformation. Thereafter, the real world pose is corrected based on Eq. (8). Let $\boldsymbol{p}_n^{\mathfrak{M}}$ be the current pose within the map, $\boldsymbol{p}_0^{\mathfrak{M}} = (x_0^{\mathfrak{M}}, y_0^{\mathfrak{M}}, \theta_0^{\mathfrak{M}})$ the initial

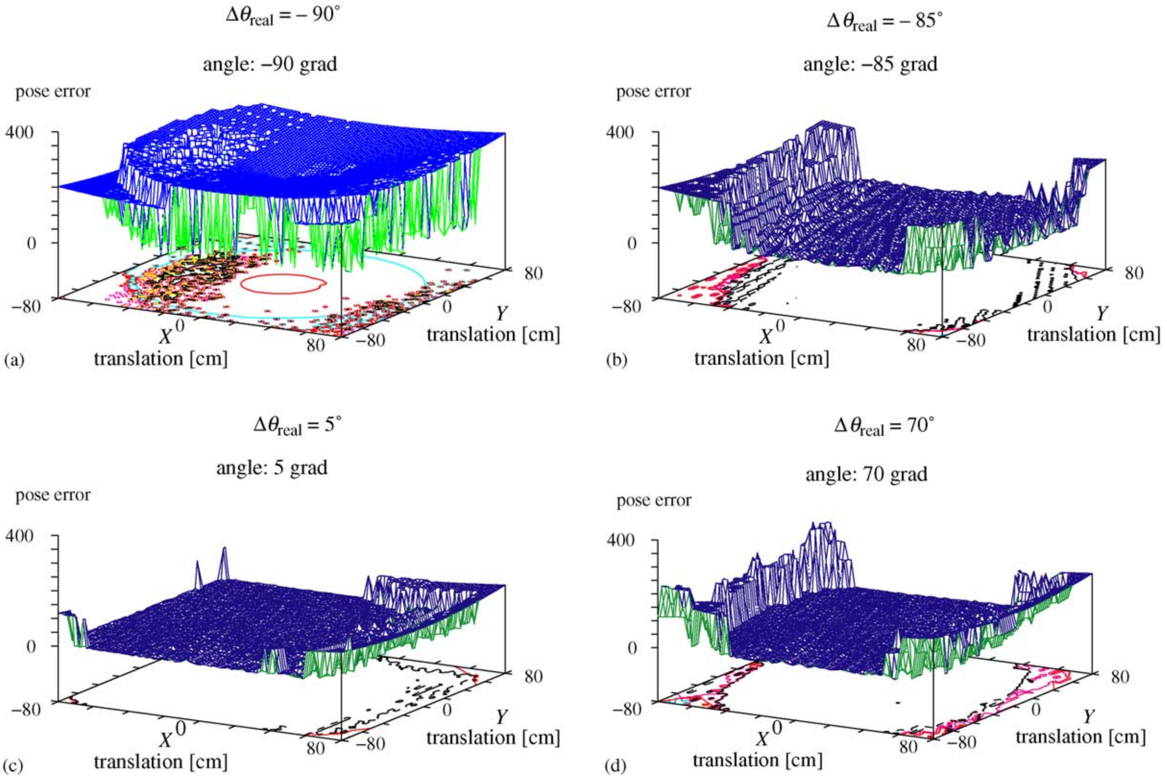

Fig. 13. Evaluation of the scan matching's limits, by the example of an office corridor. Given a fixed rotation, here $\Delta\theta_{\text{real}} \in \{-90^\circ, -85^\circ, 5^\circ, 70^\circ\}$, the calculation error $e = \sqrt{(\Delta x_{\text{real}} - \Delta x)^2 + (\Delta y_{\text{real}} - \Delta y)^2} + |\Delta\theta_{\text{real}} - \Delta\theta|$ is plotted between each two scans with a real transformation of $\boldsymbol{\Delta p}_{\text{real}} = (\Delta x_{\text{real}}, \Delta y_{\text{real}}, \Delta\theta_{\text{real}})$ and a calculated transformation of $\boldsymbol{\Delta p} = (\Delta x, \Delta y, \Delta\theta)$.

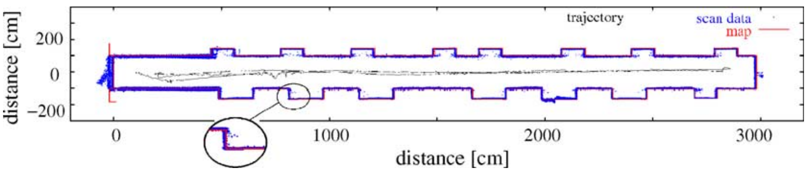

Fig. 14. Re-localization in a given map. Scan matching between real and simulated scans enables a precise absolute localization. Comparison between scan points — transformed by the computed respective pose — and the ground-truth map visualizes the precision of the trajectory calculated by HAYAI. The zoomed view highlights the exact match as well as data noise.

absolute pose, and a rotation matrix $\boldsymbol{R}_\theta$,

$$\boldsymbol{R}_\theta = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Then, the current pose $\boldsymbol{p}_n$ is corrected by

$$\boldsymbol{p}_n = \boldsymbol{R}^{-1}_{\theta_0^{\mathfrak{M}}} \boldsymbol{p}_n^{\mathfrak{M}} - \boldsymbol{R}^{-1}_{\theta_0^{\mathfrak{M}}} \boldsymbol{p}_0^{\mathfrak{M}}. \tag{8}$$

Naturally, the quality of a map-based localization depends on the correctness of the initial absolute pose $\boldsymbol{p}_0^{\mathfrak{M}}$. However, the described algorithm is capable of correcting uncertainties in $\boldsymbol{p}_0^{\mathfrak{M}}$ up to a certain degree: Initially, before moving the robot, a scan comparison between $\boldsymbol{p}_0^{\mathfrak{M}}$ and $\boldsymbol{p}_0$, e.g., the first real world laser scan, should ideally result in a zero transformation $\Delta\boldsymbol{p} = (0, 0, 0)$. Any other calculation result denotes an error in the initial absolute pose and is used to adjust it, $\boldsymbol{p}_0^{\mathfrak{M}} \underset{\Delta\boldsymbol{p}}{\rightarrow} \boldsymbol{p}_0^{\mathfrak{M}}$. Fig. 13 demonstrates that in a standard environment, adjustments in the magnitude of ±80 cm and ±70° are possible.[4] In fact, these are the same differences that two scans may differ from each other in order to be matched correctly when tracking the pose locally.

Fig. 14 shows the trajectory calculated with HAYAI when utilized for absolute localization. To evaluate the trajectory, each acquired scan is plotted into a common coordinate system, transformed by the respective pose. Comparison between scan data and the a priori map shows the correctness of the computed pose. Figs. 15–17 demonstrate the reference algorithms' results, Table 2 presents the times needed by all four algorithms. It is noticeable that only the IDC algorithm, in spite of its unsatisfactory result at relative localization (see Fig. 8), leads — given enough time — to a result nearly as good as HAYAI. This result suggests that the IDC algorithm is in principle suitable for tracking the robot in this kind of environment, but has significant problems with the accumulation of small errors that occur at each and every scan matching. This problems is normally reduced by utilizing external information, like a map or odometry information. However, even in case of absolute localization, accuracy and calculation speed of HAYAI are superior. The other two algorithms proved to be unsuitable for the setting demonstrated in this section.

Re-localizing in a consistent, static model solves the inherent problem of any relative localization technique, e.g., boundless error accumulation, at the expense of restricting the robot to already mapped environments. Furthermore, a new potential source of disturbance is induced by discrepancies between map and reality. However, practical experiments show that with an adequately correct map, HAYAI achieves a stable, highly precise and still very fast localization.

[4] Hence, the algorithm is in principle able to locate a robot driving with a velocity of 60 m/s, being limited by the laser scanner's speed; if the scan data could be acquired as fast as HAYAI can process them, i.e. at a rate of 2300 Hz, a theoretical velocity of 1.7 km/s would be manageable in the given setting, on a Pentium-III-800 MHz.

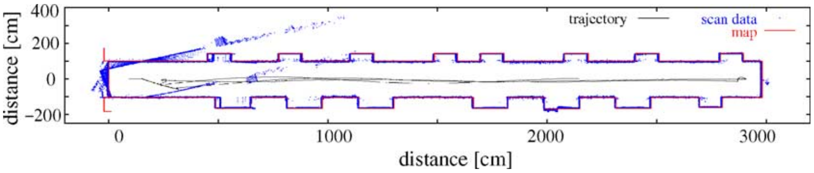

Fig. 15. Re-localization in a given map. Trajectory calculated by IDC.

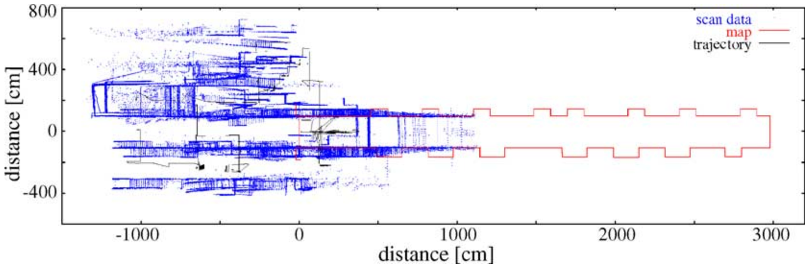

Fig. 16. Re-localization in a given map. Trajectory calculated by Hist.

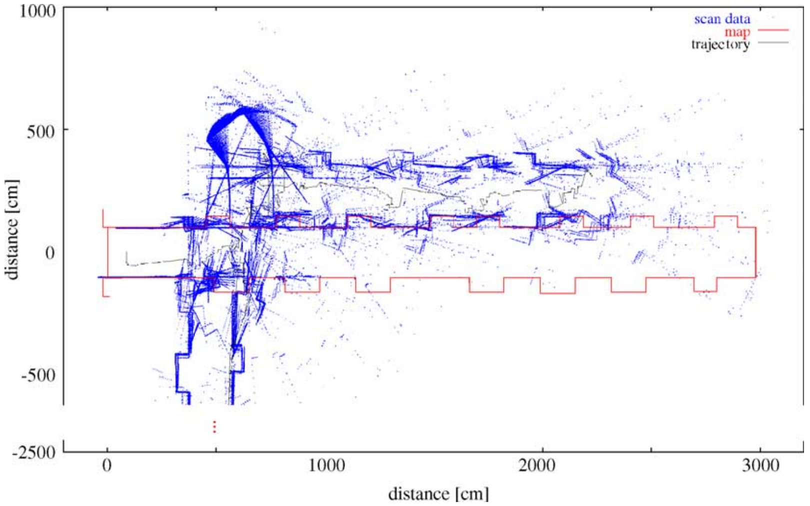

Fig. 17. Re-localization in a given map. Trajectory calculated by Cox.

Table 2
Speed comparison of scan matching algorithms, utilized for *absolute* localization: time (s) needed for computing the trajectories from Figs. 14–16 (84 m, 2844 scans), including time for scan simulation

| | |
|---|---|
| HAYAI | 10.38 |
| Cox | 79.91 |
| Hist | 338.29 |
| IDC | 101.61 |

## 5. Conclusion

Mobile robots depend on a stable and reliable self-localization. The described, newly developed localization algorithm, based on pairwise scan matching, is capable of tracking a robot's trajectory in indoor as well as outdoor environments, driving with a speed of up to 4 m/s. Furthermore, given a map of the environment, the algorithm can be used for stable absolute localization as well. Comparison with three state of the art scan matching algorithms showed the outstanding performance of the proposed method in terms of precision and computation speed. HAYAI is able to process scans with up to 2300 Hz on a moderate laptop (Pentium-III-800 MHz) — if only the scanner supplied the data so fast. In reality, the scanner provides the data with 75 Hz, that is one scan every 5.3 cm at a speed of 4 m/s. The scan matching algorithm HAYAI is based on matching features that are naturally given in any standard operational environment and are extracted efficiently in the order of $\mathcal{O}(n)$, given scans of $n$ points. The method is not restricted to polygonal environments, independent from structural alteration of the surrounding like beacons or artificial landmarks, and proved to be stable even in dynamic environments. The presented method succeeded in providing our 6D-SLAM algorithm [1,2] with a required pose estimation and enables the system to build three-dimensional indoor and outdoor maps.

## References

[1] A. Nüchter, H. Surmann, K. Lingemann, J. Hertzberg, S. Thrun, 6D-SLAM with an application in autonomous mine mapping, in: Proceedings of the International Conference on Robotics and Automation (ICRA'04), New Orleans, USA, 2004, pp. 1998–2003.

[2] H. Surmann, A. Nüchter, K. Lingemann, J. Hertzberg, 6D-SLAM — preliminary report on closing the loop in six dimensions, in: Proceedings of the 5th Symposium on Intelligent Autonomous Vehicles (IAV'04), Lisbon, Portugal, 2004.

[3] S. Thrun, D. Fox, W. Burgard, A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'00), San Francisco, CA, USA, 2000, pp. 321–328.

[4] F. Lu, E. Milios, Globally consistent range scan alignment for environment mapping, Auton. Robots 4 (1997) 333–349.

[5] K. Lingemann, A. Nüchter, J. Hertzberg, H. Surmann, About the control of high speed mobile indoor robots, in: Proceedings of the European Conference on Mobile Robots (ECMR'05), 2005, submitted for publication.

[6] Z. Xiang, V.M.F. Santos, J. Liu, Robust mobile robot localization by fusing laser and sonar, in: Proceedings of the 11th IEEE International Conference on Advanced Robotics (ICAR'03), Coimbra, Portugal, 2003, pp. 276–280.

[7] S. Se, D. Lowe, J. Little, Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks, Int. J. Robot. Res. 21 (2002) 735–758.

[8] C. Cauchois, E. Brassart, L. Delahoche, C. Drocourt, Spatial localization method with omnidirectional vision, in: Proceedings of the 11th IEEE International Conference on Advanced Robotics (ICAR'03), Coimbra, Portugal, 2003, pp. 287–292.

[9] S. Pfister, K. Kriechbaum, S. Roumeliotis, J. Burdick, Weighted range sensor matching algorithms for mobile robot displacement estimation, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'02), 2002, pp. 667–1674.

[10] J.-H. Moon, B.-J. You, H. Kim, S.-R. Oh, Fast Markov localization using angle-histogram, in: Proceedings of the 11th IEEE International Conference on Advanced Robotics (ICAR'03), Coimbra, Portugal, 2003, pp. 411–416.

[11] T. Röfer, Building consistent laser scan maps, in: Proceedings of the 4th European Workshop on Advanced Mobile Robots (EUROBOT'01), vol. 86, 2001, pp. 83–90.

[12] K.J. Kyriakopoulos, N. Skounakis, Moving obstacle detection for a skid-steered vehicle endowed with a single 2D laser scanner, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'03), Taipei, Taiwan, 2003, pp. 7–12.

[13] E. Santiso, M. Mazo, J. Ureña, J.A. Jiménez, J.J. García, M.C. Serra, Mobile robot positioning with natural landmark, in: Proceedings of the 11th IEEE International Conference on Advanced Robotics (ICAR'03), Coimbra, Portugal, 2003, pp. 47–52.

[14] J.V. Bitterling, S. Küpper, B. Mertsching, Incremental feature-based pose tracking for a mobile robot, in: Proceedings of the 11th IEEE International Conference on Advanced Robotics (ICAR'03), Coimbra, Portugal, 2003, pp. 1038–1043.

[15] C.S.J.-S. Gutmann, Amos: comparison of scan matching approaches for self-localization in indoor environments, in: Proceedings of the 11th European Workshop on Advanced Mobile Robots (EUROBOT'96), Kaiserslautern, Germany, 1996, pp. 61–67.

[16] F. Lu, E. Milios, Robot pose estimation in unknown environments by matching 2D range scans, in: Proceedings of the IEEE Computer Vision and Pattern Recognition Conference (CVPR'94), 1994, pp. 935–938.

[17] I.J. Cox, Blanche — an experiment in guidance and navigation of an autonomous robot vehicle, J. IEEE Trans. Robot. Automat. (TRA'91) 7 (2) (1991) 193–204.

[18] G. Weiß, C. Wetzler, E. von Puttkamer, Keeping track of position and orientation of moving indoor systems by correlation of range-finder scans, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94), Munich, Germany, 1994, pp. 595–601.
[19] T. Einsele, Localization in indoor environments using a panoramic laser range finder, Ph.D. Thesis, Technische Universität München, Germany, 2001.
[20] B. Horn, Closed-form solution of absolute orientation using unit quaternions, J. Opt. Soc. Am. A 4 (4) (1987) 629–642.
[21] H. Surmann, K. Lingemann, A. Nüchter, J. Hertzberg, Fast acquiring and analysis of three-dimensional laser range data, in: Proceedings of the 6th International Fall Workshop Vision, Modeling, and Visualization (VMV'01), Stuttgart, Germany, 2001, pp. 59–66.
[22] G. Welch, G. Bishop, An introduction to the Kalman filter, Technical Report No. TR 95-041, Department of Computer Science, University of North Carolina, March 2003.
[23] F. Lu, Shape registration using optimization for mobile robot navigation, Master's Thesis, University of Toronto, 1995.
[24] J. Folkesson, H. Christensen, Outdoor exploration and SLAM using a compressed filter, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'03), Taipei, Taiwan, 2003, pp. 419–426.
[25] A. Lankenau, T. Röfer, Mobile robot self-localization in large-scale environments, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'02), San Francisco, CA, USA, 2002, pp. 1359–1364.
[26] D. Hähnel, R. Triebel, W. Burgard, S. Thrun, Map building with mobile robots in dynamic environments, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'03), 2003, pp. 1557–1563.
[27] C.-C. Wang, C. Thorpe, S. Thrun, Online simultaneous localization and mapping with detection and tracking of moving objects: theory and results from a ground vehicle in crowded urban areas, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'03), Taipei, Taiwan, 2003, pp. 842–849.
[28] S. Thrun, D. Fox, W. Burgard, F. Dellaert, Robust Monte Carlo localization for mobile robots, Artif. Intell. 128 (1–2) (2001) 99–141.
[29] J.S. Gutmann, Markov–Kalman localization for mobile robots, in: Proceedings of the IEEE International Conference on Pattern Recognition (ICPR'02), Quebec, Canada, 2002.

**Kai Lingemann** is a research associate and Ph.D. student at University of Osnabrück. Past affiliations were with the Fraunhofer Institute for Autonomous Intelligent Systems (AIS, Sankt Augustin, Germany), University of Bonn (Germany) and Kyoto University (Japan). He received the diploma degree in computer science from University of Bonn in Januar 2004. His research interests include mobile robot localization, reliable robot control, 3D vision, and laser scanning technologies.

**Andreas Nüchter** is a research associate at University of Osnabrück and a Ph.D. student at University of Bonn. Past affiliations were with the Fraunhofer Institute for Autonomous Intelligent Systems (AIS, Sankt Augustin) and University of Bonn. He received the diploma degree in computer science from University of Bonn in 2002 and was awarded with the best paper award by the German society of informatics (GI) for his thesis. His research interests include reliable robot control, 3D vision, and laser scanning technologies. He is a member of the GI and the IEEE.

**Joachim Hertzberg** is a Full Professor for computer science at the University of Osnabrück, where he is heading the Knowledge-Based Systems research group. He holds a diploma and a doctorate degree in computer science, both from the University of Bonn. Past affiliations and stays were with the Fraunhofer Institute for Autonomous Intelligent Systems (AIS, Sankt Augustin), with the Universities of Bonn (Germany), Dortmund (Germany), and Auckland (New Zealand), and with the International Computer Science Institute (ICSI), Berkeley, CA. His area of research is artificial intelligence, where he has contributed to action planning, robot localization and mapping, plan-based robot control, robot control architectures, temporal reasoning, logical reasoning about action and change, constraint-based reasoning, and applications of these techniques. He is currently a member of the Executive Council of the International Conference on Automated Planning and Scheduling (ICAPS).

**Hartmut Surmann** heads a research group within the Fraunhofer Institute for Autonomous Intelligent Systems (AIS). He received his diploma in computer science from the Department of Computer Science, University of Dortmund and his Ph.D. in electrical engineering from the Faculty of Electrical Engineering, University of Dortmund, Germany, in 1989 and 1994, respectively. His primary research interests are autonomous robotics and computational intelligence. He has published several papers in the area of design, architecture, hardware and application of robotics and fuzzy logic controllers. He is a member of the GI and VDE and received several awards, e.g., the FUZZ-IEEE/IFES'95 robot intelligence award and the Ph.D. award for his thesis from the German AI institutes in 1996. He gives lectures in control theory and robotics at FH Bonn-Rhein-Sieg.