

Route Navigation Using Motion Analysis

Thomas Röfer

Bremer Institut für Sichere Systeme, TZI, FB 3,
Universität Bremen, Postfach 330440, 28334 Bremen, Germany
Phone +49 421 218 4659, Fax +49 421 218 3054
`roefer@tzi.de`

Abstract. This paper presents a simple approach for the acquisition and representation of spatial knowledge needed for controlling a semi-autonomous wheelchair. Simplicity is required in the domain of rehabilitation robotics because typical users of assistive technology are persons with severe impairments who are not technical experts. The approach proposed is a combination of carrying out so-called *basic behaviors* and the analysis of the wheelchair's *track of motion* when performing these behaviors. As a result, autonomous navigation in the user's apartment or place of work can be learned by the wheelchair by teaching single routes between potential target locations.

This paper focuses on the analysis of the motion tracks recorded by the vehicle's dead reckoning system. As a means for unveiling the structure of the environment while the system is moving, an *incremental generalization* is applied to the motion tracks. In addition, it is discussed how two of such generalized motion tracks are matched to perform a one-dimensional self-localization along the route that is followed.

Keywords. Route Learning, Navigation, Motion Tracks, Generalization, Robotics, Rehabilitation

1 Introduction

During the recent decade, the research in the field of robotics has evolved from the development of stationary manipulators to autonomous mobile systems that can be employed as *service robots*. It can be assumed that service robots will revolve many areas of private and business life the same way as the stationary robots have changed the production process in factories. The basic skill of a mobile system is *navigation*, i. e. moving from a start position to a target position. Navigation is normally a combination of two different tasks: a navigator has to determine his current position and he has to plan and execute at least the next step on his way to the goal.

In an industrial environment, a mobile system is often designed as part of the factory. Thus, the environment was constructed in a way that supports the movement of the autonomous system, e. g., guidance marks were installed to support the system's self-localization, and the robot has a map of the environment

with the information relevant for motion planning. Such prerequisites are hard to realize in office and private environments in which people live and work, because these places are often subject to fluent changes, and they are not prepared for the use of robots. Therefore, navigation techniques used in industrial applications cannot directly be adopted for service robots. Instead, other methods have to be developed that are able to deal with unknown and partially dynamic environments to fulfill the needs in service robotics.

In addition, service robots are normally surrounded by persons that are not technical experts. Laymen and laywomen often associate artificial autonomous systems with natural ones, and therefore they have the expectation that these systems behave similarly. Hence, to gain a high acceptance for service robots, they must act in a predictable way. In addition, and in contrast to biological systems, a service robot must be *safe*, i. e., it must be guaranteed that it will not do any harm to persons, animals, or objects in the environment.

The application domain addressed by the author's research group is the area of *rehabilitation robotics*. Due to the changing distribution of ages in future populations, more and more elder people, in addition to handicapped persons, will be forced to maintain their mobility with the help of rehabilitation technology. The dominating factor in this field are the additional costs that result from the use of assistive technology. Therefore, it is required that the enhancements are cheap, e. g. an "intelligent" wheelchair should cost less than \$1000,- more than a normal power wheelchair. This avoids the use of expensive equipment such as laser scanners.

2 Acquisition and Representation of Spatial Knowledge

An autonomous mobile system needs spatial knowledge as basis for any form of navigation. In service robotics in general and in rehabilitation robotics in particular, this spatial knowledge must be easy to acquire, because usually, the system has to be set-up when it is delivered. For example, it would not be acceptable if the delivery man or woman must draw up a precise metrical map of the customer's apartment, including all furniture, etc, without any technical assistance. Instead, the autonomous system should support the service technician in obtaining the spatial knowledge employing its sensory equipment.

A very common approach is to represent spatial knowledge as *maps*. There are several types of maps that were used in robotics. Most approaches use some kind of metrical grid map. These maps normally consist of a homogeneous grid of cells, but also adaptive, heterogeneous grids have been realized (Kollmann *et al.*, 1997). In two-dimensional *obstacle maps* (e. g. in Jörg *et al.*, 1993), each cell encodes the probability of the presence of an obstacle at the corresponding (x, y) location in the scene. In three-dimensional *configuration maps* (e. g. in Hoyer *et al.*, 1994), each cell states whether the robot can occupy the corresponding (x, y, θ) position (including the orientation of the robot). Configuration maps can easily be used to implement path planning, but known algorithms are very time-consuming (Hoyer *et al.*, 1994). In combination with stochastic sensor models, this type of

map can be employed for the realization of a robust self-localization technique (Burgard *et al.*, 1997).

Mojaev and Zell (1998) developed a system that generates a metrical 2-D grid map while the robot is moving, i.e. on the fly. In their approach, either sonar readings or laser scans are inserted into small local probabilistic grid maps that are then integrated into a single global map. The integration is performed by searching for the greatest similarity between overlapping regions of the new local map and the existing global map. The search space is limited to the neighborhood of the estimated position of the local map. The estimation is based on the odometry readings of the system. Thus, a map can be generated by either manually driving the robot around in the scene, or by letting it explore the environment by itself. The major disadvantage of this approach is that the map only states where obstacles are and where are none, but not, where usable connections between positions are. Therefore, navigation requires the usage of a path planning algorithm, which can generate high computational costs if the map has a lot of cells.

In contrast to grid based methods, Gutmann and Nebel (1997) used laser scans to measure and represent the environment. Again, in their approach the robot has to be controlled on an arbitrary route through the scene. During this drive, it takes laser scans of its environment and stores them together with a dead reckoned position. Afterwards, these scans are integrated into a single, consistent map of scan points, and in addition, a topological structure of the environment is generated based on the assumption that positions are neighbored where the scans taken have many scan points in common. However, the laser scans, i.e. the metrical information, is preserved; it is required for the self-localization of the system. This approach has two drawbacks: On the one hand and in contrast to the method of Mojaev and Zell (1998), the map is generated offline, i.e. the whole approach does not work in real-time. On the other hand, it is not guaranteed that all connections in the topological map can really be traveled by the robot. Therefore, the connections have to be verified by trying to pass through each of them. However, it would be acceptable to do this during the normal operation of the system, i.e. after the learning phase.

Franz *et al.* (1997) developed a method that generates a topological *view map* based on one-dimensional panoramic images, i.e. *views*. The special thing about their approach is that the views recorded do not only mark the nodes in the topological structure, but they are also the means to get from one position to a neighboring one. The major weakness of their technique is that each view has to be unique, and thus the method is not able to map arbitrarily large environments.

The major feature of a representation of spatial knowledge in maps is that a path can be planned from any start position to any target location. However, to allow a user to select start and end of a route, a graphical presentation of the map is required. If this is not possible, e.g., because a screen to display this information is too expensive, maps lose their major advantage over topological representations. So if navigation should be realized at low costs, *route learning* is

an interesting alternative, because routes can be presented to the user as simple texts, e.g. “from desk to printer.” This allows the implementation of a variety of input methods, e.g., the user can pick up a route from a list, or he or she can select one by speech input.

3 The Bremen Autonomous Wheelchair

In the *Bremen Autonomous Wheelchair* project, a system is developed that realizes several levels of support: from a safe collision avoidance (Lankenau and Meyer, 1997) to an automatic speed control (Lankenau *et al.*, 1998) to obstacle avoidance (Röfer and Lankenau, 1998) to several basic behaviors such as wall-following or turning round. Currently, the highest level of support is the autonomous driving of pre-taught routes in networks of passages. As was described by Krieg-Brückner *et al.* (1998), routes can be represented as sequences of basic behaviors and so-called *routemarks* that trigger the switching between the behaviors. To satisfy the demands for simplicity in this domain, these combinations are learned during teaching drives—one for each route—that are controlled, e.g., by the salesperson or the wheelchair mechanic. After that, the wheelchair is able to travel routes autonomously. This will support users suffering from severe impairments who are currently not able to control a power wheelchair, e.g. blind people. Röfer and Müller (1998) used a camera to detect artificial routemarks. However, this approach had two drawbacks: on the one hand, the environment has to be prepared to use this method, and on the other hand, the wheelchair has to be equipped with a camera on a turntable. As this is an expensive solution, it is not preferable for the health care domain. Therefore, a more cost-effective approach has been chosen to find points of reference along the routes. It only uses the sensory equipment that is already required by the other modules on the wheelchair. The method developed analyzes the wheelchair’s motion to represent the course of the taught routes and it uses this representation for two purposes: to decide whether the system is still on the right track, and to trigger the switching between the basic behaviors.

4 The Robotics Platforms

The navigation approach presented in this paper was developed and tested on three different systems: the first prototype of the Bremen Autonomous Wheelchair, the current version “Rolland,” and a Nomad 200 owned by the Computer Science Department in the University of Manchester:

The first prototype of the Bremen Autonomous Wheelchair (cf. Fig. 1a) is a non-holonomic vehicle that is driven by its front wheels and steered by its back wheels (Krieg-Brückner *et al.*, 1998). The system is equipped with twelve bumpers, six infrared sensors, 16 ultrasonic sensors and a camera. Due to two wheel-encoders, the wheelchair can measure the rotations of its front wheels. Thus, it is able to perform dead reckoning.

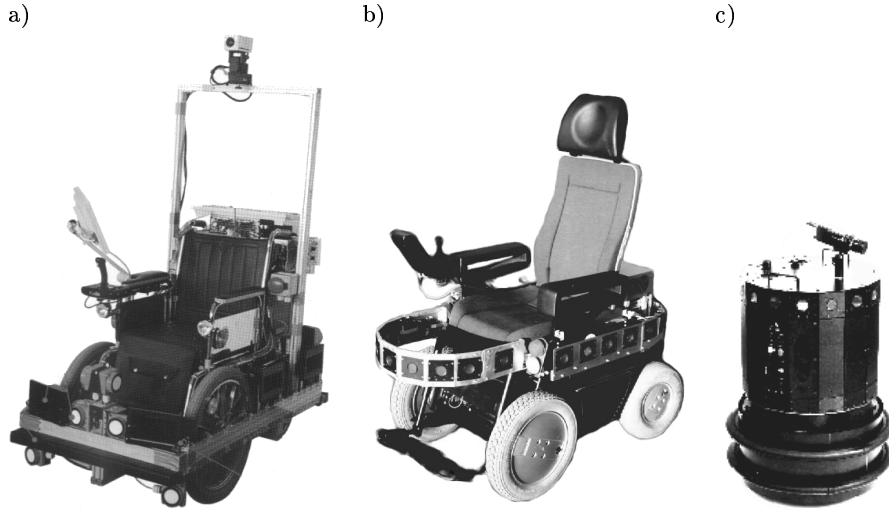


Fig. 1. The robotics platforms. a) The first prototype of the Bremen Autonomous Wheelchair. b) The second prototype “Rolland.” c) The Nomad 200.

Rolland. The basis for the actual Bremen Autonomous Wheelchair (cf. Fig. 1b) is a commercial power wheelchair manufactured by the German company Meyra (cf. Fig. 1). Its kinematics are quite similar to the ones of the first prototype. However, the new wheelchair is faster, can drive narrower curves, and its sonar sensors are distributed better around the system. The maximum speed currently used is 84 cm/s. The wheelchair makes use of 27 Polaroid sonar sensors with an opening angle of 25° that are mounted around the system and a PC placed behind the seat that is running the real-time operating system QNX. Even the original Meyra wheelchair is able to measure its actual speed and its steering radius as standard, and thus “Rolland” performs dead reckoning. However, curves are not measured very precisely.

The Nomad 200 (cf. Fig. 1c) is a cylindrical robot that can turn on the spot. Its maximum speed is 50 cm/s. It is equipped with 16 sonar sensors that are equal to the ones used on Rolland. Nomad’s mechanics are optimized for a precise odometry. However, over longer distances, the dead reckoning readings also drift away (Owen and Nehmzow, 1997). The Nomad 200 in Manchester uses only a 486/80 as host computer that is significantly slower than the Pentium PCs on the wheelchairs.

5 Navigation Approach

Odometry (dead reckoning) is easy to realize but is also known for its proneness to mistakes because small deviations can accumulate to large errors. Therefore, it can only be used locally to reckon a robot’s position and cannot be employed

for a global self-localization without a correction mechanism based on external information.

The basic idea of the route navigation approach presented here is the following: when the wheelchair drives using basic behaviors such as wall-following, its movements reflect the structure of the environment. The dead reckoning system of the wheelchair can record these movements. The resulting *motion tracks* can be employed to generate representations of the routes the system has followed. If the system drives along a route a second time, its dead reckoning system will produce a very similar track. In order to be able to use such representations for navigation, a method has to be found to match different tracks to perform, e.g., a self-localization along the route.

In comparison to methods that try to generate a map-like representation of the environment, e.g., with the help of distance sensors, the major advantage of motion tracks is their continuity that is a result of the continuous motion of the wheelchair. The noise in the sensor readings is considerably reduced by the inertia of the mobile system because the distance measurements of the sensors influence the route representation only indirectly via the basic behaviors.

In the navigation approach presented here, the wheelchair is controlled along a route by switching between the basic behaviors. The system records its dead reckoning positions as well as the changes of the behaviors. As the odometry data can consist of many measurements, it is *generalized* to generate a compact representation of the route. This information is stored, and it is used as reference for autonomous drives along this route. Based on the assumption that navigation in buildings is essentially a combination of following corridors and turning at corners, a representation has been chosen in which routes consist of *straight lines* that cross under certain *angles*. Therefore, a route description is a sequence of distances and angles, e.g. “800 cm, 89°, 345 cm, -83°, 566 cm”.

In an autonomous replay, the dead reckoning data is recorded, too. It is generalized the same way as during the teaching drive. The description stored always represents the complete route whereas the current track only stands for the part of the route traveled so far. Therefore, the current description can only be matched to the beginning of the stored one. The segment in the stored representation that is matched with the last segment of the current track is the *current segment*. Together with the length of the last segment in the current track, i.e. the distance to the last corner, this defines the wheelchair’s *current position* with respect to the route representation stored. This position can be used to switch between the basic behaviors at appropriate locations, and thus enables the system to repeat a route that has been stored.

In addition, this approach allows a second application: during the matching of the two generalizations, it can be determined whether they represent the same route or not. Thus it can be noticed when the wheelchair has lost its way, e.g., because a behavior was performed erroneously. This is depicted in Fig. 2 that shows three route descriptions: the first one was learned, the second is a correct repetition of the first one, and the third is an erroneous replay. In case of the

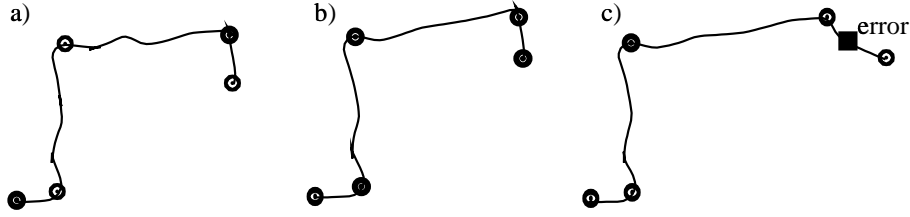


Fig. 2. Three motion tracks (each approximately 25 m in length) and the corners detected. a) Original track. b) Correct replay. c) Erroneous replay.

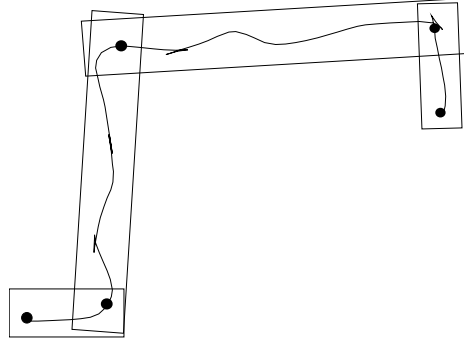


Fig. 3. The acceptance intervals for the generalization of a motion track.

latter, the mistake was detected shortly after the erroneous behavior occurred, i.e. the wheelchair missed a door because it was closed.

6 Generalization

The basic idea of generalizing a given track S is to find a simpler track G representing the general shape of S —i.e. the important global information—and suppressing small zigzags and deviations (Musto *et al.*, 1999). In the case of the wheelchair application, such deviations are generated by variations in the execution of the basic behaviors, e.g., because the system has to avoid a dynamic obstacle such as people. A simple approach to generalize S is to create a polygon track G differing less than a certain distance from S (cf. Fig. 3). As it is the goal of the generalization to unveil the structure of the environment, each segment should correspond to a corridor in reality. The wheelchair’s freedom of movement is limited by the walls of the corridors it is following. Therefore, it is reasonable to use the widths of the passages the system is traveling as threshold for the generalization.

A motion track S is a sequence of positions in a Cartesian system of coordinates with an unknown origin and with an unknown orientation. Each position consists of a x and a y coordinate. In addition, the width w of the corridor

is stored for each position. This value is measured using the wheelchair's ultrasonic sensors that are oriented sideways. In the work presented here, the distance between two subsequent measurements always was approximately 20 cm.

$$S = [s_1, s_2, \dots, s_n]$$

$$s_i = \begin{pmatrix} x_i \\ y_i \\ w_i \end{pmatrix}, \forall i \in \{1, \dots, n\} \quad (1)$$

Based on a start point s_a , longer and longer segments are constructed. For each of these temporary segments, a straight line is constructed between the starting position s_a and the temporary end position s_b . Then, the maximum deviation from this straight line is determined as error e :

$$e_{a,b}^S = \max_{i=a}^b \frac{\begin{pmatrix} x_i - x_a \\ y_i - y_a \end{pmatrix} \begin{pmatrix} x_b - x_a \\ y_b - y_a \end{pmatrix}}{\left| \begin{pmatrix} x_b - x_a \\ y_b - y_a \end{pmatrix} \right|} \quad (2)$$

As soon as this error exceeds the generalization threshold, a point s_c is known that is not a valid member of the current segment. As has been discussed earlier, the threshold for the partitioning of the motion data is the width of the corresponding corridor. For each position s_i in the motion track, a width w_i has been measured, but because these distances have been determined using sonar sensors, many outliers can be expected. To robustly estimate the widths of the corridors, all measurements $w_a \dots w_b$ are inserted into a histogram, and then this histogram H returns only the most frequent one, eliminating typical noise in the sonar measurements. As a result, the index c is either calculated as index of the first point in S behind s_a that is violating the generalization threshold, or—if the end of S is reached—as the last index n :

$$c_a^S = c_{a,a+1}'^S$$

$$\text{where } c_{a,b}'^S = \begin{cases} n & \text{if } b \geq n \\ b & \text{if } e_{a,b}^S > H_{a,b}^S \\ c_{a,b+1}^S & \text{otherwise} \end{cases} \quad (3)$$

In the next step, the point is searched for that is an adequate end position of the segment and that represents best the corner between this straight line and its successor. The basic idea for determining this position is that two successive line segments build two sides of a triangle. Seen from both end positions of the two sides, the sum of the lengths of both sides should be maximal. Therefore, the point is searched for where this sum has its biggest value. This position is supposed to be the corner between the two segments. However, this approach only works robustly if both sides of the triangle have approximately the same length. Therefore, the second segment is lengthened by the distance between s_a and s_c . As the orientation of the second segment is unknown, the segment is extended towards the direction of movement at s_c which can be approximated

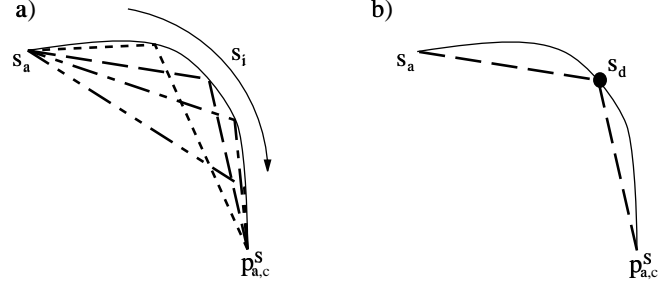


Fig. 4. Detecting the corner between two segments. a) Each s_i is connected to s_a and $p_{a,c}^S$. b) The point s_d with the maximal sum of distances is assumed to be the corner.

as the difference between the x/y components of s_c and s_{c-1} . Thus a position p is constructed as second end point of the triangle:

$$p_{a,c}^S = \begin{pmatrix} x_c \\ y_c \end{pmatrix} + \begin{pmatrix} x_c - x_{c-1} \\ y_c - y_{c-1} \end{pmatrix} \frac{\begin{vmatrix} x_c - x_a \\ y_c - y_a \end{vmatrix}}{\begin{vmatrix} x_c - x_{c-1} \\ y_c - y_{c-1} \end{vmatrix}} \quad (4)$$

With the help of p , the index d of the position that represents the corner in the track can be estimated. Hence, for each index i between a and c , the sum of the lengths of the two straight lines $s_i s_a$ and $s_i p$ are calculated, and the index with the maximum sum is selected as the index d of the corner's position (cf. Fig. 4):

$$d_{a,c}^S \in \left\{ o \mid o \in \{a \dots c\}, b_o^S = \max_{i=a}^c b_i^S \right\} \quad (5)$$

where $b_i^S = \left| \begin{pmatrix} x_i - x_a \\ y_i - y_a \end{pmatrix} \right| + \left| \begin{pmatrix} x_i \\ y_i \end{pmatrix} - p_{a,c}^S \right|$

As a result, the first segment has been determined. It ranges from s_a to s_d . The corner at s_d is also the start of the subsequent segment that can be calculated in the same way. So, the partitioning is continued until the end of S , i. e., until s_n is reached. Thus, a list G can be calculated that is the generalization of S . Each entry in this list consists of the start position and the width of a segment. The only exception from this is the last member of the list: it simply describes the last position in the original motion track.

$$\begin{aligned}
G^S &= \left[\begin{pmatrix} x'_1 \\ y'_1 \\ w'_1 \end{pmatrix} \dots \begin{pmatrix} x'_k \\ y'_k \\ w'_k \end{pmatrix} \right] = G_1^S \\
\text{where } G_a^S &= \begin{cases} [g'_{n,n}] & \text{if } a = n \\ g'_{a,d'} + G_{d'}^S & \text{otherwise} \end{cases} \\
\text{where } c' &= c_a^S \\
d' &= \begin{cases} n & \text{if } c' = n \\ d_{a,c'}^S & \text{otherwise} \end{cases} \\
g'_{a,b} &= \begin{pmatrix} x_a \\ y_a \\ H_{a,b}^S \end{pmatrix}
\end{aligned} \tag{6}$$

After the generalization G is known, the metrical lengths l_i of the $k - 1$ segments, the angles α_i between them, and the average widths of the corresponding corridors can easily be determined as abstract route description R :

$$\begin{aligned}
R &= \left[\begin{pmatrix} l_0 \\ \alpha_0 \\ w_0 \end{pmatrix} \dots \begin{pmatrix} l_{k-1} \\ \alpha_{k-1} \\ w_{k-1} \end{pmatrix} \right] \\
\text{where } l_i &= \left| \begin{pmatrix} x'_{i+1} - x'_i \\ y'_{i+1} - y'_i \end{pmatrix} \right|, \forall i \in \{1 \dots k - 1\} \\
\alpha_i &= \begin{cases} \arctan \left(\frac{x'_{i+2} - x'_{i+1}}{y'_{i+2} - y'_{i+1}} \right) & \\ -\arctan \left(\frac{x'_{i+1} - x'_i}{y'_{i+1} - y'_i} \right) & \text{if } i \in \{1 \dots k - 2\} \\ 0 & \text{if } i = k - 1 \end{cases} \\
w_i &= w'_i
\end{aligned} \tag{7}$$

An implicit assumption in the navigation approach presented in section 5 is that similar motion tracks are generalized to similar route representations. However, there are two cases in which this conjecture is violated:

1. If there are long parts with a slight curvature in a motion track, either because a corridor has this shape, or—more likely—because of odometry drift, its generalization may be arbitrary. Small variations in the wheelchair's course may generate very different segments. However, the angles between such segments that were arbitrarily separated are always small.
2. If a corridor's width is similar to its length, it may be generalized to a separate segment in one track (cf. Fig. 5a) and integrated into an adjacent segment in another track (cf. Fig. 5b).

Whereas the first problem could be handled during the generalization process by combining neighboring segments connected under small angles, the second problem cannot be eliminated during the generalization. Therefore, both problems are not taken into account before the matching process because during this phase both of them can be dealt with.

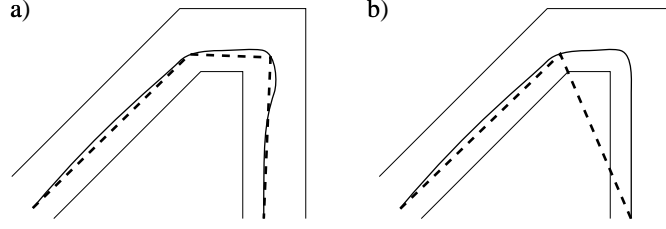


Fig. 5. Two motion tracks and their generalizations. a) The short segment was detected. b) The short segment was integrated into the following one.

7 Matching Generalized Motion Tracks

Two generalized tracks are matched by running through both of them segment by segment. On the one hand, this allows to determine corresponding segments, on the other hand, it can be checked whether both tracks describe the same route. To perform the latter, the segments of the tracks as well as the angles between them are compared. If they are not similar enough, the tracks are incompatible, and therefore it is assumed that they do not describe the same route, i. e., the wheelchair has moved along different trajectories.

As has been discussed in section 5, the current track will usually be shorter than the reference track. Therefore, the matching of the two tracks is not a symmetric process because one of the two route representations is allowed to be shorter than the other one. Hence, the matching function tests whether the current track is *less or equal* than the reference track, and if it is, it determines the position in the reference track where the current track ends, i. e. it calculates the corresponding segment in the reference track and uses the length of the last segment in the current track as metrical offset from the beginning of the segment.

The matching of the two tracks is performed segment by segment, starting at their beginnings, i. e. at the segments with the first indices:

$$M^{R,R'} = M_{1,1}^{R,R'} \quad (8)$$

For each pair of segments from both tracks, it is first checked whether the segment in the current track is significantly longer than the corresponding one in the reference track. If this is the case, the tracks seem to be incompatible. However, as has been discussed in the previous section, it is possible that the segment in the current track is too long because it is the counterpart for two segments in the reference track. Therefore, the function J is called, that attempts to join the actual segment in the reference track with its successor, and then the matching is retried. The opposite case of the previous one is given if the actual segment in the current track is significantly shorter than the one in the reference track, and it is not the last segment (indicated by $\alpha' \neq 0$). Then, it is tried to overcome the mismatch by joining two adjacent segments in the current track.

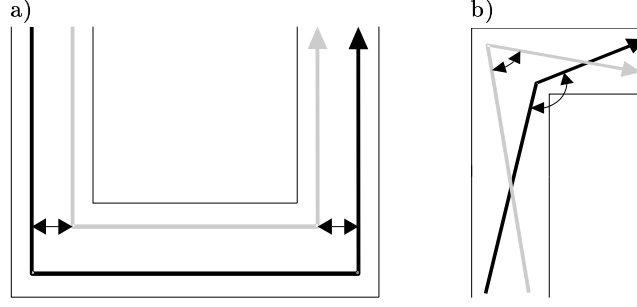


Fig. 6. Tolerances possible in the course of motion. a) In segment length. b) Angular deviations between two segments.

$$M_{i,i'}^{R,R'} = \begin{cases} M_{i,i'}^{W,R'} & \text{if } l_{i'}' > l_i + T_i^R \\ M_{i,i'}^{R,W'} & \text{if } l_{i'}' < l_i - T_i^R \wedge \alpha_{i'}' \neq 0 \\ M_{i,i'}^{R,R'} & \text{otherwise} \end{cases} \quad (9)$$

where $W = J_i^R$
 $W' = J_{i'}^{R'}$

The function T defines the tolerance for the comparison of the lengths. This threshold consists of four values:

- The width of the previous segment w' if the current segment is not the first one (cf. Fig. 6a).
- The width of the successive segment w'' if the current segment is not the last one (cf. Fig. 6a).
- A base tolerance t_{base} . This value should be larger than the distance between two neighboring positions in the original tracks. In the work presented here, it was set to 50 cm.
- A factor t_{factor} that models odometry errors depending on the distance traveled. The wheelchair's odometry is quite reliable in measuring lengths of straight lines. So, t_{factor} can be small, i. e. 2%.

$$T_i^R = w' + w'' + t_{\text{base}} + l_i t_{\text{factor}}$$

where $w' = \begin{cases} w_{i-1} & \text{if } i > 1 \\ 0 & \text{otherwise} \end{cases}$ (10)

$w'' = \begin{cases} w_{i+1} & \text{if } \alpha_i \neq 0 \\ 0 & \text{otherwise} \end{cases}$

The function J tries to join the segment i with its successor in a route description R . An integration of two segments into one is only possible

- if the segment i is not the last one,

- and either the segment i or $i + 1$ is short, i.e. it is not more than twice as long as the corresponding corridor is wide,
- or the angle between both segments is small, i.e. less than α_{\min} .

Otherwise, an error is returned because both route descriptions are incompatible. If the segments can be joined, this is performed geometrically. The length l_i of the new segment is determined from a vectorial addition of the lengths of the original segments. The resulting angle α_i to the following segment is either the sum of the original angles, or it is zero, if the new segment is the last one in the route description. It is unnecessary to adapt the angle to the previous route segment, because it is ignored by any further processing. The width w_i of the new segment is the maximum width of the two segments replaced.

$$J_i^R = \begin{cases} R' & \text{if } \alpha_i \neq 0 \wedge (l_i < 2w_i \vee l_{i+1} < 2w_{i+1} \vee |\alpha_i| < \alpha_{\min}) \\ \text{error} & \text{otherwise} \end{cases}$$

$$\text{where } r'_j = \begin{cases} r_j & \text{if } j \in \{1 \dots i - 1\} \\ \begin{pmatrix} l'' \\ \alpha'' \\ w'' \end{pmatrix} & \text{if } j = i \\ r_{j+1} & \text{otherwise} \end{cases} \quad (11)$$

$$l'' = \left| \begin{pmatrix} l + l_{i+1} \cos \alpha_i \\ l_{i+1} \sin \alpha_i \end{pmatrix} \right|$$

$$\alpha'' = \begin{cases} 0 & \text{if } \alpha_{i+1} = 0 \\ \alpha_i + \alpha_{i+1} & \text{otherwise} \end{cases}$$

$$w'' = \max(w_i, w_{i+1})$$

If the actual segment in the current track is the last one (indicated by $\alpha' = 0$), the matching was successful, and the index of the corresponding segment in the reference track as well as the length of the last segment in the current track are returned. This pair represents the wheelchair's current position, and it is used to start the basic behaviors at the same locations as during the teaching drive. Otherwise, if the end of the reference track is reached, the current track is longer than the reference track, and hence the representations are incompatible:

$$M_{i,i'}^{R,R'} = \begin{cases} \begin{pmatrix} i \\ l'_{i'} \end{pmatrix} & \text{if } \alpha'_{i'} = 0 \\ \text{error} & \text{if } \alpha'_{i'} \neq 0 \wedge \alpha_i = 0 \\ M_{i,i'}^{R,R'} & \text{otherwise} \end{cases} \quad (12)$$

Next, the angles following these segments in both tracks are compared. This comparison is performed in a qualitative way because the angles may differ heavily, either because the motion tracks can vary between two recordings (cf. Fig. 6b), because of the wheelchair's weakness in measuring angles (cf. Fig. 7b and 7c), or if parts of the route are generalized with a different number of segments. The angles are mapped to the four qualitative categories "left", "right", "forwards", and "backwards" that describe overlapping angular ranges:

$$\begin{aligned}
\text{left}_\alpha &= \alpha \in]0 \dots \pi[\\
\text{right}_\alpha &= \alpha \in]-\pi \dots 0[\\
\text{forwards}_\alpha &= \alpha \in [-\alpha_{\min} \dots \alpha_{\min}] \\
\text{backwards}_\alpha &= \alpha \notin [-\pi + \alpha_{\min} \dots \pi - \alpha_{\min}]
\end{aligned} \tag{13}$$

Two angles are assumed to be compatible if they both share at least one qualitative category. This is determined by the function C :

$$\begin{aligned}
C_{\alpha,\beta} = & \text{left}_\alpha \wedge \text{left}_\beta \vee \text{right}_\alpha \wedge \text{right}_\beta \vee \text{forwards}_\alpha \wedge \text{forwards}_\beta \vee \\
& \text{backwards}_\alpha \wedge \text{backwards}_\beta
\end{aligned} \tag{14}$$

So, if both angles are compatible, the comparison is continued with the next segments. Otherwise, an error is indicated.

$$M_{i,i'}^{'''R,R'} = \begin{cases} M_{i+1,i'+1}'^{R,R'} & \text{if } C_{\alpha_i,\alpha_{i'}} \\ \text{error} & \text{otherwise} \end{cases} \tag{15}$$

8 Results

The generalization and matching algorithms presented in this paper are improvements of methods developed earlier (Röfer, 1998). The motion tracks in Fig. 2 were recorded by the first prototype of the Bremen Autonomous Wheelchair, and they were generalized using the old version of the algorithm. The experiments with the Nomad 200 of the University of Manchester were also performed using the first generalization and matching approach. The results were documented in detail by Röfer (1998). Although the robot was able to follow very long routes, it had problems with tracks containing a short segment between two very long ones. Using the enhanced version of the algorithm that has been presented in this paper, the actual Bremen Autonomous Wheelchair “Rolland” had no problems to follow arbitrary routes, as long as they could be described using its basic behaviors. Figure 7a schematically shows a typical route that was tested. The resulting motion tracks, together with the corners detected, are depicted in Fig. 7b and 7c. Although the odometry drift is quite different between both tracks, the same corners are detected in both of them. In addition, the wheelchair was able to cope with situations as shown in Fig. 5. Therefore, the route following approach is very robust against typical odometry errors and against variations in the execution of the basic behaviors. In addition, its computational complexity is small enough to run on slower computers as the 486/80 used on the Nomad 200.

9 Conclusion

In this paper, a simple approach for the acquisition and representation of spatial knowledge for the purpose of navigation has been presented that satisfies

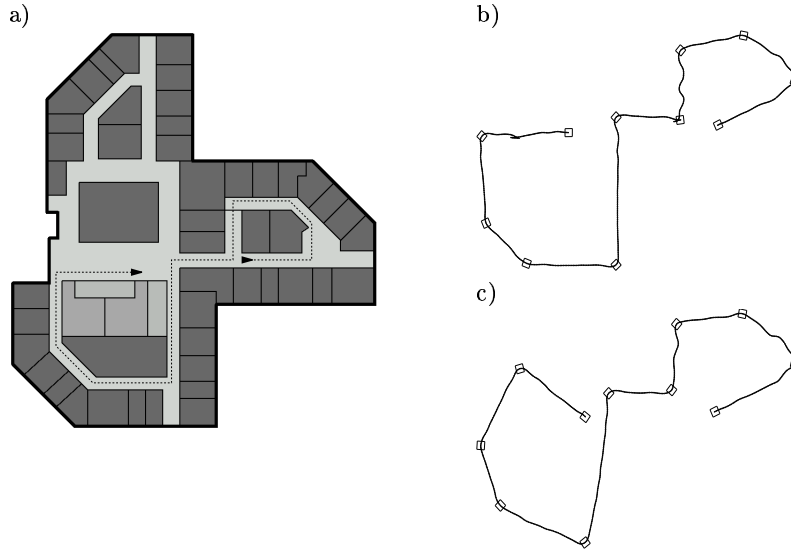


Fig. 7. Angular errors in long (> 100 m) motion tracks. a) The test scene. b+c) Two different tracks and the corners detected.

the demands for simplicity and robustness of the target application that was chosen: a semi-autonomous wheelchair supporting persons with severe impairments. The navigation approach discussed is a combination of the execution of basic behaviors and the generalization of the track of motion recorded while the wheelchair is performing these behaviors. After a route has been trained once, it is stored in a very compact representation, enabling the wheelchair to follow it autonomously.

During the development of the navigation method, the approach was implemented on three different robotics platforms. Thus, it has proven to be a universal and robust method for navigation.

Acknowledgement

The author is supported by the Deutsche Forschungsgemeinschaft through the priority program “Spatial Cognition.”

References

- Burgard, W., Fox, D., and Henning, D. (1997). Fast grid-based position tracking for mobile robots. In G. Brewka, C. Habel, and B. Nebel, editors, *KI-97: Advances in Artificial Intelligence*, Lecture Notes in Artificial Intelligence, pages 289–300, Berlin, Heidelberg, New York. Springer.

- Franz, M. O., Schölkopf, B., Georg, P., Mallot, H. A., and Bülthoff, H. H. (1997). Learning view graphs for robot navigation. In W. L. Johnson, editor, *Proc. 1st Int. Conf. on Autonomous Agents*, pages 138–147, New York. ACM Press.
- Gutmann, J.-S. and Nebel, B. (1997). Navigation mobiler Roboter mit Laserscans. In P. Levi, T. Bräunl, and N. Oswald, editors, *Autonome Mobile Systeme*, Informatik aktuell, pages 36–47, Berlin, Heidelberg New York. Springer.
- Hoyer, H., Hoelper, R., and Pabst, U. (1994). Benutzerspezifische Wegplanung für omnidirektionale und kinematisch beschränkte Rollstühle. In P. Levi and T. Bräunl, editors, *Autonome Mobile Systeme*, Informatik aktuell, pages 274–284, Berlin, Heidelberg, New York. Springer.
- Jörg, K.-W., v. Puttkamer, E., and Richstein, H.-J. (1993). Integration und Fusion heterogener Multisensorinformation zur geometrischen Weltmodellierung für einen Autonomen Mobilen Roboter. In G. Schmidt, editor, *Autonome Mobile Systeme*, pages 287–298, Technische Universität München.
- Kollmann, J., Lankenau, A., Bühlmeier, A., Krieg-Brückner, B., and Röfer, T. (1997). Navigation of a kinematically restricted wheelchair by the parti-game algorithm. In *Spatial Reasoning in Mobile Robots and Animals*, pages 35–44, Manchester University. AISB-97 Workshop.
- Krieg-Brückner, B., Röfer, T., Carmesin, H.-O., and Müller, R. (1998). A taxonomy of spatial knowledge and its application to the Bremen Autonomous Wheelchair. In C. Freksa, C. Habel, and K. F. Wender, editors, *Spatial Cognition*, volume 1404 of *Lecture Notes in Artificial Intelligence*, pages 373–397, Berlin, Heidelberg, New York. Springer.
- Lankenau, A. and Meyer, O. (1997). *Der autonome Rollstuhl als sicheres eingebettetes System*. Master's thesis, Universität Bremen.
- Lankenau, A., Meyer, O., and Krieg-Brückner, B. (1998). Safety in robotics: The Bremen Autonomous Wheelchair. In *Proceedings of AMC'98, 5th Int. Workshop on Advanced Motion Control*, pages 524–529, Coimbra, Portugal.
- Mojaev, A. and Zell, A. (1998). Online-Positionskorrektur für mobile Roboter durch Korrelation lokaler Gitterkarten. In H. Wörn, R. Dillmann, and D. Henrich, editors, *Autonome Mobile Systeme*, Informatik aktuell, pages 93–99, Berlin, Heidelberg, New York. Springer.
- Musto, A., Stein, K., Eisenkolb, A., and Röfer, T. (1999). Qualitative and quantitative representations of locomotion and their application in robot navigation. In *Proc. Int. Joint Conf. On Artificial Intelligence*. to appear. Already available as Forschungsbericht Künstliche Intelligenz, FKI-228-99, Technische Universität München.
- Owen, C. and Nehmzow, U. (1997). Middle scale navigation – a case study. In *Spatial Reasoning in Mobile Robots and Animals*, pages 104–112, Manchester University. AISB-97 Workshop.
- Röfer, T. and Lankenau, A. (1998). Architecture and applications of the Bremen Autonomous Wheelchair. In P. P. Wang, editor, *Proc. of the Fourth Joint Conference on Information Systems*, volume 1, pages 365–368. Association for Intelligent Machinery.
- Röfer, T. and Müller, R. (1998). Navigation and routemark detection of the Bremen Autonomous Wheelchair. In T. Luth, R. Dillmann, P. Dario, and H. Wörn, editors, *Distributed Autonomous Robotics Systems*, pages 183–192, Berlin, Heidelberg, New York. Springer.
- Röfer, T. (1998). Routenbeschreibung durch Odometrie-Scans. In H. Wörn, R. Dillmann, and D. Henrich, editors, *Autonome Mobile Systeme 1998*, Informatik aktuell, pages 122–129, Berlin, Heidelberg, New York. Springer.