iWalker – An Intelligent Walker providing Services for the Elderly

Dr. Thomas Röfer, Tim Laue, Dr. Bernd Gersdorf,
Deutsches Forschungszentrum für Künstliche Intelligenz, Germany

Abstract

The DFKI iWalker is the prototype of a walker equipped with electric brakes, wheel encoders, a laser range sensor, and a small control PC. The laser range sensor enables the iWalker to measure the distances to the objects in the environment. Thereby, the iWalker can provide a variety of services. By controlling the electric brakes, it can help the user keeping clear of obstacles. In addition, by using a laser-scan-based self-localization algorithm, the iWalker can determine its position inside a building and provide location-based services. For instance, it can guide the user to a desired target location by displaying navigation instructions on the screen (as in a car navigation system) or by using the electric brakes to indicate the direction the user has to walk to.

1 Introduction

The iWalker is based on a commercial walker. It is upgraded with electric brakes, wheel encoders, and its own battery [1], but the current model has no active drive. A Hokuyo laser range sensor at the front of the iWalker (cf. Fig. 1 bottom right) and a netbook-class PC below the seat (cf. Fig. 1 top right and Fig. 3) were installed. The netbook can be used with either a closed lid or an open lid. If the lid is open, the screen is visible behind the seat and can be used to display information. In both cases, the seat can still be used. Three assistants were developed on the basis of the iWalker: the Walking Assistant, the Navigation Assistant, and the Guiding Assistant.

Fig. 1. The iWalker with electrically breakable rear wheels, a netbook, and a laser range sensor. The netbook is mounted below the seat (upper right picture, also cf. Fig. 3). The Hokuyo laser range sensor is installed at the front (lower right picture).

2 Walking Assistant

The purpose of the Walking Assistant is to avoid obstacles while the user is walking with the iWalker. The user remains in complete control of the system as long as there are no obstacles on the intended path. Whenever obstacles block the way, the Walking Assistant takes control and avoids the obstacle with as little braking forces as possible. This behaviour is based on the readings of the laser range sensor mounted to the iWalker. The readings are used to maintain a map representing the local environment around the iWalker (cf. Fig. 2). The basic idea of the Walking Assistant is to detour obstacles in a way that is most likely to be acceptable for the user. By taking into account the desired travelling direction in terms of the curve indicated by the user, this module decides whether the iWalker should steer to the right or to the left. Without braking, the current motion speeds of the iWalker clearly indicate the intended walking direction. However, when the Walking Assistant intervenes to avoid an obstacle, the direction of movement is a combination of the forces applied by the user and the forces applied by the brakes of the iWalker. Hence, the motion direction does not reflect the user's original command anymore. Therefore, the cur-

rent speeds (translational and rotational) measured by the wheel encoders together with the braking force currently applied are used to estimate the actual direction the user wants to go to.

A typical situation is depicted in Fig. 2: The iWalker is placed in front of a doorway. If the user simply walked forward, it would hit into the right door post. If the user walks a right turn (as shown by the region checked for obstacles), the Walking Assistant infers that he or she wants to pass the obstacle (i.e. the right door post) on the right-hand side. Thus it reinforces its steering command to the right. If, instead, the user would indicate a left curve, the assistive system assumes that the driver wants to pass the obstacle on the left-hand side, i.e., to pass through the doorway.

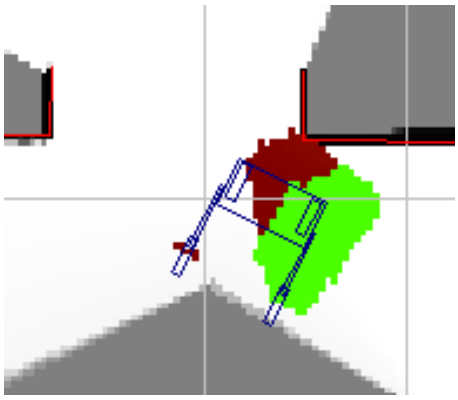

Fig. 2. Deciding on which side the user wants the obstacle to be passed. Obstacles within the green (bright) region are avoided to the left, while obstacles in the red (darker) region are avoided to the right.

All the time, the Walking Assistant searches the local map for obstacles based on the intended walking direction employing the corresponding *safety region*, i.e. the region that the iWalker will reach within the next few seconds. If the iWalker is already detouring, the direction indicated by the user may deviate from the system's current steering direction. Thus, the Walking Assistant always assesses the world from the user's point of view, and this view includes the judgement whether an obstacle is on the left or on the right side of the intended driving direction.

To allow the Walking Assistant to reconstruct this assessment, it was pre-processed for each cell in a safety region whether it is better to avoid an obstacle at the corresponding real-world position on the left side or on the right side. For instance, if the iWalker would hit the obstacle with its front left, it would try to avoid it on the right side. In contrast, if a collision was expected on the left side behind the driving axle, the iWalker would also turn to the left. So, the avoidance direction depends on the obstacle's position relative to the centre of the iWalker's driving axle. Fig. 2 shows an example of a resulting decision diagram. Note that if the indicated course directly points towards an object, the Walking Assistant will not intervene at all. This is because in such a case it is likely that the driver's intention is to move close to that obstacle and not to pass around it, e.g. when going to a desk. In contrast, obstacles are always avoided to both sides if they are on the iWalker's sides (i.e. next to it on the left or on the right), e.g., to make sure that the iWalker does not get stuck between doorposts.

3 Navigation Assistant

The Navigation Assistant allows the user to select a target location and guides the user to that location by showing a guidance arrow on the screen of the iWalker (cf. Fig. 3 left). Since the iWalker currently has no real user input devices, odometry is used as input channel. When the iWalker is standing still, it displays a question mark in the upper left corner of the screen (cf. Fig. 3 left). This is a signal that the iWalker is ready to accept the gesture to enter the selection menu. The gesture is to quickly turn the iWalker slightly to the left and the right (in any order). After that, the menu appears on the screen (cf. Fig. 3 right). The menu lists all available target locations in alphabetical order (in the current implementation, the menu is also used to select the assistant currently running). The user can now scroll through the menu items by moving the iWalker forwards and backwards. A target location is selected by turning the iWalker to the left. The previous selection is kept by turning the walker to the right (i.e. cancel the menu).

Many different components are necessary to realize such an assistant. First of all, a map of the environment needs to be created and the iWalker must be able to self-localize within this map (cf. Sect. 3.1). Afterwards, a *Route Graph* [4] is specified by using a special editing component (cf. Fig. 4). The navigation within this graph, i.e. localization within the graph, and route search, is described in Sect. 3.2. Finally, the guidance information is provided to the user (cf. Sect. 3.3).

Fig. 3. The Navigation Assistant. The left picture shows the arrow guiding the user to the target location. The right picture shows the menu to select a new target location. The selection is done by moving the iWalker.

3.1 Mapping and Localization

The simultaneous localization and mapping (SLAM) problem has been intensively studied in the robotics community in the past. We have chosen the *GMapping* suite from [6] since it has been developed for sensorial equipment similar to the iWalker's and has shown to be able to cope with environments up to a large scale. The *GMapping* approach uses a Rao-Blackwellized particle filter in which each particle carries an individual grid map of the environment. It includes adaptive techniques to reduce the number of particles (and thus the computing time) for learning grid maps. The approach computes an accurate proposal distribution taking into account not only the movement of the robot but also the most recent observation. This drastically decreases the uncertainty about the robot's pose in the prediction step of the filter. Furthermore, an approach to selectively carry out re-sampling operations is applied which seriously reduces the problem of particle depletion. Detailed information about this approach is found in [2].

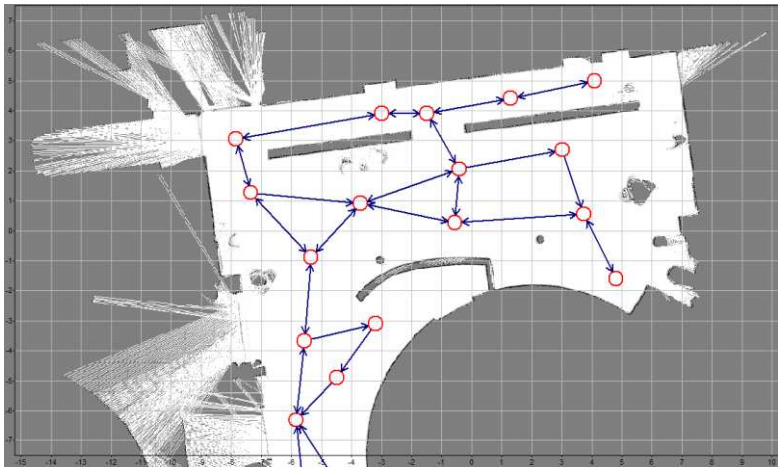

Fig. 4. Example of a simple route graph embedded in a map created by *GMapping*

3.2 Navigation

Before planning any path towards the destination, the iWalker must be localized within the graph. Initially, this is done by simply determining the edge (out of all being part of the graph) closest to the iWalker. During the drive along a computed path, the graph localization only considers edges which are part of the planned route to prevent any oscillations and to avoid any re-planning. Given the position of the iWalker as well as a destination, planning a path is reduced to a graph search problem, which is trivial compared to efficiently planning a path in continuous space. We implemented the A* algorithm [3] since it is quite fast (in our application by using a metric distance heuristics) and proven to always find the shortest path.

3.3 Displaying the Guidance Arrow

A virtual target point is continuously moved along the planned path, always being at a certain distance ahead of the iWalker. This target point is used together with three other points to compute a cubic Bézier curve. The curve is the base for a *Guidance Arrow* which represents the necessary driving direction, similar to the one of a car navigation system (cf. Fig. 3 left). The first point of the Bézier curve is placed straight behind the iWalker in a fixed distance. It simple lets the arrow start a little bit lower in the view. Together with the second point that is at the position of the iWalker itself, it ensures that the base of the arrow is always pointing upwards. The third point is located in front of the iWalker, but shifted sideways in the direction of the target point. It makes sure that the arrow is even a curve when it is pointing backwards.

4 Guiding Assistant

The Guiding Assistant is the combination of the two assistants described above, i.e. the Walking Assistant and the Navigation Assistant. Instead of displaying a guidance arrow on the screen, the user is guided to the target location by braking. The general approach consists of guidance controller that uses the current direction to drive generated by the Navigation Assistant to compute an "intended walking direction" for the Walking Assistant. The latter is then using the brakes of the iWalker to guide the user to the target location. The guidance controller generates motion commands as if the iWalker would drive in free space, i.e., it completely ignores the current obstacle situation. However, since the Walking Assistant avoids obstacles, the iWalker is robustly able to navigate even in narrow environments, as long as the target poses are selected appropriately. In general, target poses should be located at narrow passages, e.g. between door posts, with the orientation perpendicular to the narrowness. Since the navigation graph described above is created manually, this requirement can rather easily be fulfilled.

5 Conclusions

The paper has presented the iWalker that provides services for the elderly. Three assistants have been implemented that support the user of a walker in different ways to solve navigation tasks. In an automated environment such as the Bremen Ambient Assisted Living Lab (BAALL) [5], the known position of the iWalker can also be used to open doors, prepare the bed for entering comfortably, or to adapt the height of the kitchen equipment for the user of the iWalker. The next version of the iWalker will also be equipped with motors so that it can park itself away if it is not used, and return on demand.

Acknowledgements

This work has been funded by the European Commission in the context of the 6th Framework Programme for RTD in the context of the project "SHARE-it - Supported Human Autonomy for Recovery and Enhancement of cognitive and motor abilities using information technologies" under the contract number FP6-045088.

6 References

[1] U. Cortés, A. Martínez-Velasco, C. Barrué, T. Benedico, F. Campana, J. Fernández, R. Annicchiarico (2008). A SHARE-it service to elders' mobility using the i-Walker. In *Gerontechnology*, 7(2):95.

[2] G. Grisetti, C. Stachniss, W. Burgard (2006). Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters. In *IEEE Transactions on Robotics*.

[3] P.E. Hart, N.J. Nilsson, B. Raphael (1968). A formal basis for the heuristic determination of minimum cost paths. In IEEE Transactions on Systems Science and Cybernetics, vol. 4, no. 2, pp. 100-107.

[4] B. Krieg-Brückner, U. Frese, K. Lüttich, C. Mandel, T. Mossakowski, R. Ross (2005). Specification of an Ontology for Route Graphs. In *Spatial Cognition IV*, vol. 3343 of Lecture Notes in Artificial Intelligence, pp. 390-412, Springer-Verlag.

[5] B. Krieg-Brückner, B. Gersdorf, M. Döhle, K. Schill (2009). Technik für Senioren in spe im Bremen Ambient Assisted Living Lab. In *2. Deutscher AAL-Kongress 2009*, VDE-Verlag Berlin-Offenbach.

[6] C. Stachniss, U. Frese, G. Grisetti (2009). OpenSLAM.org. http://openslam.org/.

Contact: Thomas Röfer, Thomas.Roefer@dfki.de, +49(421)218-64200