Applying a 3DOF Orientation Tracker as a Human-Robot Interface for Autonomous Wheelchairs

Christian Mandel* and Thomas Röfer† and Udo Frese‡

Abstract

Within this work, we demonstrate the applicability of a three degrees of freedom orientation tracker as suitable controlling equipment for an autonomous wheelchair. It is shown that a small-size sensor device, like the XSens MTx, can either serve as a joystick replacement, or as an interface device for a newly developed navigation algorithm. Specifically, the sensor device is mounted at the back of its operator's head, where it permanently measures posture values that are converted into adequate steering commands. Comprehensive experiments in a real world office scenario with untrained participants are used for evaluation purposes.

1. INTRODUCTION

Disabled people often find the use of electrical wheelchairs an indispensable aid for their rehabilitation or daily life. Since the most common way of operating such a device is given by a control loop that consists of a technical system e.g. a joystick connected to actuating elements, and the human operator, engineers have historically focused on improving aspects of these kinds of systems. The gathered results range from safety-layers that act in-between the control loop and prevent the wheelchair-bound person from colliding with any obstacles, to autonomous navigation modules that sometimes require different kinds of user interfaces than a joystick. While the work that is presented in this paper is still a technical one, its motivation clearly grounds on the user's requirements in an intelligent wheelchair.

*C. Mandel is with Department of Computer Science, University of Bremen, P.O.Box 330440, 28334 Bremen, Germany cmandel@uni-bremen.de

†T. Röfer is with DFKI Lab Bremen, Safe and Secure Cognitive Systems, Robert-Hooke-Straße 5, 28359 Bremen, Germany roefer@informatik.uni-bremen.de

‡U. Frese is with is with DFKI Lab Bremen, Safe and Secure Cognitive Systems, Robert-Hooke-Straße 5, 28359 Bremen, Germany ufrese@informatik.uni-bremen.de

People with high level quadriplegia suffer from paralysis of all four limbs while still being able to move their heads. Thus a common wheelchair-interface like a joystick moved by hand is obviously inappropriate. For this kind of handicap, we propose a head-mounted three degrees of freedom orientation tracker[1] to allow the operator of an automated wheelchair to steer his/her vehicle by intuitive head movements. In the following we will develop two methods with which a small-size IMU can be exploited for this purpose by forwarding head-posture dependent joystick-like signals, or by interfacing a fully-fledged autonomous navigation module respectively. The later one computes a nearby target-pose by intersecting the operator's line of view with a two-dimensional local obstacle map, so that a geometric path planner can subsequently compute an optimal trajectory within that map.

We begin in section 2 with a brief overview of approaches that deal with a variety of human-robot interfaces, in particular focusing on control mechanisms for automated wheelchairs. In section 3 we continue with an introduction of our experimental platform *Rolland III*, followed by a detailed view on the *XSens MTx* IMU, as well as a short overview of the software-framework applied for this work. Section 4 proceeds with a detailed account on the implementation of an IMU-based head-joystick, before we then describe in section 5 our developed geometric path planner that applies cubic Bezier curves and is interfaced with the IMU. After section 6 shows the results of an experimental evaluation that comprised both of the presented IMU-controllers, we conclude in section 7 with an assessment of the gathered results.

2. Related Work

For many years, human-robot interaction (HRI) has been an active research field that studies methods of interfacing with (semi-)autonomous robot systems.

[1] In the following we will abbreviate *three degrees of freedom orientation tracker* or rather *inertial measurement unit* by *IMU*.

Within the domain of operating electrical wheelchairs, scientists began developing techniques that should enable paralysed people to steer their vehicle.

Jaffe proposed in [1] a head position interface that applies two ultrasonic sensors, mounted at the wheelchair's head-rest in order to sense the user's head position. Despite the contactless and therewith smart fixation of the hardware components, this approach implicates the drawback that the user's head position can only be measured in two-dimensional space. Thus only basic head movements within a plane parallel to the ground can be evaluated. Within a clinical trial [2], an evaluated version of Jaffe's work has been rated by 17 participants, all suffering from spinal cord injury dysfunction. Due to the fact that the assessed equipment was nearly identical in construction to its predecessor, virtually one third of all subjects reported poor performance in commanding turns, while about 40% judged straight-line driving poor or very poor.

More recent work of Chen and colleagues [3] proposes a tilt sensor module that is attached to the back of the user's head. With the help of two integrated inertial sensors, the overall device interprets two-dimensional head movements and subsequently triggers an appropriate translational or rotational motion of the vehicle. Due to the drifting output of the sensor, the overall system can only determine movements of the head but no globally correct posture angles. Hence, only discrete steering commands can be generated, e.g. $70cm/s$ translational speed when the operator moves his/her head forward once, or $100cm/s$ translational speed wheen the operator moves his/her head forward twice.

A completely different approach in the development of an advanced wheelchair user interface is presented in the work of Canzler and Kraiss [4]. The authors describe a system that extracts and analyses facial features like head posture, gaze direction, and lip movement with the help of computer vision. While we want to abstract away from actual implementation details that comprise the adaption of detailed geometry and texture information, it should be noted that their system has been tested under real world conditions. Here it was able to recognize at least four gesture dependend commands like *go*, *stop*, *left*, and *right*.

3. System Overview

In this section we overview the hard- and software components that provide the prerequisites for the implementation of an IMU-based interface to electrical wheelchairs.

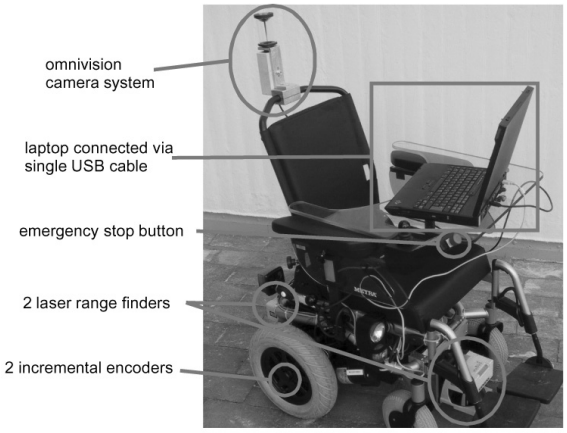


Figure 1. The autonomous wheelchair *Rolland* along with its sensorial equipment and a processing laptop.

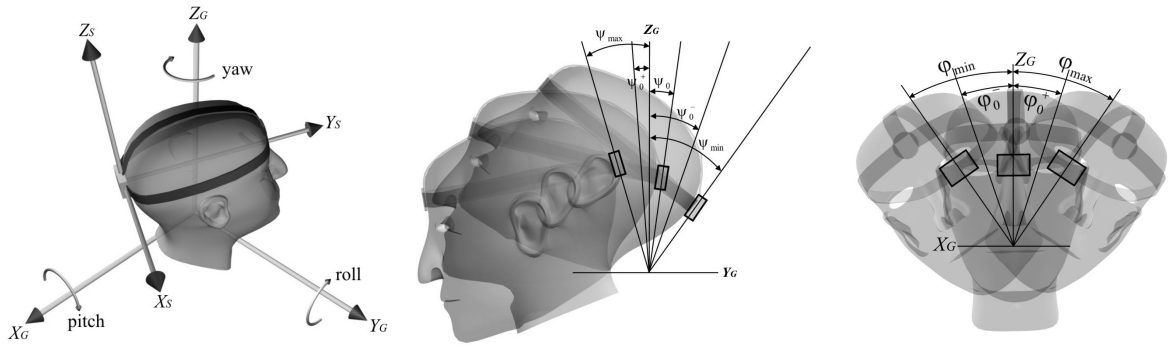
3.1. Hardware

For several years, the autonomous wheelchair *Rolland* has been used in our laboratory as an experimental platform for the investigation of questions related to the field of service- and rehabilitation-robotics, cf. [5, 6, 7]. The current model is based on the electrical wheelchair *Champ 1.594*, produced by the German company *Meyra*. It is equipped with two laser range finders, mounted beneath the operator's feet, that sense distances to nearby obstacles. Further hardware includes two incremental encoders that measure wheel-rotation for dead reckoning, an omnivision camera system, and a processing laptop, cf. Fig.1.

The *XSens MTx* IMU is a small-scale[2] electronical device that provides inertial information, namely 3D acceleration, 3D rate of turn, and 3D earth-magnetic field [8]. By the combination of the output of the device's internal accelerometers, gyroscopes, and magnetometers the IMU outputs also drift-free absolute 3D orientation data. In order to use this device for measuring the posture of a person's head, we have mounted the IMU at the head's back with the help of a small-sized and easy to wear frontlet, cf. Fig.2(a). Throughout this work we have set the configuration of the IMU to output Euler angles that describe the orientation of the IMU's local coordinate system S with respect to the fixed global coordinate system G. We refer to a single IMU reading by the triple $\mathscr{I} = (\psi, \varphi, \theta)$, as defined in (1).

$$\begin{aligned} \psi &= pitch = rotation\ around\ X_G \in [-90°...90°] \\ \varphi &= roll = rotation\ around\ Y_G \in [-180°...180°] \\ \theta &= yaw = rotation\ around\ Z_G \in [-180°...180°] \end{aligned} \quad (1)$$

[2] outline dimensions: $53 * 38 * 21\ mm^3$, weight: $30\ g$



(a) Schematic view of the inertial measurement unit mounted at the back of the user's head, including the global coordinate system G and the sensor's local coordinate system S.

(b) *left:* Maximal pitch deflection ψ_{max}, minimal pitch deflection ψ_{min}, and mean pitch deflection ψ_0, as adopted during calibration phase of the IMU. ψ_0^+ and ψ_0^- characterise the pitch dead zone, i.e. only pitch angles exceeding these values will be accepted as control commands. *right:* Roll angles $\varphi_{max}, \varphi_0^+, \varphi_0, \varphi_0^-, \varphi_{min}$ are defined in analogy.

Figure 2. Illustration of an inertial measurement unit that is attached to the user's head with the help of an easy to wear frontlet. The depicted head posture angles are calculated during the calibration phase of the IMU and given in the global coordinate system G.

3.2. Software

The hardware described above, which is used within the context of this work, is driven by an adaption of a software architecture that was originally developed for the GermanTeam, the world champion 2004 and 2005 in the Four-Legged League in RoboCup [9]. The framework structures the code into *modules*, *representations*, and *processes*. A module solves a specific task and is encapsulated by a well-defined interface consisting of representations. For each module, several solutions may exist, between which one can switch at runtime, and a module may also be deactivated completely. Processes run concurrently and group modules together. They define the flow of information (the representations) between the modules, whether modules are part of the same process or of different ones. For the purpose of applying a head-mounted IMU as a joystick replacement, there exist at least two important modules. The *Drive Controller* gathers the raw data coming from the IMU, and converts this information into desired translational and rotational speeds for the vehicle. See sec. 4 for an in-depth discussion of this module. Before these values are forwarded to the executing actuators, they are assessed by a crucial second module, the so-called *Safety Layer*. The key concept in the implementation of the safety layer is the *Virtual Sensor*. For a given initial orientation (θ) of the robot and a pair of translational (v) and rotational (w) speeds, it stores the indices of cells of a local obstacle map that the robot's shape would occupy when initiating an immediate full stop manoeuvre. A set of precomputed virtual sensors for all combinations of (θ, v, w) then allow us to check the safety of any driving command issued by the Drive Controller.

In the case of utilising the IMU as an interface device for a geometric path planning algorithm, there are three major modules involved. At first, the *TargetRequestGenerator* tries to convert the actual head posture information, i.e. the current IMU reading $\mathscr{I}$, into a desired *TargetRequest*, cf. sec. 5.1. Afterwards, the *LocalPathPlanner* searches for a suitable trajectory towards the given target request. See section 5.2 for a discussion of this module. Beside the subsequent execution of a path- and a velocity control module, whose discussion is left out for the sake of compactness, it should be noted that within this scenario the safety layer is also applied. Therefore the overall system gains a second level of safety mechanism, this in addition to the local path planner module that itself only generates obstacle free paths.

4. IMU-based Head-Joystick

This section presents implementation details for a Drive Controller that interprets IMU-readings as head-joystick signals. In the targeted application scenario, the operator controls his/her vehicle by pitching his/her head forwards and backwards in order to control translational velocity. In analogy, left and right roll movements of the users's head control rotational velocity.

Before we can use the IMU as a joystick replacement, that is in use by a certain operator, we have to calibrate the device in the sense that we want to adopt the minimal, the mean, and the maximal deflection of the person's head w.r.t. each axis in use. For this rea-

son, let P_{min} and P_{max} be two sets of IMU-readings that have been taken while the user has pitched his/her head with maximal deflection forwards and backwards respectively. In analogy, let R_{min} and R_{max} be two sets of IMU-readings that characterise the minimal and the maximal roll deflection of the user's head. By computing the arithmetic mean of the ψ and φ components of each of the four sets, we get a good approximation for the user's minimal and maximal head deflections, i.e. ψ_{max}, ψ_{min}, φ_{max}, and φ_{min}. Furthermore we describe the rest position of the person's head by $\psi_0 = \frac{\psi_{max}+\psi_{min}}{2}$ and $\varphi_0 = \frac{\varphi_{max}+\varphi_{min}}{2}$.

In order to allow the wheelchair bound person to move his or her head a bit without causing unintended motion, we now define a dead zone around ψ_0 and φ_0 by introducing ψ_0^+, ψ_0^-, φ_0^+, and φ_0^-, cf. Fig.2(b) for an illustration of the defined angles. A valid head-joystick command (v,w) is now solely defined for input values $(\psi_{valid}, \varphi_{valid})$ that satisfy (2), and computed as in (3). Constants c_v and c_w are used to map v and w onto the velocity-domain of a particular vehicle.

$$\begin{aligned} \psi_{valid} &\in \left[\psi_{max} \ldots \psi_0^+\right] \cup \left[\psi_0^- \ldots \psi_{min}\right] \\ \varphi_{valid} &\in \left[\varphi_{max} \ldots \varphi_0^+\right] \cup \left[\varphi_0^- \ldots \varphi_{min}\right] \end{aligned} \tag{2}$$

$$\begin{aligned} v = c_v \frac{\psi_{valid} - \begin{cases} \psi_0^+ & : \quad \psi_{valid} > \psi_0^+ \\ \psi_0^- & : \quad \psi_{valid} < \psi_0^- \end{cases}}{\begin{array}{rcl} \psi_{max} - \psi_0^+ & : & \psi_{valid} > \psi_0^+ \\ -\psi_{min} + \psi_0^+ & : & \psi_{valid} < \psi_0^- \end{array}} \\ w = c_w \frac{\varphi_{valid} - \begin{cases} \varphi_0^+ & : \quad \varphi_{valid} > \varphi_0^+ \\ \varphi_0^- & : \quad \varphi_{valid} < \varphi_0^- \end{cases}}{\begin{array}{rcl} \varphi_{max} - \varphi_0^+ & : & \varphi_{valid} > \varphi_0^+ \\ -\varphi_{min} + \varphi_0^+ & : & \varphi_{valid} < \varphi_0^- \end{array}} \end{aligned} \tag{3}$$

During early experiments with the proposed head-joystick, we observed strong feedback effects between the translational acceleration v' of the wheelchair, and the user's head pitch angle ψ. In order to achieve a smoothed acceleration behaviour, we implemented a basic damping mechanism that replaces a velocity command v_t at a given point in time by the arithmetic mean of former velocity commands $\dot{v}_t$.

$$\dot{v}_t = \frac{1}{n} \sum_{n=0}^{n_{max}} v_{t-n} \tag{4}$$

Note that the computation of $\dot{v}_t$ as can be seen in (4), still preserves the safety issue of setting the translational velocity to zero if the actual pitch angle of the user's head lies within the defined dead zone, i.e. ψ_t does not hold to (2).

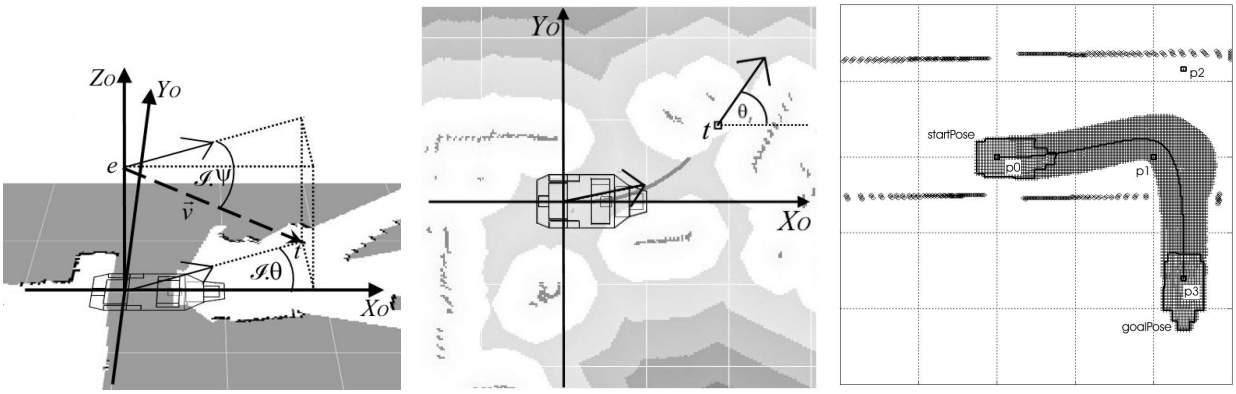
5. IMU-based Interface for Local Path Planning Algorithms

The application scenario in mind, when using the IMU as an interface-device for a geometric path planning algorithm, does no more put the operator in a continuous control loop like the implementation of a head-joystick in section 4 did. Instead, the user issues discrete driving commands by facing his/her head to a desired target-position once, and lets the system autonomously execute the given task. Within this section we first describe the deduction of a target-position by intersecting the line of the pilot's view with a local obstacle map. Furthermore we show that a suitable goal-orientation is given by a tangent lying at the target-position within a given distance map that stores the distances to the closest obstacle for each cell. After that, our geometric path planning approach is discussed. It applies cubic Bezier curves for the computation of navigable paths that connect the actual pose of the vehicle with the target-pose commanded by the operator.

5.1. Computation of Target-Pose from IMU-Reading

The first step in the computation of a target-pose from a single IMU-reading $\mathscr{I}$ is given by the computation of the target's location. For this purpose we assume a vector $\vec{v}$ based in-between the eyes of the pilot e, initially aligned with the wheelchair's heading and parallel to the ground. We now rotate $\vec{v}$ by the directed difference between the wheelchair's heading θ_{wc} and the global yaw angle of the pilot's head $\mathscr{I}.\theta$ around Z_G, or around Z_O respectively. This approach requires that the odometry coordinate system O of the wheelchair is always consistent with the global coordinate system G of the IMU. To overcome drifting errors within O, caused by wheel slippage and imprecisely modeled wheel diameter, we currently adapt $\theta_{wc} - \mathscr{I}.\theta$ manually if the user looks forward and sets up a synchronization command. Next, $\vec{v}$ is rotated by the user's head pitch angle $\mathscr{I}.\psi$ around X_G, or Y_O respectively. The resulting vector $\vec{v}$, that represents the direction of the user's view within O, is now intersected with an obstacle map. Each cell of that map describes whether a given cell in the vicinity of the robot is free or occupied by an obstacle. Under the assumption that the user hasn't looked in parallel to the ground, nor pitched his/her head upwards, the overall procedure yields an intersection point t, that serves as the aimed target position. For an illustration of this step, confer Fig.3(a).

In the second step, we augment the target position t by a desired target heading θ_t. In order to reduce the de-



(a) Obstacle map including the shape of the wheelchair, heading to the right ($\theta_{wc} = 0$). The direction of the operator's view $\vec{v}$ determines the selected target position t. The odometry coordinate system O is consistent with the IMU's global coordinate system G.

(b) Distance map including the shape of the wheelchair, heading to the right. The direction of the pilot's view is shown by the arrow based in the center of the map and rotated somewhat to the left. The target-pose (t, θ_t) is depicted by the arrow to the upper right.

(c) Obstacle map including cubic Bezier curve-based path, connecting startPose and goalPose. Control points $\vec{p}_1$ and $\vec{p}_2$ are located on straight lines that i) pass through $\vec{p}_0$ resp. $\vec{p}_3$ and ii) are aligned to the orientation of *startPose* resp. *goalPose*.

Figure 3. The pipeline of processes within the scenario of using the IMU as an interface device for a geometric path planning algorithm ranges from Fig.3(a), the derivation of a suitable target position t, over Fig.3(b), the augmentation of t with an appropriate target heading θ_t, up to Fig.3(c), the computation of an obstacle free path from the current pose of the vehicle to (t, θ_t).

mands on the user in handling his/her interface device, this step is fully automated. The idea is, that an appropriate orientation of the vehicle while driving around is almost in paralell to the surrounding obstacles. Thus we compute θ_t as the angle between the tangent to the isodistance lines in a given distance map at target position t, and the x-axis of the odometry coordinate system O. For an illustration of this step, confer Fig.3(b). It should be noted, that the current implementation selects θ_t in such a way that the target heading always points away from the current pose of the vehicle. Future implementations that should not only allow corridor navigation, but also more complicated shunting behaviours, have to differentiate at this point.

5.2. Geometric Path Planner

Almost all wheeled mobile robots that are designed to autonomously navigate in a populated and unstructured environment, use periodic sensor measurements in order to perceive their actual surrounding. Static as well as dynamic obstacles are then inserted into a local obstacle map that is big enough to cover imminent movements of the vehicle. Within that map, common navigation approaches either apply reactive behaviours like landmark tracing, or plan complete trajectories to the desired target. Prominent examples for the first class of navigation approaches are the Virtual Force Field Method (VFFM)[10], Nearness Diagramm Navigation (NDN)[11], Reactive Navigation in the Ego-KinoDynamic Space (EKDS)[12] applied with the Potential Field Method (PFM)[13]. Instances of the class of geometric path planners are given by the Dynamic Window Approach (DWA)[14], an integrated approach to goal-directed obstacle avoidance under dynamic constraints in dynamic environments (GDOA)[15], and many others.

Despite this vast spectrum of available navigation techniques that each works well within its own application scenario, we observed the need for an own development due to the kinematic restrictions of our experimental platform. *Rolland's* actuating system is given by a common differential drive at which the two actuated wheels are located at the back of the vehicle. Thus a turn on the spot command will lead to a rotation around the midpoint of the rear-axle. Considering now the task of entering a small door from within a narrow corridor, cf. Fig.3(c), it is obvious that a simple circular path that approaches *goalPose* would lead to a collision with the door's outer durn. Instead we have to model a sufficient haul-off movement that early enough brings the vehicle into a pose that is orthogonal to the passage to be crossed. From the class of behaviouristic navigation methods, VFFM, NDN and PFM would force the wheelchair in the scenario above to pass along the right-hand wall until the door frame. However, the following right turn beahaviour would lead to a collision because of the problem stated above. Similar problems

arise when employing representatives out of the class of geometric path planners. The DWA for example is well suited for circular shaped robots because it only plans one circular arc ahead when searching for a path that leads to the target. However the problem of necessary haul-off movements is left untreated.

5.2.1. Basic Spline Search-Space. Motivated by the insights above, we decided to employ a geometric path planner using cubic Bezier curves, since they are able to connect two given points while accounting for a desired curve progression and for directional requirements in the start point and end point. Considering the work of Hwang et al. [16] which gives a broad overview on approaches using this type of curve, we will now sketch our basic algorithm. Given the current pose of the wheelchair $startPose = (x_s, y_s, \theta_s)$ and the desired target $goalPose = (x_g, y_g, \theta_g)$, we search the space of cubic Bezier curves for paths that

i) connect $\vec{p}_0 = (x_s, y_s)$ with $\vec{p}_3 = (x_g, y_g)$,
ii) are smoothly aligned with θ_s in $\vec{p}_0$ and with θ_g in $\vec{p}_3$,
iii) are obstacle free in the sense that a contour of the robot shifted tangentially along the path does not intersect with any obstacle point from a given obstacle map.

Equation (5) describes a cubic Bezier curve, connecting the points $\vec{p}_0$ and $\vec{p}_3$, at a given arc length $t \in [0..1]$. In order to unequivocally determine the characteristics of the curve, we still have to chose the control points $\vec{p}_1$ and $\vec{p}_2$ such that we fulfil requirements ii) and iii).

$$\begin{aligned} \vec{p}(t) &= \vec{a}t^3 + \vec{b}t^2 + \vec{c}t + \vec{p}_0,\ t \in [0..1] \\ \text{with } \vec{c} &= 3(\vec{p}_1 - \vec{p}_0), \\ \vec{b} &= 3(\vec{p}_2 - \vec{p}_1) - \vec{c}, \\ \vec{a} &= \vec{p}_3 - \vec{p}_0 - \vec{b} - \vec{c} \end{aligned} \tag{5}$$

The computation of the free parameters $\vec{p}_1$ and $\vec{p}_2$, as can be seen in (6), spans the basic search space over the cubic Bezier curves, whose solution is intended to solve our path planing problem.

$$\begin{aligned} \vec{p}_1(l_1) &= \vec{p}_0 + l_1 \begin{pmatrix} \cos(\theta_s) \\ \sin(\theta_s) \end{pmatrix},\ l_1 max > l_1 > 0 \\ \vec{p}_2(l_2) &= \vec{p}_3 - l_2 \begin{pmatrix} \cos(\theta_g) \\ \sin(\theta_g) \end{pmatrix},\ l_2 max > l_2 > 0 \end{aligned} \tag{6}$$

Fig.3(c) illustrates the result of a basic path planning cycle. It shows the obstacle map including the integral of former sensor measurements along with the current state of the robot *startPose* and the desired target *goalPose*. The solid drawn cubic Bezier curve has been chosen as a solution in fulfillment of requirements i) - iii) that minimizes the time of travel. The upper velocity-bound of the robot at point $\vec{p}(t)$ is therefore determined by the minimal distance between the robot-contour tangentially located at $\vec{p}(t)$ to any obstacle-point, and the curvature $c(t)$.

Table 1. The comparison includes experimental data of 15 subjects that each drove an approximately $25m$ long path by using a common joystick and the IMU-based head-joystick respectively. For a discussion of the recorded data confer sec. 6.

Criterion	Common Joystick	IMU as Head-Joystick
∅ time of travel	30.73 s	55.03 s
∅ length of travel	22.45 m	25.03 m
∅ average speed	0.76 m/s	0.50 m/s
∅ safety layer interventions	111.04 ms	445.76 ms

5.2.2. Extended Spline Search-Space. Even though the computation of the basic spline search space is adequate for most of the real world navigation situations, there exist special cases that ask for a different treatment. Imagine the case where *startPose* and *goalPose* are both located on a straight line, and that further holds $\theta_s = \theta_g$. If there is now an obstacle located on the direct connection between both poses, the selection of p_1, p_2, p_3 and p_4 according to section 5.2.1 does not yield any navigable spline-based path. To overcome this situation, we introduce an extended search space, that temporarily replaces $\vec{p_3}$ by $\vec{p'_3}$ as in (7), and computes p_1 and p_2 as in (6).

$$\vec{p'_3}(l_3) = \vec{p_3} \pm l_3 \begin{pmatrix} \cos(\theta_g + \frac{\pi}{2}) \\ \sin(\theta_g + \frac{\pi}{2}) \end{pmatrix}, l_3 max > l_3 > 0 \tag{7}$$

This heuristic rule translates the target's position on a line that is orthogonal to the vector from p_0 to p_3.

6. Experimental Evaluation

Both wheelchair interfaces that were presented throughout this work have been tested in a twofold experimental evaluation. By means of a first empirical test, we compared the performance of 15 untrained participants to steer *Rolland* in an unstructured and populated office-like environment. Thereby all participants used a common joystick first, and afterwards the IMU-based head-joystick. Fig.5 gives a first look on the collected data. Both parts show the navigated paths in drift-

ing odometry coordinates, whereby we can explain the strong deviations culminating in the aimed target at the upper right part of the plots. Nevertheless, it is easy to see that the curves resulting from paths driven by the IMU-based head-joystick show oscillations particularly in sections of straight ahead movement. This indicates a problem of most of the participants, in that they couldn'd steer the vehicle on an almost straight line while moving with high translational speed. We expect this effect to be weakened by an upcoming redesign of the dead zone that currently disregards head-joystick commands from within a static intervall around the head's rest position. A dynamic roll dead zone that increases for high pitch angles, promises to neglect slight roll movements of the user's head in situations of unrestricted straight line movement.

Further data on the comparative experiment can be taken from Table 1. It is shown that the average time of travel by using an IMU-based head-joystick is about one third longer compared to the same path driven with a common joystick. The drawbacks of the head-joystick become even more evident when we interpret the average number of safety layer interventions for both types of interfaces. In section 3.2 we described the safety layer as a software module that permanently monitors the commanded translational and rotational velocity. In a situation where a combination of the desired speeds wouldn't allow for a safe breaking manoeuvre, the safety layer initiates a hard full stop. Considering the number of safety layer interventions as a valid safety metric, and the modules frequency of execution which is at least $30Hz$, Table 1 reveals that the application of the head-joystick is about four times more unsafe than the use of a common joystick.

Following the empirical evaluation that is described above, we want to prove our concept of interfacing with a geometric path planner by interpreting the user's head posture. We have chosen the format for an experimental test run instead of a comparative test series, and justify this approach because of the still too complex overall user interface. For example, the operator of the autonomous wheelchair had to give 32 verbal go-commands[3], while each time facing his/her head to the desired goal, in order to complete the path that is depicted in Fig.4. The reason for this undesired high amount of simple instructions is that a valid target-pose can currently only be located within the local obstacle map, that is $7 * 7m^2$ wide.

[3]For recognizing simple utterances like *go* for the confirmation of a selected target pose or *sync* for the alignment of the drifting odometry coordinate system with the IMU's globally correct coordinate system, we employ the of-the-shelf speech recognizer *Vocon* [17].

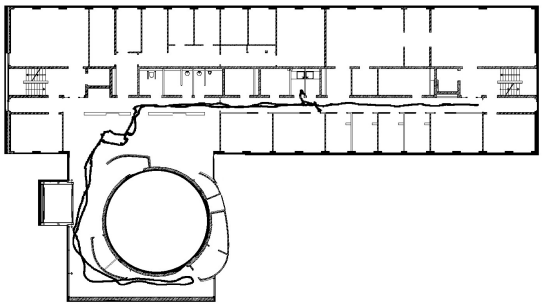


Figure 4. The depicted path has an overall length of approximately $127m$, and was estimated by a Monte-Carlo-Localisation method that is based on an implementation of the German RoboCup Team [18].

7. Conclusion

Although that we have shown the applicability of a small-size IMU as suitable controlling equipment for a (semi-)autonomous wheelchair, there persist several disadvantages in interfacing with a vehicle by head movements compared to the application of a common joystick. Future work must therefore bring up better solutions for the system's intrinsic inaccuracies like feedback effects or oscillations in steering control. Using the IMU for commanding a path planning module, we found that the number of triggering voice commands must be reduced for the sake of usability.

ACKNOWLEDGMENTS

This work has been funded by the Deutsche Forschungsgemeinschaft (DFG) in context of the Sonderforschungsbereich/Transregio 8 Spatial Cognition and by the EU project SHARE-it (FP6-045088).

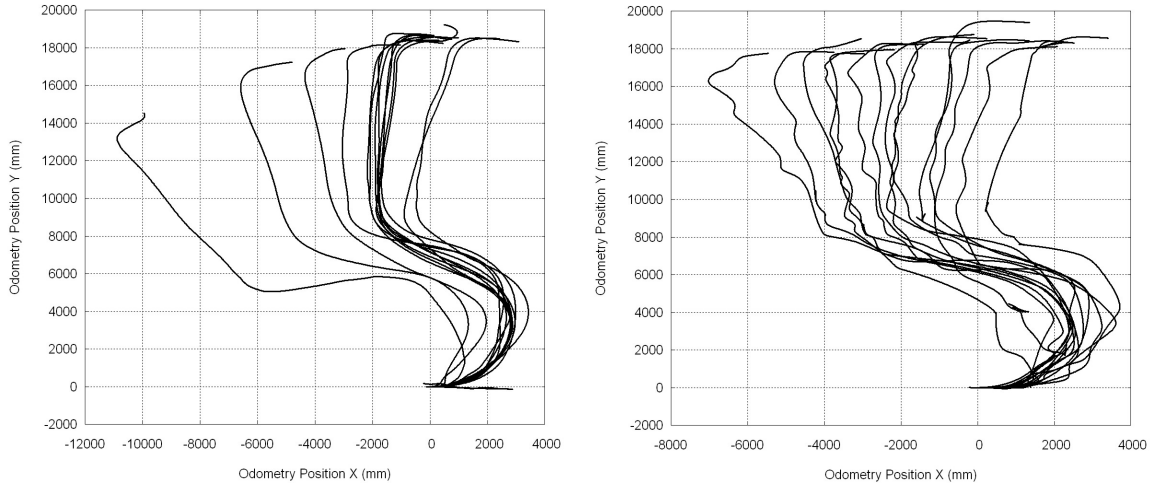
References

[1] D. L. Jaffe, "An ultrasonic head position interface for wheelchair control," *Journal of Medical Systems*, vol. 6, no. 4, pp. 337–342, 1982.

[2] J. M. Ford, "Ultrasonic head controller for powered wheelchairs," *Journal of Rehabilitation Research and Development*, vol. 32, no. 3, pp. 280–284, 1995.

[3] Y.-L. Chen, S.-C. Chen, W.-L. Chen, and J.-F. Lin, "A head orientated wheelchair for people with disabilities," *Disability and Rehabilitation*, vol. 25, no. 6, pp. 249–253, 2003.

[4] U. Canzler and K.-F. Kraiss, "Person-adaptive facial feature analysis for an advanced wheelchair user-interface," in *Proceedings of the IEEE Intl. Conf. on Mechatronics and Robotics*, 2004.



(a) Paths driven by a standard joystick.

(b) Paths driven by a head-joystick.

Figure 5. In an experimental evaluation, 15 subjects were asked to navigate *Rolland* on an approximately $25m$ long s-like shape. Both plots show the driven paths in drifting odometry coordinates, whereas the left-hand one depicts the results driven via a common joystick, and the right-hand one via a head-joystick respectively.

[5] A. Lankenau and T. Röfer, "A safe and versatile mobility assistant," *IEEE Robotics and Automation Magazine*, vol. 8, no. 1, pp. 29–37, 2001.

[6] C. Mandel, K. Hübner, and T. Vierhuff, "Towards an autonomous wheelchair: Cognitive aspects in service robotics," in *Proceedings of Towards Autonomous Robotic Systems (TAROS)*, 2005.

[7] C. Mandel, U. Frese, and T. Röfer, "Robot navigation based on the mapping of coarse qualitative route descriptions to route graphs," in *Proceedings of the IEEE Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2006.

[8] XSens Motion Technologies. (2006) Xsens inertial measurement unit product description and documentations. XSens Technologies. [Online]. Available: http://www.xsens.com/

[9] T. Röfer, H.-D. Burkhard, J. Hoffmann, M. Jüngel, D. Göhring, M. Lötzsch, U. Düffert, M. Spranger, B. Altmeyer, V. Goetzke, O. Stryk, R. Brunn, M. Dassler, M. Kunz, M. Risler, M. Stelzer, D. Thomas, S. Uhrig, U. Schwiegelshohn, I. Dahm, M. Hebbel, W. Nistico, C. Schumann, and M. Wachter, "Germanteam robocup 2004, team-report," online, Tech. Rep., 2004. [Online]. Available: http://www.germanteam.org

[10] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 5, pp. 1179–1187, - 1989.

[11] J. Minguez and L. Montano, "Nearness diagram (nd) navigation: Collision avoidance in troublesome scenarios," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 1, 02 2004.

[12] J. Minguez and L. Montano, "The ego-kinodynamic space: Collision avoidance for any shape mobile robots with kinematic and dynamic constraints." [Online]. Available: citeseer.ist.psu.edu/minguez03egokinodynamic.html

[13] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotic Research*, no. 1, pp. 90–98.

[14] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance, Tech. Rep. IAI-TR-95-13, 1 1995. [Online]. Available: citeseer.ist.psu.edu/fox97dynamic.html

[15] C. Stachniss and W. Burgard, "An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Lausanne, Switzerland, 2002, pp. 508–513.

[16] J.-H. Hwang, R. C. Arkin, and D.-S. Kwon, "Mobile robots at your fingertip: Bezier curve on-line trajectory generation for supervisory control," in *Proceedings of the IEEE Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2003.

[17] Nuance Communication Technologies. (2007) Nuance vocon speech recognizer product description and documentations. Nuance Technologies. [Online]. Available: http://www.nuance.com/vocon

[18] T. Röfer and M. Jüngel, "Vision-based fast and reactive monte-carlo localization," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.