Autonomous Navigation and Security: A 13’000h/3’000km Case Study

N. Tomatis[†‡], S. Bouabdallah[‡], R. Piguet[†], G. Terrien[†], R. Siegwart[‡]

[†]BlueBotics SA
PSE-C
CH-1015 Lausanne
nicola.tomatis@bluebotics.com

[‡]Autonomous Systems Lab, EPFL
Swiss Federal Institute of Technology Lausanne
CH-1015 Lausanne
roland.siegwart@epfl.ch

Abstract

This paper presents the design of an autonomous mobile platform and its security system. The MB835 mobile platform has been adopted for RoboX, a fully autonomous tour guide robot. In 2002, 11 of these tour guides have served the Robotics exhibition at Expo.02 (Swiss National Exhibition) from May 15 to October 20. This project has been conjointly conducted by the Autonomous Systems Lab, Swiss Federal Institute of Technology Lausanne (EPFL) and BlueBotics SA the spin-off company of the lab, which has produced the robots. The goal was to maximize the autonomy and mobility of the platform while ensuring high performance, robustness and security. The paper presents the platform, its navigation and security, which resulted in the ANT product (Autonomous Navigation Technology) and the results of the Robotics exhibition as empirical validation of the whole system.

1. Introduction

Today’s solution for the industry, the so called autonomous guided vehicles (AGVs), usually employ expensive and inflexible environment modifications for their navigation such as inductive wires in the floor or reflectors as beacons, which obligate the machine to stay on the trajectory defined a priory. Any exception due to the assumption that the environment is static and the trajectory free, result in situations, where human assistance is needed.

Nowadays, autonomous robot navigation approaches are ready for *unmodified* environments and flexible motion. However, until now, these approaches have not leaved the research labs, but for very rare occasions. When leaving the lab, even if the environment is known and accessible, a general approach requiring no environmental changes remains better suited for real-world and industrial purposes. For the same reason, a fully-autonomous and self-contained robot is preferable. Furthermore, outside the research world, such a machine is required to have a long live cycle and a high mean time between failure (MTBF), which minimizes the need of human supervision, remain efficient and cost effective.

A limited number of researchers have demonstrated autonomous navigation in exhibitions or museums [6], [12], [15], [7] and [14]. Furthermore, most of these systems have still some limitations in their navigation approaches. For instance *Rhino* [6] and *Minerva* [15] have shown their strengths in museums for one week, 19 kilometers and two weeks, 44 kilometers respectively. However, their navigation has two major drawbacks: it relies on off-board resources, and due to the use of raw range data for localization and mapping it is sensible to environmental dynamics. *Sage* [12], *Chips*, *Sweetlips*, *Joe* and *Adam* [14], use a completely different approach for permanent installations in museums: the environment is changed by adding artificial landmarks to localize the robot. This approach performed well, as shown with a total of more than half a year of operation and 323 kilometers for *Sage* [12] and a total of more than 3 years and 600 kilometers for *Chips*, *Sweetlips*, *Joe* and *Adam* [14]. However their movements, but for *Adam*, are limited to a predefined set of unidirectional safe routes in order to simplify both localization and path-planning. Another robot permanent installation which is operating since March 2000, is presented in [7]. Three self-contained mobile robots navigate in a restricted and very well structured area. Localization uses segment features and a heuristic scheme for matching and pose estimation. Another exhibition where *Pygmalion*, a fully autonomous self-contained robot was accessible on the web during one week [1] has shown its positive characteristics but, due to the unimodal characteristic of the used Extended Kalman Filter, the robot can still lose track if unmodeled events take place.

This paper presents the design of an autonomous mobile platform and its security system. Then the *Robotics* exhibition will be presented with its results for more than 13’000 hours of operation and more than 3’300 km.

In 2002, from May 14 to October 20, the Expo.02 - Swiss National Exhibition - took place on four places around the three lake region. The *Robotics* exhibition took place in Neuchâtel, where the main thematic was *nature and artifice*. *Robotics* was intended to show the increasing proximity between man and machine. The visitors interacted with up to 11 autonomous, freely navigating tour guide robots, which present the exhibit going from industrial robotics to cyborgs on a surface of 320 m^2.

2. Design

The MB835 platform has been designed for environments crowded with people and other dynamic and static objects. This requires the following characteristics for the mobile platform:

- Highly reliable and fully autonomous navigation in unmodified environments crowded of humans.
- Safety for humans, objects, and the robot itself all the time.
- Minimal human intervention and simple supervision.

This has been achieved with the following design.

The navigation platform consists mainly in a CompactPCI rack, two laser range sensors (SICKs LMS-200), the batteries, eight bumpers and the differential drive actuators with Harmonic Drive gears. The base (figure 1) has an octagonal shape with two actuated wheels on a central axis and two castor wheels. In order to guarantee good ground contact of the drive wheels, one of the castor wheels is mounted on a spring suspension. This gives an excellent manoeuvrability and stability for hardware extensions as the 1.65 m high RoboX.

The control system has been designed very carefully by keeping in mind that the safety of the humans and the platform has to be guaranteed all the time. It is composed of a CompactPCI rack containing a Motorola PowerPC 750 card, which can be extended with an Intel Pentium III card as for the control of the interaction turret of the RoboX. The PowerPC card is connected by the PCI backplane to an analog/digital I/O card, a Bt848-based frame grabber, an encoder IP module and a high bandwidth RS-422 IP module. Furthermore a Microchip PIC processor is used as redundant security system for the PowerPC card.

The result of the design is the MB835 platform: A mobile vehicle ready for the real world (figure 1).

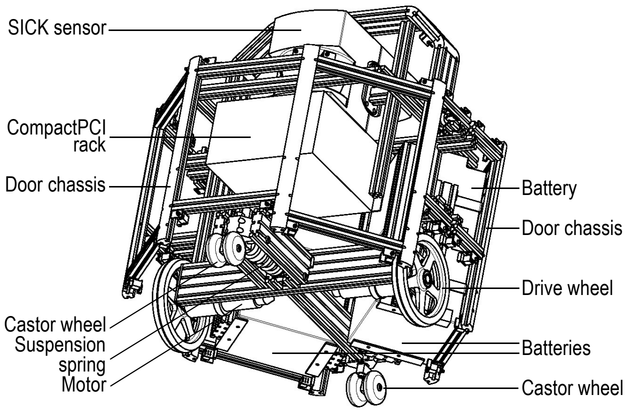

Figure 1: Mechanical design of the MB835 base.

3. Navigation

Even though the presentation of the navigation is not the main goal of this work, it is briefly described in this section as introduction to the next one: Security. This is indispensable to better understand the security requirements of the system.

The section divides the navigation task in three parts:

- Map
- Planning and Motion
- Localization

This approaches, in combination with the security, build ANT - Autonomous Navigation Technology, as it is described in figure 2.

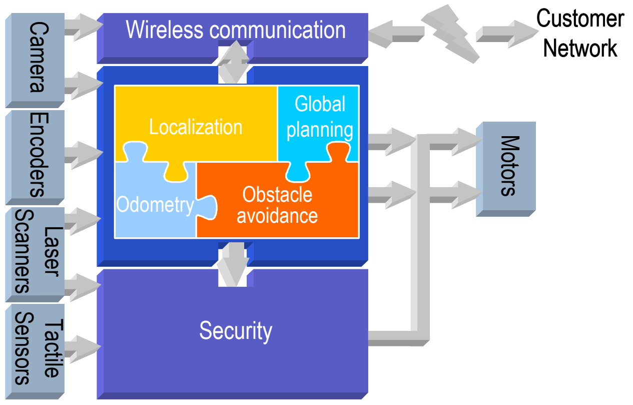

Figure 2: ANT, the Autonomous Navigation Technology is a BlueBotics product, which allows transforming any vehicle into an autonomous navigating system.

3.1 Map

The map of the environment is a graph-like structure with nodes representing $[x, y, \theta]^T$ positions the robot has to reach in order to perform a certain task. This graph is therefore used for path-planning. Furthermore it contains the information about all the features in the environment. This permits to calculate which feature is visible from the current position of the robot and to use it for localization.

3.2 Path Planning and Motion

ANT implements three path planning algorithms. They work on different levels of abstraction and take sensor readings into account in varying degrees. The topmost layer is the graph-based global planner. It is based on the above mentioned graph structure where nodes are locations of interest (e.g. a showcase, a docking station) and edges denote traversability between locations. The planner employs a depth-first search and generates a length-optimal path. Since the path is global and no sensor readings are taken into account, dynamic path modification cannot be treated on this level.

The second layer of path planning uses the NF1 navigation function in a local grid around the robot [10]. It can thus take into account the current sensor readings and is not limited to nodes of the a-priori map. However, the paths generated by NF1 have a very poor geometry, consisting of linear segments that lie on angles which are multiples of 45°. Another disadvantage is their tendency to graze obstacles.

Smoothing the path and adapting it to dynamic surroundings is done in the third layer of path planning. It is based on the elastic band [15]. The initial plan, generated by the NF1, evolves toward a smoother curve (a list of via points) as long as the elastic band does not "snap". In case dynamic obstacles move in such a way that the minimum clearance along the path cannot be maintained, or if the path lengthens beyond a reasonable amount, the NF1 is called upon again to re-initialize the path.

The motion is under control of the real-time obstacle avoidance task, which is based on the dynamic window method [7]. Using the dynamic window allows to:

- Take into account the actuator limits of the robot (speed which could result in later collisions are not allowed, motion commands never exceed the robot's speed or acceleration limits).
- Take into account the "exact" robot shape as represented by a convex polygon (extension to any polygon can be done by decomposition).

In comparison to the original dynamic window publications [7], two adaptations have been made:

- Instead of using the distance travelled before hitting an obstacle, the time until collision is used. This solves a singularity when the robot is turning on the spot (any collisions would seem instantaneous because the distance travelled seems zero). It also means the robot will choose more clearance when travelling at higher speeds.
- The objective functions for speed, heading, and clearance are calculated on the actuator phase space (v_l, v_r) instead of the usual (v, ω). Actuator limits are thus more directly taken into account.

The dynamic window task is executed with a frequency of 10Hz. It is part of the time- and security-critical processes of the navigation technology. Special attention has therefore been paid to optimize its execution time, which should be short and predictable. Both issues are addressed by the use of look-up-tables [16]: Their fixed size give an upper bound to the number of operations, and the intensive calculations (intersecting circles with line segments) can be done once at the initialization step. Adopting look-up-tables means large memory usage, especially when storing floating point values. This problem has been addressed by compressing the tables, using a Lloyd-Max quantizer. The compression is handled transparently, a fact that was facilitated by the object oriented design philosophy underlying our navigation software.

3.3 Localization

While autonomous guided vehicles (AGVs) usually employ for their navigation expensive and inflexible environment modifications such as floor tracks or reflectors as beacons, nowadays localization approaches are ready for *unmodified* environments, i.e. *natural features*.

The localization system is inspired by the experience from earlier work [1] gathered over a five year period and more than hundred kilometers travelled distance. It adds features from the localization system presented in [2]. This method is a global feature-based multi-hypothesis localization using the Kalman filter as estimation framework. It overcomes limitations of the single-hypothesis Kalman filter [7], since the data association problem is explicitly addressed. The robot preserves the typical advantages of feature-based approaches, such as very high localization accuracy and an efficient implementation and adds an important feature in the case the robot looses the track of its position: It can generates hypotheses about its current position and therefore relocate itself.

The technique which provides this property is a constrained-based search in an interpretation tree [10], [2]. This tree is spanned by all possible local-to-global associations, given a local map of observed features and a global map of model features. The same search is consistently employed for hypothesis generation and pose tracking.

4. Security

This section presents the security system, which guarantees the safety of humans, objects, and the robot itself.

All the software which relates to the movement of the robot is defined as safety critical. The safety is on three levels: The operating system, the software implementation and the redundancy of the hardware. The operating system is presented by focusing on the approaches, which have been adopted in order to make it as ergonomic as possible for the development of complex mechatronics applications. The other two sections are related to the concrete implementation of the ANT product on the MB835 mobile base.

4.1 Operating System: XO/2

XO/2 is an object-oriented, hard-real time system software and framework, designed for safety, extensibility and abstraction [5]. It is written in, and designed for the object-oriented language Oberon-2 [12]. It takes care of many common issues faced by programmers of mechatronic products, by hiding general design patterns inside internal mechanisms or by encapsulating them into easy-to-understand abstractions. The criterion by which the system has been crafted has been the careful handling of the safety aspects. These mechanisms, pervasive yet efficient, allow the system to maintain a *deus ex-machina* knowledge about the running applications, thus providing higher confidence to the application programmer. The latter, relieved from many computer-science aspects, can better focus his attention to the actual problem to be solved.

Safety, as commonly used, is a rather general notion of "the system does what it should, and does not what it should not". However, in order to analyze in greater detail how safety can actually be achieved and supported, a more formal separation of what is perceived as "safety" is required.

Szyperski [17] separated safety in the more technical terms of *safety*, *progress*, and *security*. These terms can be summarized as follows: Nothing bad happens, the right things do (eventually) happen, and things happen under proper authorization (or potentially bad things happen under proper supervision). All three interact to make a system safe in broader sense.

Safety can be enforced statically or dynamically. In some cases it is in fact possible to statically detect safety violations (or security or progress) by means of (simple) formal verification performed off-line. In other cases, safety needs to be enforced through supervised execution: If something potentially unsafe is detected, execution is stopped and proper countermeasures are taken. XO/2 addresses safety concerns through the deployment of several distinct mechanisms.

Memory safety is solved with the effort of both the programming language, which takes care of the static type-safety and a run-time enforcement system, which supervises memory accesses while isolating spurious fetches through run-time mechanisms, such as the light-weight sandboxing approach, described in [4] or the automatic real-time compatible garbage collector presented in [3].

Progress is handled by an earliest deadline driven scheduler (EDF), which allows the application programmer to specify the task's execution priority by means of its timing constraints, i.e. its *duration*, its *deadline* and its *period*. The scheduler tests new tasks for admission in the task set upon task creation, while continuously monitoring the application run-time for constraints violations.

4.2 Software Security

Tasks whose failure could cause injuries to people or damage objects required special attention during design. Build on top of the XO/2 operating system the security critical software has been designed to ensure that failures will be under complete control of the *security controller*, a real-time task running at 50Hz.

This task has the control over the amplifiers and can therefore block the robot if a critical exception takes place. It senses the speed controller and the obstacle avoidance by means of watchdogs and reads the inputs from the bumpers. Failure of one of these tasks is detected by the security controller which then either restarts the failed task or stops the robot and sends an e-mail to the maintenance.This permit to centralize the control of the security and to ask to a single object if a defect is disturbing the system.

Furthermore, the security controller generates signals for the redundant security processor. On one side it sends a watchdog signal on a digital output permitting to know if both the operating system and the security controller are still running, on the other, it acknowledges the detection and treatment of bumpers signals to the same processor.

4.3 Hardware Redundancy

In order to be robust against hardware failures, the platform has a second processor: A Microchip PIC. Its task is firstly to check the emergency button, the watchdog and a part of motor's amplifiers before enabling the motors. Furthermore, the software running on it checks the watchdog generated by the security controller, controls that the pre-defined maximal speed is never exceeded, awaits acknowledgements from the security for each bumper contact and ensures that the bumper's switch circuit is properly connected. If one of these conditions is not respected, the redundant security software on the PIC safely stops the robot (it short-cuts the phases of the motors) and puts it in emergency mode (acoustic alarm). This state is detected by the PowerPC (if it is not the cause of the failure), which will then send an email to the maintenance. The PIC also shows the kind of error with a row of LEDs and stores each error for statistics.

5. Experimental Results

The whole operational period of the *Robotics* exhibition at Expo.02, is available for statistics. This makes a total of 159 days, from May 14 to October 20, 2002. During this period, each day (9:30am-8:00pm till August, 9:00am-8:00pm in September, 9:00am-9:00pm in October) six to eleven freely navigating tour-guide robots have given tours on the exhibition's surface which is approximately 320 m^2.

5.1 Definitions

Failure: A failure is any kind of problem which requires human intervention. The only exceptions are the emergency button, which can be pressed and released also by visitors, and the situations where the robot remains blocked because it is to near too an object. In the latter case, the staff can easily displace the 115 kilograms robot by means of a switch disconnecting the motors from the amplifiers.

Uncritical: Uncritical failures are those which do not stop the task of the robot. For example, a failure consisting in a robot which stops sending an image to the supervisor is not critical for the tour the robot is giving to the visitors.

Critical: Critical failures stop the robot until human intervention is performed. An example is the failure of the scenario controller or of the obstacle avoidance.

Reboot: Critical failures requiring a reboot of the PowerPC are treated separately because they require more time before the robot is again operational.

5.2 Results

During the 159 days of operation the robots served more than 680’000 visitors for a total of 13’313 hours of operational time. In order to perform their job, they travelled 3’316 kilometers for a total moving time of more than 9’415 hours meaning that the mean displacement speed is 0.098 meters per second. As it can be seen in table 1, the uncritical failures represent 23.8% of the total amount of failures. However, they do not disturb the operation of the robot. They are therefore not treated in the following analysis which will focus on the critical and reboot failures. Furthermore, this work presents the results from the navigation platform only, not of the interaction turret, which had also a complex software. More details about the latter topic can be found in [19].

Run time	13’313 h
Movement time	9’415 h
Travelled distance	3’316 km
Speed (average / max)	0.098 / 0.6 m/s
Failures (total / critical / uncritical)	1141 / 870 / 271
Critical failures (PowerPC / HW)	694 / 176
Critical PowerPC (software / *lost*)	190 / 504
Visitors	686’405

Table 1: *Five months of operation. More than 13’000 hours of work, where the RoboXes have travelled 3’300 kilometers and served more than 680’000 visitors.*

In figure 3 all the critical failures coming from the navigation software (PowerPC) are shown. During the first three weeks, errors in the safety-critical tasks were treated by the security controller, but could sometimes require a reboot in order to restart the trapped task. This has been partly corrected allowing for much faster intervention in case of failure.

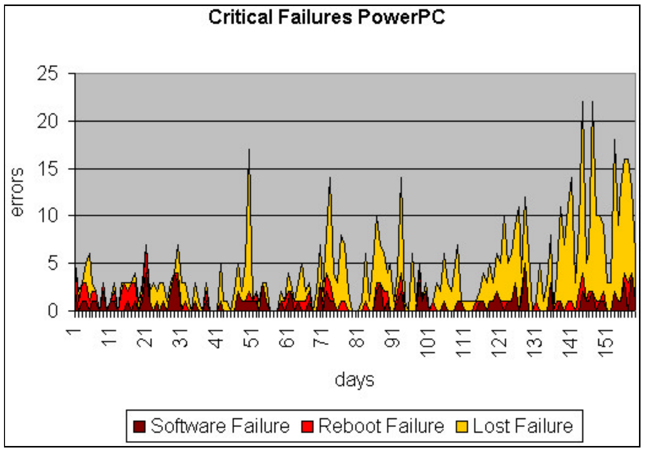

Figure 3: The critical failures of the PowerPC (navigation system). Some of the critical errors require the reboot of the PowerPC. Lost failures are not software errors but failure of the localization approach due to human misuse.

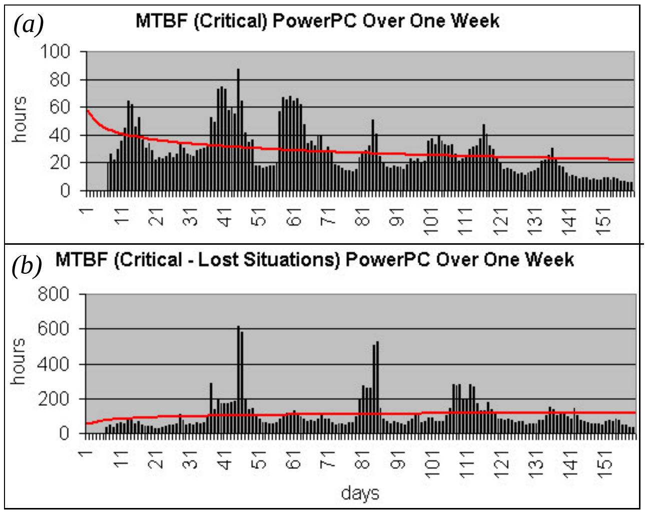

Figure 4: The MTBF (critical) with (a) and without (b) lost situations. By not taking into account the lost situations (b) the MTBF is very high (mean 70.1 hours).

Critical failures in figure 3 count errors which have not directly to do with the software: Failures of the localization system (504 meaning 72.6% of the 694 critical failures of the PowerPC) also requires human intervention. It turned out that the vast majority of the failures in the localization software were due to visitors or untrained personnel, who fiddled with the robots without disconnecting the motors from the amplifiers. This caused large errors in the odometry not taken into account in the models and, consequently, failures in the localization[1]. This explains the holes in the MTBF chart of figure 4.

The MTBF for the PowerPC (figure 4 (a)) was between 20 and 80 hours already at the beginning of the exposition. By taking into account only the software errors (figure 4 (b)), the MTBF over the whole period is 70.1 hours.

Hardware failures (figure 5) are due to some uncritical design errors at the beginning (robot doors), some motor-amplifier problems and the temperature, which was up to 35°C in the exhibit and therefore up to 90°C near the rack, between day 33 and day 40 causing some component failures.

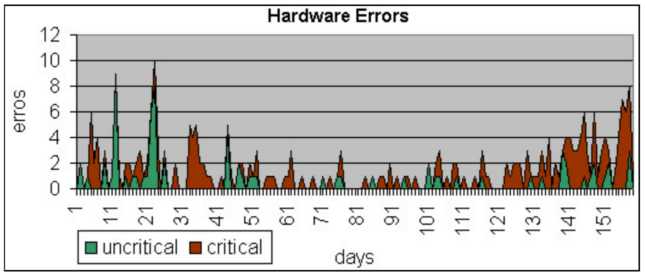

Figure 5: Hardware problems also cause critical failures.

1. The failures in the localization system are often called *lost* situations.

5.3 Lessons to be Learned

The characteristics of this project give an extraordinary chance of learning by experience. Thousands of hours of operation permit improving the software and hardware to a level which is simply not achievable with smaller projects. This was shown during the exploitation, where some errors were found after few days of operation while others appeared for the first time after one or two months. For instance, the peaks of failures in figure 3 allowed to understand that the majority of the localization errors are due to bad human manipulations of the robot. This has been discovered around day 50 by seeing a new member of the staff fiddling with the robots without disconnecting the motors from the amplifiers. However, the best example of this long term learning process are the failures of the laser scanners on week 5 due to the temperature in the exhibit. This failure wasn't taken into account by the security causing the obstacle avoidance to permanently receive the last available scan and the robot to collide with the next encountered object. This problem is since then under supervision of the security controller.

6. Conclusion

In this paper, the design of the MB835 platform has been presented with its the navigation and security approach. This approach constitutes the ANT - Autonomous Navigation Technology - product. In sections 2 through 4, it is shown how a coherent design starting from the mechanics through the electronics and finally the software, permits creating a machine, which is compatible with the real-world under the requirements of performance, robustness and security. The security issue is specifically faced with the aim of ensuring the security to the humans, the environment, the objects and the robot itself.

The experimental section presents the *Robotics* project. There, the MB835 base has been employed with an interaction turret to create a tour guide robot named RoboX. This project represents a milestone in the field of mobile robotics: For the first time 11 interactive mobile robots are produced and used for a long time (five months) as real products instead of prototypes as in former projects. The results of the whole project (159 days of operation) of the *Robotics* exhibition are presented into details for the mobile platform and its software. The analysis focuses on the amount and type of robot failures and shows the reliability and robustness of both the hardware and software.

Acknowledgements

The *Robotics* project at *Expo.02* is the result of a very strong team effort: The Autonomous Systems Lab at the Swiss Federal Institute of Technology, Lausanne (EPFL), BlueBotics SA and various other people from the academia and the industry contributed to its successful realization. The project has been funded by Expo.02 and EPFL.

References

[1] Arras, K. O., N. Tomatis, B. Jensen, and R. Siegwart (2001). "Multisensor On-the-Fly Localization: Precision and Reliability for Applications." Robotics and Autonomous Systems 34(2-3): 131-143.

[2] Arras, K. O., J. A. Castellanos, and R. Siegwart (2002). Feature-Based Multi-Hypothesis Localization and Tracking for Mobile Robots Using Geometric Constraints. IEEE International Conference on Robotics and Automation, Washington DC, USA.

[3] Brega R., F. Wullschleger (2001). A Personal Robot for Personal Robot Programmers—The Role of Automatic Storage Reclamation and Programming Languages in the Lifetime of a Safe Mechatronic System.IEEE International Conference on Advanced Intelligent Mechatronics, Como, Italy.

[4] Brega R. (2002). Safety ~~vs.~~ Speed. MSy'02 Embedded Systems in Mechatronics, Winterthur, Switzerland.

[5] Brega R. (2002). *A Combination of System Software Techniques Aimed at Raising the Run-Time Safety of Complex Mechatronic Applications*. Dissertation ETH Nr. 14513, Zürich, Switzerland.

[6] Burgard, W., A. B. Cremers, et al. (1999). "Experiences with a Interactive Museum Tour-Guide Robot." Artificial Intelligence 00(1999): 1-53.

[7] Crowley, J. L. (1989). World Modeling and Position Estimation for a Mobile Robot Using Ultrasonic Ranging. IEEE International Conference on Robotics and Automation, Scottsdale, AZ.

[8] Fox, D., W. Burgard, et al. (1997). "The Dynamic Window Approach to Collision Avoidance." IEEE Robotics & Automation Magazine: 23-33.

[9] Graf, B., R. D. Schraft, et al. (2000). A Mobile Robot Platform for Assistance and Entertaiment. International Symposium on Robotics, Montreal, Canada.

[10] Grimson W.E.L., Lozano-Pérez (1987). "Localizing Overlapping Parts by Searching the Interpretation Tree." IEEE Trans. on Pattern Analysis and Machine Intelligence 9(4): 469-82.

[11] Latombe, J.-C. (1991). Robot motion planning. Dordrecht, Netherlands, Kluwer Academic Publishers.

[12] H. Mössenböck (1995). *Object-Oriented Programming in Oberon-2*, Springer Verlag.

[13] Nourbakhsh, I., J. Bodenage, et al. (1999). "An Effective Mobile Robot Educator with a Full-Time Job." Artificial Intelligence 114(1-2): 95-124.

[14] Nourbakhsh, I., C. Kunz, and T. Willeke (2003). The Mobot Robot Installations: A five Year Experiment.IEEE International Conference on Robotics and Automation, Las Vegas, USA.

[15] Quinlan, S. and O. Khatib (1993). Elastic bands: connecting path planning and control. IEEE International Conference on Robotics and Automation.

[16] Schlegel, C. (1998). Fast local obstacle avoidance under kinematic and dynamic constraints for a mobile robot. IEEE International Conference on Intelligant Robots and Systems, Victoria, B. C., Canada.

[17] Szyperski C. A. (1992). *Insight ETHOS: On Object-Orientation in Operating Systems*. VDF.

[18] Thrun, S., M. Beetz, et al. (2000). "Probabilistic Algorithms and the Interactive Museum Tour-Guide Robot Minerva." International Journal of Robotics Research 19(11): 972-99.

[19] Tomatis, N., G. Terrien, R. Piguet, D. Buriner, S. Bouabdallah, K.O. Arras, R. Siegwart (2003). Designing a Secure and Robust Mobile Interacting Robot for the Long Term. IEEE International Conference on Robotics and Automation, Taipei, Taiwan.