Robotica (2003) volume 21, pp. 271–281. © 2003 Cambridge University Press
DOI: 10.1017/S0263574702004770 Printed in the United Kingdom

Using distributed systems in real-time control of autonomous vehicles

Urbano Nunes[†], José Alberto Fonseca[‡], Luís Almeida[‡], Rui Araújo[†] and Rodrigo Maia[†]

(Received in Final Form: November 9, 2002)

SUMMARY
In this paper distributed architectures for autonomous vehicles are addressed, with a special emphasis on its real-time control requirements. The interconnection of the distributed intelligent subsystems is a key factor in the overall performance of the system. To better understand the interconnection requirements, the main techniques and modules of a global navigation system are described. A special focus on fieldbuses properties and major characteristics is made in order to point out some potentialities, which make them attractive in autonomous vehicles real-time applications, either in terms of reliability as in terms of real-time restrictions.

KEYWORDS: Mobile robotics, Autonomous vehicles, Autonomous navigation, Fieldbus, Cost-oriented automation.

1. INTRODUCTION

Autonomous vehicles are penetrating the market and there are high expectations in that the applications of these machines will spread all over the world with strong social and economical impacts, namely in manufacturing, oceans and space exploration, medicine, highway and urban transportation. We are moving to the goal of intelligent service robots that can assist people in their daily living activities. The development of new multi-modal human-machine interfaces and new navigation systems are current research robotic trends towards human-oriented robotics. One of the main goals is to reduce human efforts to control, communicate and interact with robotic systems. Particularly, rehabilitation and assistive robotics[1–4] plays an important role in assistive technology, i.e. technology that contributes for the disabled and elderly social integration and independence with the level of dignity that they deserve.

Technologies of intelligent vehicles have also the potential to contribute to sustainable development of the cities with new approaches for improving urban mobility, complementing today's public mass transit systems, while reducing the negative effects of the private car use, like congestion and pollution (e.g. this is the goal of CyberCars[5] European project). Free-ranging autonomous vehicles, i.e. vehicles that navigate based on dead-reckoning and some technique of absolute re-positioning provided at frequent intervals, are already operational in outdoor applications, namely in driverless public transportation (e.g. the Frog Parkshuttle running at the Schiphol airport and in Rivium, Holland). This is an example of a cost-effective navigation technology because the operational costs are moderate and the investment cost on infrastructures is reduced, since the unique physical guidance requirement is the use of beacons embedded in the floor (e.g. transponder technology). Intelligent vehicles equipped with improved navigation technologies as far as concerns more advanced localization and collision avoidance systems, will offer, in the near future, a transportation solution that is safe and reliable, flexible, environmentally friendly and cost-effective.

Autonomous vehicles are complex systems requiring real-time distributed embedded control composed by multiple acquisition, processing and actuation devices. The interconnection of the distributed intelligent devices is a key factor in the overall performance of the system. The main modules of a global navigation system are conceptually described, and a special focus in fieldbus properties is made in order to point out some potentialities, which make them attractive in autonomous vehicles real-time applications.

In spite of the many currently available fieldbus solutions, future disseminations will depend on cost. In this sense, the automotive industry, involving high scale productions, will definitely contribute to the availability of efficient and low-cost fieldbuses. The fieldbuses are a key component to integrate heterogeneous subsystems with local processing for distributed control and the implementation of cost-effective autonomous systems.

In this paper the following components of real-time autonomous vehicles are addressed: The architectures of embedded systems, the communication links with emphasis on fieldbuses, and the main modules for a global navigation system. The problem of integrating all these components when using current technologies, solutions, and standards is discussed with a special emphasis on achieving real-time behaviour.

2. ARCHITECTURAL SOLUTIONS FOR EMBEDDED SYSTEMS IN AUTOMATION AND CONTROL APPLICATIONS

Embedded systems are often associated with the control of a specific machine or device. This is the case of the

† Institute for Systems and Robotics (ISR), University of Coimbra, 3030-290 Coimbra (Portugal)
‡ Department of Electronics and Telecommunications, University of Aveiro, 3810-193 Aveiro (Portugal)

https://doi.org/10.1017/S0263574702004770 Published online by Cambridge University Press

envisaged applications of mobile robots or autonomous vehicles, as it is in automotive industries. As these are complex devices, they must include several control loops. Each control loop implies a periodic sampling of sensor data, as well as, a periodic actuation operation. The sampling rate depends on the dynamic characteristics of the controlled system. Thus, different control loops in a vehicle may have different sampling rates.

Three architectures can be used to perform these control operations. One, depicted in Figure 1, consists in using a set of separated control subsystems for each controlled system. This requires, firstly, a set of specific cabling to interconnect the sensor to the controller and this one to the actuator. However, control loops are not isolated. Setpoints must be changed dynamically which means that, secondly, there must be also an interconnection to a higher level system element. This architecture has a high penalty in cabling length and a difficult coordination between subsystems.

The second architectural solution is the traditional centralized control system (Figure 2). A computational device with the adequate processing power executes the control algorithms of every controller. Connections to every sensor and actuator in the system are thus required. This architecture simplifies the coordination problem, as it is easy to schedule the operations to achieve real-time performance using well known scheduling theory and techniques.[6] However, it incurs also in a high penalty in cabling. Other disadvantages are the limited scalability and the costs of dependability.

Considering scalability, a system must be able to accommodate more and more control subsystems during the lifetime of the architectural solution, without jeopardising the performance of the ones already in operation. In this case, as the use of a complex unit is expensive, the installed processing power cannot be overestimated and thus scalability will be very limited.

Dependability is one of the important properties in safety-critical applications. One of the related issues is fault-tolerance. This architecture is not very suitable to include fault tolerance. The main reason is that the use of redundant units becomes costly, since the powerful central unit is typically expensive. Also, the handling of backups is not easy and the physical localisation of replicas is a problem, too. It is known that they must be separated to avoid simultaneous destruction as a result of a crash. This will add a huge amount of extra cabling and difficult assembly.

The third alternative is a distributed architecture using a common communication medium (Figure 3). In this approach, each control system can be divided in simpler units with different roles: controller nodes, sensor nodes and actuator nodes. The control loop functionalities may then be distributed between three or more different system nodes. Besides a significant reduction in cabling, due to the sharing of the communication medium, this distribution has several other advantages. One is the possibility to move the functional blocks to locations close to the subsystem with

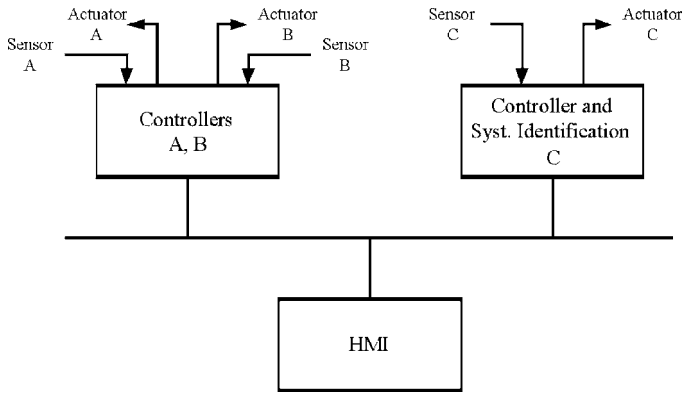


Fig. 1. Architectural solution using a set of separated control subsystems for each controlled system.

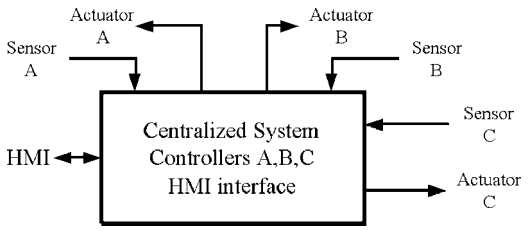


Fig. 2. Architectural solution using a centralized control system.

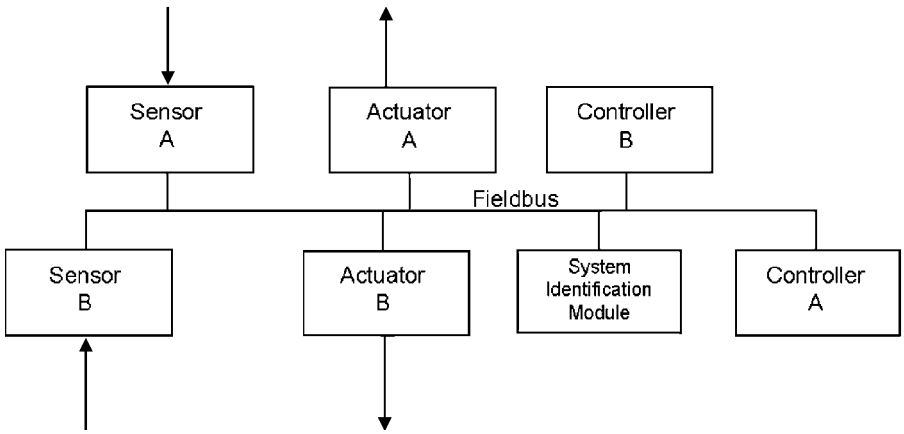


Fig. 3. Distributed architecture using a common communication medium.

https://doi.org/10.1017/S0263574702004770 Published online by Cambridge University Press

which they interact. The sensors and actuators can be equipped with processing and communications capabilities, so that they can execute certain functions locally, e.g. configuration and calibration. Other advantage is the possibility to install anywhere nodes that don't interact with external devices, like the controller node. Also, one major advantage is the possibility to achieve a modular design of the system by dividing the complexity by individual units in which some of the functionalities are encapsulated.

The use of a shared communication medium means that data can also be shared between nodes if an adequate communication model is used. In this scenario, an independent node, different from the controller, can, for instance, perform systems identification using the data exchanged between the other nodes without additional traffic in the communication medium. This, of course, requires a capability of the communication system to perform broadcast or at least multicast, which is an intrinsic feature of communication systems that follow a producer-distributor model. This model promotes the coherence of data between the different system nodes, which is also an important property for many distributed applications.

The dispersion of the complexity of the global system among much simpler nodes, when compared to the centralised approach, has also advantages from the dependability point of view. First of all, a replication of nodes is cheaper because nodes are simpler and only critical nodes must be replicated. Secondly, replication may often be at the functional level instead of being at the physical level. For example, a recovery from a failure in a controller may be solved using available processing power of a second fully operational controller or other node with adequate capabilities. Even if the full processing power is not available, the system may perform a graceful degradation of its operation by running the control subsystems at lower rates to compensate for the necessity to accommodate the execution of processes from faulty nodes.

Finally, using just a single bus for the interconnection makes much simpler the replication of the communication physical layer.

3. CONTROL AND REAL-TIME REQUIREMENTS IN DISTRIBUTED SYSTEMS

In the architectural solution of Figure 3, atomic control operations such as sensing, actuation, control algorithm execution, are distributed between different nodes. This is contrary to the architectural solution of Figure 1 in which the functionalities associated to a control loop are encapsulated within a subsystem. In this latter case, the connectivity to other subsystems is required to exchange "high-level" data such as setpoints, while in the former "low-level" data, such as samples, must also use the shared communications medium to be exchanged.

Although in both situations data conveyed is critical, in the former solution timeliness requirements are more stringent and the non-ideal characteristics of the communication medium have direct implications in the control performance.

One of the major implications from this sharing of the communication medium is that the exchange of data between subsystems is not temporally unrestricted. In fact, there are no guarantees that the transmission of data from an application to another is done in the exact instants in which the source application decided to do it because the transmission medium may not be available.

Considering this limitation, an important requirement in distributed systems is to achieve guarantees that real-time data streams or a sporadic transmission of an alarm are transmitted within a maximum admissible delay, called deadline. Also, a consequence of the mutual influence due to the different periodic data streams is that it becomes practically impossible, except when there is a very particular relationship between periods, to avoid variations in the instantaneous values of the period. This variation, known as jitter, can jeopardise the performance of control algorithms. In Figure 4 this effect of network-induced interference between data streams leading to jitter is illustrated.

In what concerns the fulfilment of deadlines of real-time data, it becomes fundamental to determine the worst case response time for each individual sporadic datum and for each periodic data stream. This response time measures the interval between the transmission request and the actual transmission instant of the particular datum. To obtain this value it is necessary to obtain a previous knowledge of the data streams set that will use the shared medium. The full knowledge may, however, be unnecessary if there is the possibility to prioritise critical data streams and if it is possible to define the maximum blocking time that can be imposed by the transmission of lower priority data.

The Medium Access Control (MAC) plays then a very important role in determining if a communication link is suitable for distributed real-time systems. It can even have a negative impact as it is the case, for example, of the Carrier

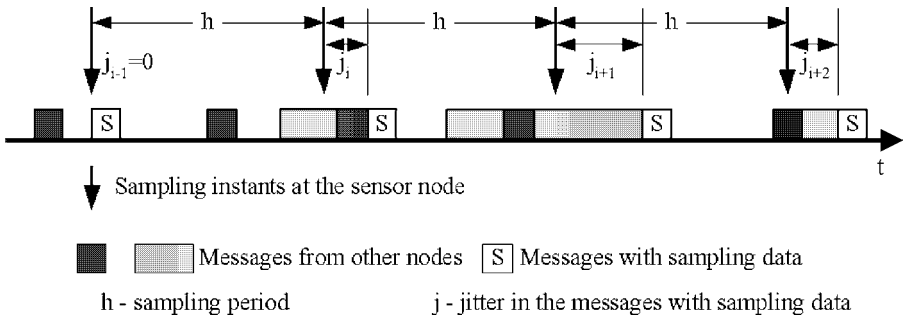


Fig. 4. Network-induced jitter.

https://doi.org/10.1017/S0263574702004770 Published online by Cambridge University Press

Sense Multiple Access with Collision Detection (CSMA-CD) MAC used in Ethernet whose non-deterministic behaviour makes difficult its use in these applications. However, considering this MAC, modifications as the ones presented by Pedreiras,[7] Jasperneit[8] and Kweon[9] may in the future overcome this limitation.

On the contrary, the MAC of certain communication systems such as Controller Area Network[10] (CAN), can show adequate characteristics for real-time applications. In this particular case, the MAC, usually called CSMA-CA, where the CA means collision avoidance, works in a fashion that makes possible to impose priorities between data streams and to determine the worst-case transmission delay if the set of possible messages is known in advance. This last issue was first addressed by Tindell and Burns,[11] who verified the timeliness of the transmission through CAN of a set of data streams needed for the control of Automatic Guided Vehicles (AGVs) using a worst case response time analysis. The message set used in the referred work was specified by the Society of Automotive Engineers (SAE) and is often used as a benchmark for fieldbus-based distributed control systems. Another study of the transmission of periodic data in CAN was done by Navet[12] using data from the Peugeot-Citroen group (PSA) vehicles.

An important issue in what concerns the medium access control is the synchronisation of the data streams in the communication link. Two main paradigms are still under debate:[13] event-triggered communication in which any system element transmits whenever it needs, i.e. depending on events independent of the communication systems, and time-triggered communication in which the transmission of messages is forced to take place in pre-defined instants and at a pre-determined rate related to the dynamics of the environment under control.

Event-triggered communication suffers from the problem of reserving resources (mainly bandwidth) under worst-case assumptions. Since events are asynchronous by nature, all of them may occur simultaneously and thus a very high amount of bandwidth is required. On the contrary, the time-triggered approach allows a relative phase control of the different data streams, reducing the number of messages that become ready for transmission simultaneously. This feature leads to the support for composability with respect to the temporal behaviour[12] thus assuring that, when two subsystems are integrated in a new system, the temporal behaviour of each of them will not be affected. The relative phase control makes it also possible to improve jitter in periodic data and to keep a higher network utilization with timeliness guarantees.

Considering average-case requirements, the network utilization of event-triggered systems is normally lower, and other types of communication with less stringent or no timing constraints (e.g. management traffic) can be supported without any additional cost.

Time-triggered communication is thus well adapted to periodic transmission of data with bounded jitter as it is the case of control applications such as motion control, engine control, temperature control, position control. Event-triggered communication is well adapted to the monitoring of alarm conditions that occur sporadically and seldom, and also to support asynchronous non-real-time traffic, e.g. for global system management. A combination of the two approaches can be very useful for safety-critical applications.

In what concerns the problem of conveying periodic control data subject to network-induced interferences there have been several issues discussed. The problem of losing samples, called vacant sampling, was analysed by Hong,[14] and the effect of jitter in the sampling and in the actuation data is analysed for different controllers by Stothert et al.[15] As far as we know, in this last work were firstly presented some illustrative designations for this phenomenon: read-in and read-out jitter. This problem can be addressed from the control point of view by taking into consideration the jitter in the system's model,[16] or from the communications point of view by minimising the network-induced interference with adequate de-phasing of the message streams,[17] which is made possible if a time-triggered approach is used.

4. COMMUNICATIONS IN EMBEDDED SYSTEMS: THE FIELDBUSES

The communication links used in embedded systems, whose characteristics and requirements were described in the previous sections, are usually known as fieldbuses. According to Thomesse,[18] the history of fieldbuses started around 1980 with preliminary experiences and working groups leading to the main fieldbus standards used today. A common meaning for fieldbus is, again according to Thomesse, "a network for connecting field devices as sensors, actuators, and field controllers as PLCs, regulators, drive controllers, and so on". So, the fieldbus is, in fact, a communication system, organised as a layered structure from the seven layers OSI model, targeted for specific applications. The first major field of application of fieldbuses has been the industrial automation domain. The main concern there was to simplify the huge amount of cabling required to automate different parts of the factories. In the early eighties different standards still used today emerged. This is the case, for example, of Profibus[19] a clear leader in industrial automation in Europe, FIP, currently named WorldFIP[20] which dominates transportation systems such as trains and CAN.

As regards CAN, the original protocol definition was mainly targeted for the use in the automotive industry. However, CAN has evolved in the direction of industrial automation due to the development of application layer protocols such as DeviceNet. This protocol made CAN one of the dominating communications technologies in these applications in the United States. It should also be noticed that this is not the only application protocol available for CAN. One can find also CANOpen, SDS, CanKingdom and others.

Another direction in the evolution of fieldbuses, in general, and CAN, in particular, is the real-time distributed embedded systems domain. In fact, fieldbuses are no longer viewed as just a simplified interconnection subsystem but they begin to be considered a subsystem in which one of the most important roles is to guarantee timeliness properties in the exchange of information between the other parts of the

https://doi.org/10.1017/S0263574702004770 Published online by Cambridge University Press

system. This view of the importance of the fieldbus in the overall distributed system led to intense research activity to find suitable solutions to achieve real-time behaviour and, also, to support the safety requirements of the target applications. The automotive industry plays here a clearly important role as a driving force for the research and development.

Besides being one of the first fieldbuses with adequate characteristics for these applications, CAN was also one of the first fieldbuses whose real-time characteristics were investigated in the context of the automotive industry, the seminal work being the already referred article by Tindell and Burns in 1994.[11] This work triggered intensive research around the use of fieldbuses in real-time applications, in general, and, in particular, in the use of CAN.

Another important step in this evolution was the appearance of the Time-Triggered Protocol[13] (TTP) for real-time communication systems. More recently, the underlying paradigm has also been addressed by the ISO Technical Committee TC22/SC3/WG1 that, in 1999, set up a task force (TF6) to work on the definition of a new CAN-based standard, Time-Triggered Controller Area Network[21] (TT-CAN), which is a time-triggered profile for the CAN fieldbus. Recently, the value of this approach was reinforced by the adoption of the time-triggered paradigm for the emerging standard for the automotive industry, the FlexRay fieldbus.[22] The importance of these and other fieldbuses for the real-time embedded systems used in autonomous vehicles is discussed later in Section 6.

5. NAVIGATION ARCHITECTURE FOR AUTONOMOUS VEHICLES

In this section, a navigation architecture capable of providing an intelligent motion control of autonomous vehicles is conceptually described in a systematic way. The architecture, presented in Figure 5, is structured in four levels: global motion planning level, local motion planning level, motion tracking level and motion control level. Parts of it have been tested in RobChair[1,2,23] prototype (Figure 6), which has the CAN-based hardware architecture illustrated in Figure 7.

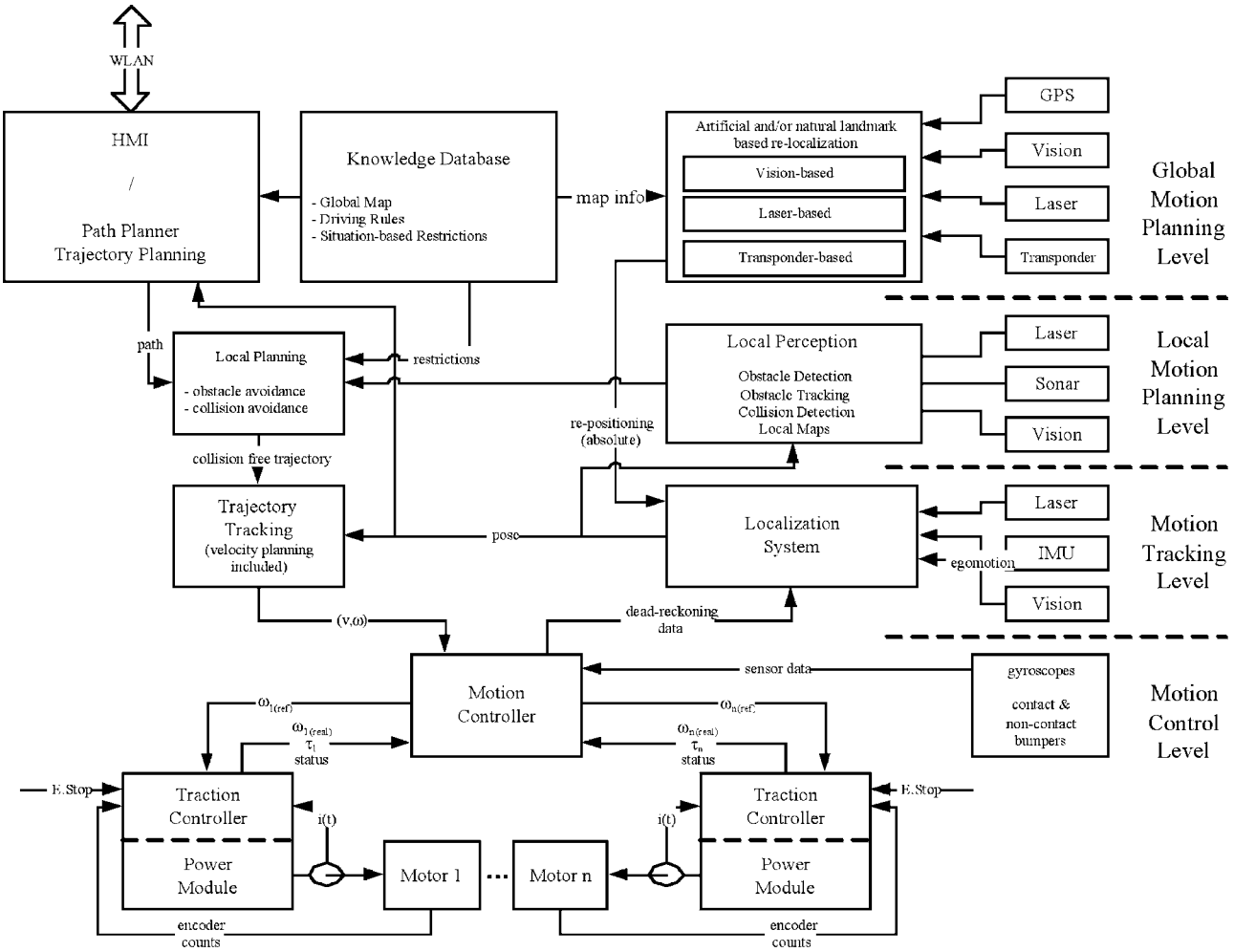


Fig. 5. Levels and modules of a global navigation system.

https://doi.org/10.1017/S0263574702004770 Published online by Cambridge University Press



Fig. 6. The RobChair prototype.

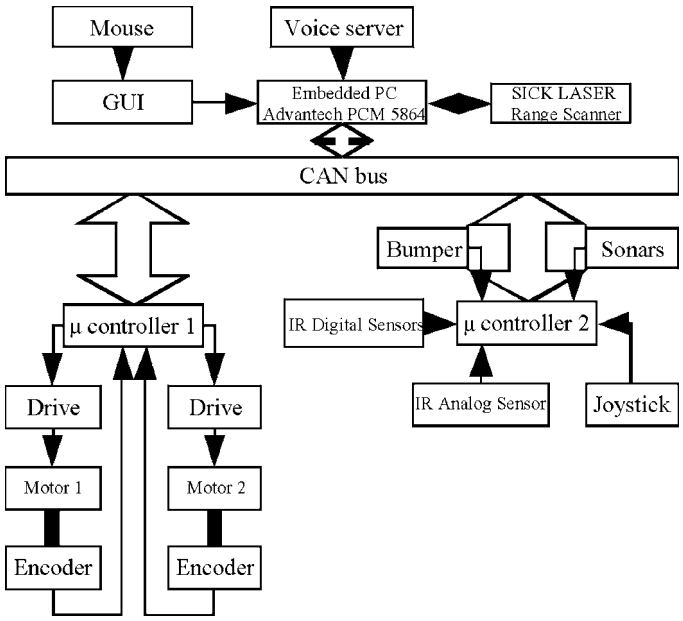


Fig. 7. RobChair CAN-based hardware architecture.

5.1. Global motion planning level

The global motion planning level is responsible for feeding the local motion planning level with a path to a predefined goal, which may be provided by a Human-Machine Interface (HMI) or a centralized task controller. For this purpose a Knowledge Database (KD) module is also considered, where information regarding the working environment, situation-based restrictions and driving rules are stored. A landmark based re-localization module is also necessary, and is considered at this level since its purpose is to determine the absolute (or global) localisation of the vehicle.

We are not introducing in this study the global map-building problem. It is assumed that an environment map exists that contains the static structure of the indoor environment, or the relevant landmarks in case of an outdoor environment. However, the vehicle needs to acquire surrounding information to cope with two problems: (a) self-localization; (b) react appropriately to the dynamic environment.

5.1.1. Re-Localization. An absolute localization system requires the existence of a set of references in the environment. These references are marks or devices that are represented in the world model by their locations. The mobile system should be able to gather information about the relative location (distance and/or angles), between the robot and a set of references. This information, when integrated with information of the environment model, yields by geometric analysis a set of restrictions about the location of the robot from which a final estimate is extracted (e.g triangulation[24] based localization). References on the environment can be active or passive. In the first case they are called *beacons* which can be devices that emit/receive signals such as grids embedded in the floor (using, for instance, optical, magnetic or transponder technologies), laser triangulation, satellites and ultra-sound. The vehicle carries on board the adequate receivers, or emitters, and its navigation system takes into account the geometric distribution of references to calculate the absolute localization.

Passive references, or landmarks, are objects or features, which the robot can perceive and recognize from the sensor data. There are two types of marks: artificial and natural. Artificial landmarks are objects or marks specifically designed to be incorporated into the environment for the navigation purposes. Landmarks with specific reflexive proprieties, geometric features of various forms[25] or bar codes that can be identified by vision sensors or laser scanning, have been used in this context. The fundamental idea of beacons and artificial landmarks is to intervene in the environment, through the *a priori* placement of references, in order to facilitate or solve the problem of establishing a world model relevant for simplifying and proceeding with the localization process. Natural landmarks are objects or features of the environment whose previous use, if any, did not include aiding the navigation of mobile system.

Absolute localization methods usually require a mechanism to optimize the correspondence between a local map and a global map of the environment (e.g. see Yamauchi et al.,[26] Duckett et al.[27]).

5.2. Local motion planning level

The local motion planning level is responsible for providing the motion tracking level with a collision free trajectory in the reference path.

For a dynamic environment the robot may need to deviate from its path or at least it can be forced to change its

https://doi.org/10.1017/S0263574702004770 Published online by Cambridge University Press

velocity not following the prescribed value. A local navigation control layer is required to endow the autonomous robot with the capability to react appropriately to the dynamic environment, which may include obstacles moving nearby with unknown trajectories. The perception model at this level is in charge of maintaining a local environment model. This model, essentially composed from sensory-based data, should contain relevant and updated information that enable the robot to navigate safely avoiding collisions.

The local planning module may have connections with the path planner, KD, local perception module and trajectory-tracking controller. When obstacles appear that can disturb the normal travel of the robot the control loop becomes closed at the local navigation level. The robot can exhibit one of two main classes of behaviours: collision avoidance or obstacle avoidance. In the first group we include the autonomous driving behaviours that contribute to the avoidance of collisions keeping the vehicle on its path. Otherwise, the robot may experience obstacle avoidance behaviour if it is allowed to navigate out of its prescribed path.

A collision avoidance module is a crucial technology, for instance, to equip intelligent vehicles running autonomously in outdoor environments following a planned path. In this case appropriate collision avoidance architecture could be based on a perception module in charge of multiple object tracking and object classification, and a decision module that mimics human driver behaviour in different driving situations, like free-flow, following leading vehicle, braking and approaching. Presently, research is carried out to produce perception modules, having inputs, range and visual data provided by radar, laser and stereovision. The data fusion of these multisensor modalities[28] is crucial for extracting relevant and robust information from raw data.

Since the initial works on obstacle avoidance methods based on artificial potential fields,[29] many methods have been proposed. In the potential field methods the robot is treated as a *free-flying object* moving under the influence of an artificial potential field, with obstacles exerting repulsive forces on the robot and the goal asserting an attractive force. The geometry and position of obstacles in the surroundings, as perceived by the robot, typically define the environment model for these methods. No information regarding the obstacles dynamic state is derived. Most obstacle avoidance methods make a similar use of the local information, i.e. object tracking is not performed. Examples of these methods include the Vector Field Histogram[30] (VFH), the Dynamic Window[31] (DW), the Elastic Band,[32] the Nearness Diagram[33] (ND), and strategies implemented by Fuzzy Systems.[2]

In the VFH, the motion direction and velocity commands are computed from a histogram description derived from an occupancy grid representation of the local environment. An empirical threshold is applied in the transformation and the VFH does not take into account the width of the robot. DW-based approaches rely upon a model of the local obstacles from which the distance to collision along with circular arcs are calculated giving rise to the dynamic window velocity space. Kinematics and dynamic constraints are taken into account, but the solution (velocity commands) depends on a set of three weighting coefficients heuristically assigned. These coefficients are used to balance the weights of the goal heading, the forward robot velocity and the obstacle clearance. The ND follows a situated-activity methodology by using five laws of motion adapted to five situations that describe the relative state of the robot, obstacle distribution, and goal location. In the perception step, the current situation is identified by looking for information regarding the environment, like regions of free space.

A different framework is proposed by the Velocity Obstacle[34] (VO) approach, in which potential collisions and the collision-free path of a robot are determined, based on information of the obstacles positions and velocities.

A local navigation architecture is under research,[35] which embeds a set of behaviours and that, in addition to the use of local maps[36] and obstacle tracking information, also take into consideration a set of driving rules and situation-based restrictions. This set of driving rules and restrictions, if applied to all vehicles sharing the same environment, make the vehicle navigation safer and more efficient, due to the increase of the behaviour predictability of the other vehicles sharing the same space. However, the level of difficulty is higher than usual, since it is mandatory that the robot has the real-time perception capacity of detect, track and classify the objects in its surroundings.

5.3. Motion tracking level

The motion tracking level determines the velocity commands to be sent to the motion controller, based on the error between the reference trajectory provided by the upper levels and the real trajectory.

5.3.1. Localization. Navigation systems can be categorized as positioning or dead-reckoning systems. A positioning system determines the vehicle's pose regardless of the traveled path. Dead-reckoning systems keep calculating its pose with respect to a previous determined pose. Concerning their flexibility, navigation systems can also be categorized as rigid or free-range. A free ranging system is not dependent of a rigid track, allowing the vehicle to follow flexible paths. Positioning systems[37] provide absolute localization with bounded error but without the required precision and, worst, without the desired frequency compatible with its use in a closed-loop real-time control system. On the other hand, dead-reckoning has accurate short-term measurements, but its precision degrades with time due to integration errors. Because absolute positioning and dead-reckoning complement each other, it is common practice to fuse the data provided by both systems in applications calling for continuous high accuracy and reliability. An example is the fusion of dead-reckoning data produced by IMU (Inertial Measurement Unit) with DC-GPS (Differential Carrier GPS).[38]

Other methodologies make use of range sensors data,[39,40] like lasers or cameras, which can also produce estimates about vehicle displacement, by matching 2D range scans. The time-to-time re-localisation data provided by the upper

https://doi.org/10.1017/S0263574702004770 Published online by Cambridge University Press

level that performs global localization through landmark detection is used to limit the growth of the integration error.

5.3.2. Path Tracking. The global or local planning level provides the planned path as a list of reference points and velocities. The input variables for the path-tracking are the desired velocity, present vehicle's pose and the next desired vehicle's pose (see Figure 8). The trajectory tracking includes a velocity planning module that calculates the desired velocity as a function of the reference velocity defined by upper levels and some constraints such as the curvature radius of the trajectory (this information can be provided by the trajectory planner). From the pose error the linear and angular velocities are computed and provide the motion control level.

Several techniques have been proposed for the trajectory tracking control of nonholonomic vehicles, such as: Kanayama controller;[41] Yang controller;[42] controllers based on fuzzy-logic;[23] sliding mode based controllers, and controllers using nonlinear control theory.[43]

5.4. Motion control level

The motion control level is responsible for the robust velocity servo-control. The motion control architecture here presented is based upon two different modules comprising two different control laws. At the lowest level a traction control module is used to control each wheel velocity. These modules use the encoder information and the torque developed by the motors to perform feedback control that drives the wheels to their reference velocities. The torque information is important at this level to reduce the wheel slippage.

The upper level (the motion controller in Figure 5) receives the desired velocity vector from the trajectory tracking module and sends the reference velocities to each traction module. For this purpose it uses inverse kinematics of the vehicle to calculate the reference velocities of both wheels. As a complement to this classical approach, dynamic models can be here applied in order to improve control robustness to wheel slippage, bumps and slopes. With the proper dynamic models and information collected by the traction modules about the torque developed by each motor, reliable estimates of the vehicle slippage and stability can be achieved and used to improve control.

At this level it is also required to perform the dead-reckoning calculations. These calculations often employ distinct types of proprioceptive sensors, e.g. odometers like optical encoder on wheels and gyroscopes.[4] By applying appropriate integration of measured quantities, the vehicle displacement can be estimated. Nevertheless, the values achieved by these techniques are noise contaminated and the estimates are subject to accumulate drift errors.

6. USING FIELDBUSES IN THE CONTROL OF VEHICLES

The control architecture described in the previous section has been partially implemented by many researchers referring to a set of heterogeneous subsystems targeted to a specific role. Several examples of these subsystems involve, for instance, the autonomous wheelchair prototypes.[2,3] These subsystems process distinct critical levels of operations, involving many complex tasks, such as multisensor data acquisition and synchronisation, which have to exchange information among them. For this purpose, it is usual to install heterogeneous communication interfaces, such as serial links (RS232), Ethernet, and fieldbuses.[2,3] The interaction between the different subsystems becomes then cumbersome to implement, requiring gateways that make it difficult to achieve timeliness, when the data is exchanged between different communication systems. The target applications, autonomous vehicles and specific mobile robots, will then benefit from using a distributed architecture based on a single fieldbus, such as the one presented in Figure 3 of Section 2. In fact, there have been recent examples of using fieldbuses in similar applications.[28,44–46]

Often, the use of a fieldbus is just targeted to simplify the interconnection of subsystems or to promote modularity by encapsulating every function in simple dedicated nodes. However, an integrated view of the distributed embedded control system is required to promote its general dissemination. In this sense, the automotive industry, involving large-scale productions, will definitely contribute to the availability of low-cost and efficient fieldbuses.

When one looks to the automotive industry nowadays, one can realize that most distributed control systems just

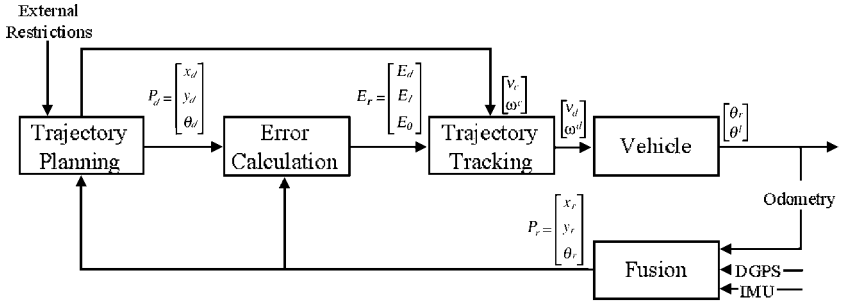


Fig. 8. Block diagram of a trajectory tracking controller.

https://doi.org/10.1017/S0263574702004770 Published online by Cambridge University Press

enhance the performance of conventional mechanical systems (e.g. ABS, assisted steering). However, in the near future, car manufacturers want to remove all mechanical links between driver and actuators, relying on the distributed system only to perform car operations (X-by-wire). The distributed system must then be considered safety-critical and thus, capable of operating in the presence of faults. There are already some prototypes running such as the SKF's Filo X-by-wire system and a Mercedes vehicle presented at the 2002 FlexRay seminar in Munich.

In this scenario, the tendency of the fieldbus protocols used in the automotive industries, analyzed later in this section, indicates a clear move towards the already discussed time-triggered (TT) paradigm. However, there is also a requirement to support sporadic/aperiodic communications whose transmission instants cannot be determined *a priori*. This is typical of alarms, and thus a fast response is often required. Several compromises must then be made to include special mechanisms to handle this type of event-triggered communication in a time-triggered system.

Regarding the control architecture in Figure 5, we may identify multiple periodic streams of information, namely: (i) motion commands from the local planning level to the trajectory tracking module and from this module to the motion controller; (ii) dead-reckoning data from the motion controller to the localization module; (iii) actual pose estimates from the localization module to the trajectory-tracking module; (iv) laser data from sensor to different perception modules. Also there is a need for supporting aperiodic streams such as alarms, e.g. an unexpected imminent collision situation detected at the local perception module from laser or ultrasound sensors should reach urgently the motion controller.

An important issue in these distributed embedded systems is the flexibility. This feature can be related to several functionalities such as the ability to support live insertion of nodes, the capacity to support on-line mode changes and download of new application software or configuration data and the ability to support dynamic communication requirements. As regards this last point, event-triggered communication systems are typically considered flexible because they react promptly to communication requests that can be issued at any instant in time. In other words, they react to instantaneous (variable) communication requirements. On the other hand, time-triggered systems are not so flexible because communication takes place at pre-defined instants only.

Examples of the importance of flexibility in terms of dynamic communication requirements can be found in situations triggered by an operator or by the system autonomously. For instance, the operator, through the HMI interface, can change operational parameters, e.g. control parameters, or even high level behaviours, e.g. path to be followed. Also, depending on the external environment, certain situations may require higher data rates from particular sensors than from others. For example, obstacle detection when there are no obstacles in a long range may require a lower scanning rate than when the vehicle approaches obstacles. Another example is tracking a given path. When the vehicle is moving with a steady low tracking error, the tracking sensors can be sampled at a lower rate. Upon error increase, a higher sampling rate can be re-established for improved control performance.

However, this flexibility can have a strong impact on dependability. One can easily understand that a change in communication requirements can possibly lead to a network overload and consequent timing failures. It is then mandatory to limit the extent of such changes so that the timely behaviour of the network is guaranteed.

Notice that the discussion on dependability is also relevant for the applications discussed in this paper since these systems, autonomous vehicles, will interact strongly with humans, which makes the safety problem a critical one. In this aspect, the modularity obtained with distributed systems facilitates the introduction of redundancy as it was discussed in Section 2. Moreover, the time-triggered communication paradigm inherently supports the detection of connectivity failures at the receiver because message arrival times are known. On the contrary, in event-triggered systems, e.g. CAN without any upper layer, a fail-silent failure in one node will not be immediately detected by the remaining nodes. To allow remote detection of failures, such systems must use a heartbeat, which is a timed mechanism using coarse synchronisation. An overview of safety critical issues in automotive networks can be found in a special issue of IEEE Micro,[47] including discussions about TTP,[48] Flexible Time-Triggered Controller Area Network[49] (FTT-CAN) and FlexRay protocols.

Looking now to the situation in the automotive industry, three fieldbuses can be considered the main candidates to win the adoption war: CAN, TTP/C (Time-Triggered Protocol) and FlexRay. CAN, which has been used until now in cars, is not inherently a TT-based protocol. However, it is possible to add upper layers that allow for TT behaviour. This is the case of TT-CAN,[21] which resulted from a standardization effort under ISO, and FTT-CAN[50] developed at the University of Aveiro, Portugal.

The widespread use of one of these protocols will depend obviously on commercial interests but also on their efficiency in transmitting various types of data and on their dependability characteristics. In what concerns the first issue, event-triggered data, generated by alarms, and time-triggered data, associated to control loops, must be supported. The handling of dynamic communication requirements also seems to be an issue of growing importance. As it can be seen in the comparison made by Almeida et al.,[50] it is difficult to determine which of these protocols is the best. In fact, when there is a better performance in one aspect in one of the protocols, e.g. TTP/C in time-triggered traffic and TT-CAN in event-triggered, there will be weaknesses in other aspects. Also, when there is sometimes some advantage in important aspects such as flexibility, as in FTT-CAN, it lacks the industrial backup.

As regards dependability issues, the FlexRay consortium is claiming a lot of interesting features for their fieldbus, mainly derived from the redundancy of the physical interconnection and the specificity of bus guardians. However, this is already available or can also be obtained in other protocols, such as TTP. Considerable discussion on

https://doi.org/10.1017/S0263574702004770 Published online by Cambridge University Press

these issues is thus expected, considering the interest in the implementation of X-by-wire solutions.

7. CONCLUSIONS

Fieldbuses will soon play the role of an important and critical component of the distributed embedded computer control system in charge of the autonomous vehicles operation. The current trend towards highly distributed architectures in the control system will continue to increase the responsibility of the fieldbus in determining important system properties, e.g. timeliness, flexibility, scalability, dependability and maintainability. Autonomous vehicles moving in unknown environments require the system capacity to respond to sudden changes in the operating conditions. Besides other requirements, this calls for accommodating on-line variable data streams, i.e. for a flexible communication.

The automotive industry is moving to a widespread use of X-by wire systems and to rely in the distributed embedded system for the control of vehicles. The results of the technological development and of further research on the communications issue will certainly lead to a fieldbus protocol that will become widely used in several applications, including autonomous vehicles, due to availability and cost reasons.

ACKNOWLEDGEMENTS

This work was partially supported by EU (European Commission) project CyberCars (IST-2000-28487) and FCT (Portuguese Science and Technology Foundation) project DISCO (POSI/34244/SRI/2000).

References

1. G. Pires, R. Araújo and U. Nunes, A.T. de Almeida, "RobChair – A Powered Wheelchair Using a Behaviour-Based Navigation", *Proc. of 5th IEEE Int. Workshop on Advanced Motion Control*, (Coimbra, Portugal, 1998), pp. 536–541.
2. G. Pires and U. Nunes, "A Wheelchair Steered Through Voice Commands and Assisted by a Reactive Fuzzy-Logic Controller", *International Journal of Intelligent and Robotic Systems* **34**(3), 301–314 (July, 2002).
3. S. Tzafestas (Guest Editor), "Research on Autonomous Robotic Wheelchairs in Europe", *Reinventing the Wheelchair, Special Issue of IEEE Robotics & Automation Magazine* **7**(1), 4–6 (March, 2001).
4. E. Prassler, J. Scholz and P. Fiorini, "A Robotic Wheelchair for Crowded Public Environments", *IEEE Robotics & Automation Magazine* **7**(1), 38–45 (March, 2001).
5. "CyberCars: Cybernetic Cars for a New Transportation System in the Cities, Annex 1 – Description of Work", *European Project proposal for the Information Society Technologies Programme (IST-2000-28487)*, (European Union, 2001)
6. G. Butazzo, *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications* (Kluwer Academic Publishers, Boston, 1997).
7. P. Pedreiras, L. Almeida and P. Gai, "The FTT-Ethernet protocol: merging flexibility, timeliness and efficiency" *Proc. of the Euromicro Conference on Real-Time Systems* (Vienna, 2002) pp. 152–160.
8. J. Jasperneit and P. Neumann, "Switched Ethernet for Factory Communication", *Proc. of 8th IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA2001)* Antibes, France (2001) pp. 206–213.
9. S-K. Kweon and K.G. Shin, "Achieving Real-Time Communication over Ethernet with Adaptive Traffic Smoothing", *Proc. of IEEE Real-Time Technology and Applications Symposium* (2000), pp. 206–213.
10. Bosch, *CAN specification version 2.0 – Tech. Report Bosch GmbH*, (Stuttgart, Germany, 1991).
11. K.W. Tindell and H. Hansson, "Analysing Real-Time Communications: Control Area Network (CAN)", *Proc. IEEE Real Time Systems Symposium* (1994), pp. 259–263.
12. N. Navet and Y.Q. Song, "Performance and Fault Tolerance of Real-Time Applications Distributed over CAN (Controller Area Network)" *CiA – CAN in Automation Research Award*, (1997).
13. H. Kopetz, "Event-Triggered versus Time-Triggered Real-Time Systems", *Lecture Notes in Computer Science* (1991), **Vol. 563**, pp. 87–101.
14. S. Hong, "Scheduling Algorithm of Data Sampling Times in the Integrated Communication and Control Systems", *IEEE Transactions on Control Systems Technology* **3**(2), 225–230 (1995).
15. A. Stothert and I. MacLeod, "Effect of Timing Jitter on Distributed Computer Control System Performance" *Proc. 15th IFAC Workshop on Distributed Computer Control Systems (DCCS'98)* (1998) pp. 25–29.
16. A. Mota, P. Fonseca and J.A. Fonseca, "Systems Modelling and Identification in CAN based Distributed Control Systems", *Proc. 16th IFAC Workshop on Distributed Computer Control Systems (DCCS'2000)* (Sydney, Australia, 2000) pp. 99–105.
17. F. Coutinho, T. Barreiras, J.A. Fonseca and E. Costa, "Jitter Minimisation with Genetic Algorithms", *Proc. 3rd IEEE International Workshop on Factory Communication Systems (WFCS'2000)* (Porto, Portugal, 2000) pp. 267–273.
18. J.P. Thomesse, "The Fieldbuses", invited tutorial, *Proc. of IFAC Symposium on Intelligent Components for Control Applications (SICICA'97)* (Annecy, France, 1997) pp. 13–23.
19. K. Bender, (editor) *PROFIBUS – The Fieldbus for Industrial Automation* (Prentice Hall, 1993).
20. P. Leterrier, "The FIP Protocol" *WorldFip Europe*, (France, 1992).
21. T. Führer, B. Müller, W. Dieterle, F. Hartwich, R. Hugel and M. Walther, "Time-triggered Communication on CAN (Time-triggered CAN-TTCAN)", *Proc. of the 7th International CAN Conference (ICC'00)* (2000) pp. 09-02–09-08.
22. R. Belschner, J. Berwanger, C. Ebner, S. Fluhrer, T. Führer, F. Hartwich, B. Hedenetz, R. Hugel, A. Knapp, J. Krammer, P. Lohrmann, B. Müller, M. Peller and A. Schedl, *FlexRay Requirements Specification – Version 1.9.7.* (FlexRay-Consortium, 2001).
23. L. Conde, G. Pires and U. Nunes, "A Behaviour Based Fuzzy Control Architecture for Path-Tracking and Obstacle Avoidance", *Proc. 5th Portuguese Conf. on Automatic Control (CONTROLO2002)* (Aveiro, Portugal, 2002) pp. 341–346.
24. M. Betke and L. Gurvits, "Mobile Robot Localization Using Landmarks" *IEEE Transactions on Robotics and Automation* **13**(2), 251–263 (April, 1997).
25. L. Feng, Y. Fainman and Y. Koren, "Estimation of the Absolute Position of Mobile Systems by an Optoelectronic Processor", *IEEE Transactions on Systems, Man, and Cybernetics* **22**(5), 953–963 (1992).
26. B. Yamauchi and R. Beer, "Spatial Learning for Navigation in Dynamic Environments", *IEEE Transactions on Systems, Man, and Cybernetics* – Part B Cybernetics **26**(3), 496–505 (1996).

https://doi.org/10.1017/S0263574702004770 Published online by Cambridge University Press

27. T. Duckett and U. Nehmzow, "Mobile Robot Self Localization and Measurement of Performance in Middle-Scale Environments", *Robotics and Autonomous Systems* **24**, 57–69 (Elsevier, 1998).
28. C. Coué, Th. Fraichard, P. Bessiére and E. Mazer, "Multi-Sensor data Fusion Using Bayesian Programming: an Automotive Application", *Proc. IEEE/RSJ International Conf. on Intelligent Robots and Systems (IROS'02),* (Lausanne, 2001) pp. 141–146.
29. O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", *The International Journal of Robotics Research* **5**(1), 90–98 (1986).
30. J. Borenstein and Y. Koren, "The Vector Field Histogram (VFH) – Fast Obstacle Avoidance for Mobile Robots", *IEEE Transactions on Robotics and Automation* **7**(3), 278–288 (1991).
31. D. Fox, W. Burgard and S. Thrun "The Dynamic Window Approach to Collision Avoidance" *IEEE Robotics & Automation Magazine* **4**(1), 23–33 (1997).
32. M. Khatib, "Sensor-Based Motion Control for Mobile Robots", *PhD Thesis* (LAAS-CNRS, December, 1996).
33. J. Minguez and L. Montano, "Nearness Diagram Navigation (ND): A new real time collision avoidance approach", *Proc. IEEE Int. Conf. on Intelligent Robots and System (IROS'00)*, (2000) pp. 2094–2100.
34. P. Fiorini and Z. Shiller, "Motion Planning in Dynamic Enviroments using Velocity Obstacles", *The International Journal of Robotics Research* **17**(7), 711–727 (1998).
35. D. Castro, U. Nunes and A. Ruano, "Reactive Local Navigation" (to appear in *Proc. 28th IEEE Industrial Electronics Conference (IECON'02)* (Sevilla, 2002)).
36. J.L. Cruz and U. Nunes, "Generating Local Maps for Mobile Robots", *Proc. 4th Portuguese Conf. on Automatic Control (CONTROLO2000)* (Guimarães, Portugal, 2000) pp. 412–417.
37. J. Lobo, L. Marques, J. Dias, U. Nunes and A. de Almeida, "Sensors for Mobile Robot Navigation", *Autonomous Robotic Systems*, A. de Almeida and O. Khatib (eds) (Springer-Verlag, 1998) pp. 50–81.
38. J. Farrell, T. Givardis and M. Barth, "Real-Time Differential Carrier GPS-Aided INS", *IEEE Transactions on Control Systems Technology* **8**(4), 709–721 (July, 2000).
39. S. Roumeliotis and J. Burdick, "Stochastic Cloning: A Generalized Framework for Processing Relative State Measurements", *Proc. IEEE Int. Conf. on Robotics & Automation* (2002) pp. 1788–1795.
40. F. Lu and E. Milos, "Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans" *International Journal of Intelligent and Robotic Systems* **18**(3), 249–275 (March, 1997).
41. Y. Kanayama, Y. Kimura and F. Miyazaki, "A Stable Tracking Control Method for a Non-Holonomic Mobile Robot", *Proc. IEEE/RSJ Int. Workshop on Intelligent Robots and Systems* (1991) pp. 1236–1241.
42. S.X. Yang, G. Yuan, M. Meng and G. S. Mittal, "Realtime Collision-Free Path Planning and Tracking Control of a Nonholonomic Mobile Robot Using a Biologically Inspired Approach", *Proc. IEEE Int. Conf. on Robotics & Automation* (2002) pp. 3402–3407.
43. N. Sarkar, X. Yun and V. Kumar, "Control of Mechanical Systems with Rolling Constraints: Application to Dynamic Control of Mobile Robots", *The International Journal of Robotics Research* **17**(1), 55–69 (1994).
44. P. Pomiers, "CEL-Based Integration Policy for Critical Multi-Layered Aplications", *Proc. IEEE Intelligent Vehicle Symposium (IV'2002)*, (on CD only) (2002).
45. J. Gil, A. Pont and G. Benet, "A CAN Architecture for an Intelligent Mobile Robot", *Proc. of IFAC Symposium on Intelligent Components for Control Applications (SICICA'97),* (1997) pp. 65–67.
46. M. Wahl and G. Dorée, "A CAN Application Layer for an Experimental Real-Time Obstacle Detection Study", *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems (ITSC'00)* (2000) pp. 276–281.
47. P. Koopman, "Critical Embedded Automotive Networks", *IEEE Micro Special issue on Critical Embedded Automotive Networks* **22**(4), 14–18 (July, 2002).
48. R. Maier et al., "The Time Triggered Architecture – A Consistent Computing Platform", *IEEE Micro Special issue on Critical Embedded Automotive Networks* **22**(4), 36–45 (July, 2002).
49. J. Ferreira, P. Pedreiras, L. Almeida and J.A. Fonseca, "The FTT-CAN protocol: improving flexibility in safety-critical systems", *IEEE Micro Special Issue on Critical Embedded Automotive Networks* **22**(4), 46–55 (July, 2002).
50. L. Almeida, P. Pedreiras and J.A. Fonseca, "The FTT-CAN Protocol: Why and How", to appear in *IEEE Transactions on Industrial Electronics* **49**(6), 1189–1201 (Dec, 2002).

https://doi.org/10.1017/S0263574702004770 Published online by Cambridge University Press