In: IEEE International Symposium on Industrial Electronics (ISIE'97), Guimarães, Portugal, July 7-11, 1997

Parallel Processing Approaches In Robotics

Dominik HENRICH and Thomas HÖNIGER

Institute for Real-Time Computer Systems and Robotics
University of Karlsruhe, D-76128 Karlsruhe, Germany
e-mail: [dHenrich, Hoeniger]@ira.uka.de

***Abstract* – This paper presents the different possibilities for parallel processing in robot control architectures. At the beginning, we shortly review the historic development of control architectures. Then, a list of requirements for control architectures is set up from a parallel processing point of view. As our main topic, we identify the levels of parallel processing in robot control architectures. With each level of parallelism, examples for a typical robot control architecture are presented. Finally, a list of keywords is provided for each previous work we refer to.**

1. INTRODUCTION

In spite of significant improvements in processing speed, sequential processors are far from rendering sufficient computing capacity for an advanced robotic system. On the other hand, modern VLSI technology offers a unique opportunity to close this gap by parallel computing. One could object that highly parallel computers do not serve as a conceivable platform for robotics due to their high cost and limited availability. However, it can be expected that the progress in the design of new VLSI circuits and the reduction in component cost will make the highly parallel machines new available very economical. Probably in the next decade, it will be possible to build parallel computers with relatively low costs.

Today's sequential computers may be sped up only through intensive technological effort since the performance is physically limited by present architectures. High computational parallelism is one solution to this problem. By adding processing units in parallel computers, the process time can be arbitrarily sped up for corresponding complex problems. On the other hand, the available computational parallelism has to be exploited in an efficient way. The solution methods from different applications can be parallelized in various ways. An improvement in performance cannot be achieved by solely increasing the number of processing units because the time necessary for communication or additional data administration may increase simultaneously.Thus, an important task is the parallelization of existing problem solutions in robotics so that they are suitable for highly computational parallelism. In several cases, fundamentally new algorithms have to be designed, so that a parallelization is feasible. Specially designed computer architectures for robotic control are surveyed in [33]. Several parallel robot control architectures have been suggested, however, which can be distinguished by different levels of parallelism that are presented in the main section of this paper.

For automated manufacturing, the historical development of control structures can be followed [21]. It ranges from the central control to the distributed control. In each of these control structures, the control components are separated from the manufacturing components and are interconnected by their control interrelationships. For parallel processing, each control component can be regarded as a single processing element (PE) (see Section 3.7).

For robot control architectures, a classification scheme has been proposed in [38]. It covers the extreme viewpoints of the historical development, hierarchical and distributed control. Additionally, function-oriented and behavior-oriented approaches are distinguished. Altogether, this results in four different classes. For parallel processing, each function or each behavior can be performed by an extra PE (see Sections 3.5 and 3.6).

The rest of the paper is organized as follows: First, we elaborate on the requirements for general control systems with emphasis on parallelism in Section 2. Then, as the main section, we distinguish eight different levels of parallel processing in robotic control architectures in Section 3. For each level, a definition, some examples and an evaluation according to the requirements are given. Finally, after a summary of results in Section 4, a list of references with a list of keywords corresponding to the parallelization levels is appended.

2. REQUIREMENTS TO CONTROL ARCHITECTURES

Before discussing parallel control architecture, it is important to explain what a control architecture is. After a short definition, we will continue explaining the requirements for the control architecture.

According to [21], a control *architecture* makes a control *system* from control *components*. The architecture determines the interrelationships between the component and the mechanisms for coordination. The architecture is a crucial point for a system, because it establishes the limitations and possibilities for changing the system in the future.

Requirements on robot control architectures can be described from a general point of view [26], for manufacturing systems [21], and for software architectures of robot control [29, 11]. Important requirements from the parallel processing point of view include:

Robustness: Robustness of a system is perceived as the ability of the system to handle imperfect inputs, unexpected events, uncertainties, and sudden malfunctions [21]. The system, for which a failure in a subsystem implicates a break down of the whole system, is not robust. This is, for instance, the case for systems built on the pipeline principle.

Modifiability / scalability (off-line): A system is said to be modifiable if changes by adding, modifying or removing elements of the system may be easily made. In this paper, we focus on a special type of element, the processing element, so that we pay a particular attention to the scalability (off-line) of the system as it is defined in [43]: The scalability of a parallel system is a measure of its capacity to increase speedup in proportion to the number of PEs. It reflects a parallel system's ability to utilize off-line changing resources effectively.

Adaptability / scalability (on-line): The robot is able to manage its internal resources on-line according to the external circumstances. In our case, this could concern the on-line mapping of the tasks which have to be fulfilled onto the computing resources.

Reactivity to the environment: The reactivity refers to the capability of detecting events and acting within a short period of time, depending on the context. It is quantitatively measured by the response time. One aim of parallel processing is to achieve short response times.

Resolving of multiple goals: In most cases, situations involving conflicting concurrent actions are inevitable. The control system should provide functions to achieve those multiple goals [26]. Sometimes, the multiple goals can be achieved by multiple tasks, which may be processed in parallel.

Programmability: Usually, complex control systems are partitioned in multiple (parallel) components for simpler handling. In this case, programming the single components becomes simpler, but interrelationships become more complex.

3. PARALLEL PROCESSING LEVELS

We now focus on the parallel processing approaches used to meet the requirements of Section 2. We show to what extent these methods have been applied, and in which cases they are advantageous and why.

First, it is necessary to remark that there are no throughoutly parallelized architectures available for robot control. Only single areas have been regarded for parallel processing. This leads us to distinguish the following eight levels of parallel processing in robot control architectures: multirobot level, robot level, kinematics level, control level, functions level, behaviours level, abstraction level, algorithm level.

In the next subsections, for each level, we will give a general definition, present a typical example followed by other examples, and conclude how we can take advantage of parallelism according to the requirements in robot control, especially what the scalability (on-line), modifiability (off-line), and robustness concerns.

3.1 Multirobot Level

For many tasks, for instance, when the problem is very complex or of a large scope (exploration mission), it is often advantageous to use several robots instead of a single one. The conventional, strictly centralized control method has to deal with many problems, such as a communication problem due to the huge amount of information to be processed. Another problem is that the strictly centralized coordination or scheduling of the robots is very difficult in an unforeseeable environment. These problems can be solved by giving the simultaneously working robots some independence, or by parallelizing the problem. Many approaches are possible involving the interaction between the robots (degree of dependence, homogenous versus heterogeneous robots, communication complexity).

For example, at one extremity, one finds a decentralized structure with non-cooperative robots (non-advanced communication), whose interactions result in emergent global behavior. In [22], this emergent behavior is used to perform the material handling requirements in a workcell (see Fig. 1). The processing machines (cutting machine, assembly machine, …) broadcast load or unload messages to the listening swarm robots. These machine-material handling requirements are satisfied by the available robots, which work in parallel without central planning and without communicating with each other. Thus, no modification to the swarm material handling system is required while the workcell environment changes (addition or deletion of robots or machines). This implies to robustness and high adaptability. But this system is subject to deadlocks and is less efficient than centralized systems due to the limitiation to solely local decision capabilities [22].

Other examples of the swarm robots model in nature are the immune system (in [53]) and a colony of ants (in [60]). [50] showed the global performance variations of the colony by modifying the number of robots and introducing low level communications among them. More complex autonomous robots are able to cooperate and partition the global task [4]. Local communication among the robots is sometimes sufficient, whereas global communication can be advantageous for heterogeneous robot and optimization problems [59]. A more centralized approach is presented in [63] with robots which have to obey a master. The robots independently plan and execute their own tasks, which introduces time uncertainty and makes the scheduling problem of the centralized master non-trivial. Holonical architectures for manufacturing multirobot workcells allow the robot to negotiate on the task with the scheduler [10]. In [52], this holonic architecture is compared with the hierarchical and heterarchical ones in terms of robustness and efficiency. Other work concerning parallel multirobot systems concentrates on the interprocess communication in an industrial context [58, 31].

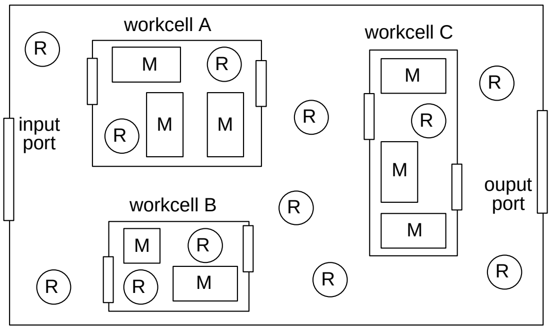

Fig. 1: Parallelism on the multirobot level: Swarm robots in a workcell [22] (M: machine, R: robot).

As a conclusion, one can adopt two extreme viewpoints (swarm robots / centralized architecture), where the advantages of one extreme constitute the disadvantages of the other. Swarm robots offer adaptability, robustness, extensibility and reactivity, but lack in efficiency, whereas centralized architectures are suitable for optimization, but are not flexible, adaptable, expandable or robust enough.

3.2 Robot Level In this level, a hardware-oriented point of view is taken. Each robot component, such as manipulator, endeffector, or overhead camera, has its own PE, which can be used for different functions. The modularized components are computationally independent from each other and are controlled in parallel.

For example, the real-time controller of the Karlsruher Autonomous Mobile Robot (KAMRO) is divided into independent subcontrollers for each component: manipulators subcontroller, vehicle subcontroller, camera subcontroller [19]. Each subcontroller has independent PEs communicating through a VMEbus (see Fig. 2). The communication between the subcontrollers through VMEbus couplers makes cooperation between the components possible, such as mobile manipulation [55] or manipulation supported by hand cameras [39].

A similar architecture, ASTRA, is proposed for a space robot testbed with redundant arms [23]. It is based on a VMEbus multiprocessor system for each component (arm 1, arm 2, motion based) and a VME-VME bus adaptor between the components. Unfortunately, the extensibility of high-speed multiprocessor bus architectures is limited due to the communication overhead. Another alternative, a multiprocessor controller based on a point-to-point architecture, which can be extended without performance degradation, was tested on a multicomponent system [5]. Three PEs for the first arm, two PEs for the second arm, and one PE for the camera were used. In [16], a manipulator is controlled by two decoupled behavior-based controllers, one for the arm and one for the hand, implemented on a set of eight PEs. The micro-robot MINIMAN also uses one PE to control the right manipulator, a second one for the left manipulator and a third for the legged mobile platform [49].

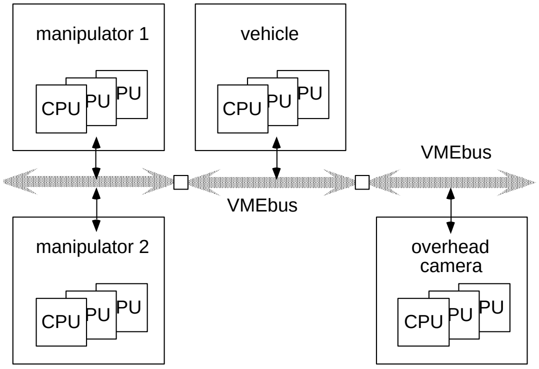

Fig. 2: Parallelism on the robot level: The autonomous mobile manipulation system KAMRO

By allocating an extra PE to each robot component, the robot components can work in parallel. This reduces the execution time, e.g., by positioning the endeffector while the robot moves. But temporarily unused components, e.g., an immobile manipulating vehicle has unused PEs, which indicates low scalability.

3.3 Kinematics Level

In many cases, the high degree-of-freedom of a manipulator makes it impossible for one PE control fast enough. One option is to decompose the main control loop into several control loops. For each joint of a kinematics chain (e.g., base, shoulder, elbow, and three joints for end-effector orientation) an extra loop may be provided, which is associated with a single PE.

For example, the walking machine LAURON has six legs with three DOF each and is controlled by 24 microprocessors [17]. Each joint has its own PE, on which an appropriated small feedback loop (sensor, control, effector) is implemented (see Fig. 3).

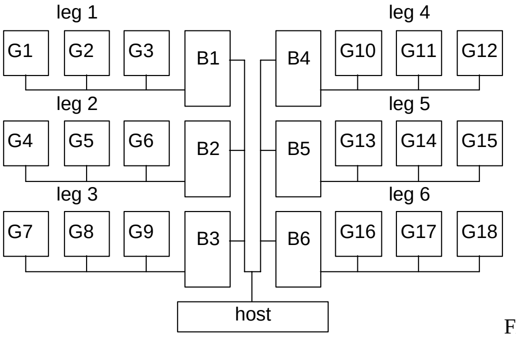

Fig. 3: Parallelism on the kinematics level: The 6-legged walking machine LAURON [17]

The redundant manipulator described in [6] moves in a two- dimensional space and has seven DOF. Each joint has its own PE. The keyboard player robot WABOT, presented in [61], is provided with 50 PEs to control the 50 DOF. A reconfigurable modular manipulator system was developed in [57], where each joint corresponds to a hardware module which can be added or removed, increasing or decreasing the number of degrees of freedom of the manipulator.

Parallel processing at the kinematics level has two advantages. First, the computer hardware architecture reflects the robot hardware architecture which provides a clear overview of the system, and makes it easier to develop and to debug. Second, these schemes are statically extensible. Thus, with an appropriate scalable algorithm (such as described in [68, 66]), an additional joint could easily be controlled by adding another PE.

3.4 Control Level

In order to guarantee the stability of the controlled system, a high sample rate is often required. In complex cases, a single PE cannot achieve the aspired timing. The control task has to be broken down into simpler subtasks which are small enough to be performed by a single PE. This task can be partitioned at the control level by pipelining the functions of the control loop.

For example, the controller of the three fingered Karlsruher Dextrous Hand requires a sample rate of 10 kHz. In order to cope with this high demand, the approach adopted in [48] splits up the control loop of one finger into single functions (sensing, controlling, acting, coordinates transformation). Each of these functions is processed by a sparate PE (see Fig. 4).

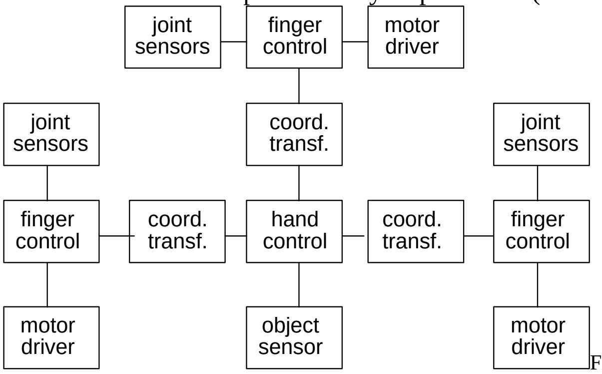

Fig. 4: Parallelism on the control level: The Karlsruher Dextrous Hand [49]

In [66], the control loop is parallelized according to the manipulator joints and according to the functions of the control loop (pipeline principle). Each joint has its own closed loop control and each of them is divided into functions, which are processed on different PEs.

The advantages of parallelization at the control level are mentioned in [25]. Each processor can be specialized to its own job (a special function of the control loop), by adding appropriate co-processors. Another advantage is that the input and output functions are separated from the algorithmic functions, and the programmer can concentrate on the algorithm. Also, the hardware architecture provides a clear overview of the functionalities of the system, which makes the development easier.

3.5 Functions level

In this section, we focus on the functions as they are defined in [38]: perception, planning, execution, exception handling. Each function is provided with a processor, so that on this level, the different functions (or tasks) of a robot are processed in parallel.

For example, the mobile autonomous robot YAMABICO was tested with an architecture based on centralized decision making and distributed functions such as: locomotion control, sensor information, inter-robot communication and world map database [42]. Each function is independently modularized and implemented on a different set of Transputers (see Fig. 5). The functions work in parallel and communicate through a dual port RAM, which can be accessed asynchronously from the other modules. The master module can also send interrupts through the dual port RAM.Different blackboard systems, which facilitate highly parallel design approaches, are presented in [62]. The car NAVLAB uses five modules (global planning, local planning, perception, mission execution and hardware control) and communicates via a parallel blackboard [32]. KAMRO uses a hybrid distributed system to implement a functional decomposition of control [46]. The Ground Surveillance Robot (GSR) is used as a platform for sensor fusion techniques with a parallel blackboard [34]. In [45], the coordination and integration of several real-time activities occurs via a blackboard for mobile robot navigation. Generalizing the concept of logical sensors developed in [36], which have their own computing capabilities, the robot HILARE presented in [11, 29] uses independent modules on its functional level. For this robot, the onboard partition of the functional modules on the different PEs is shown in [56].

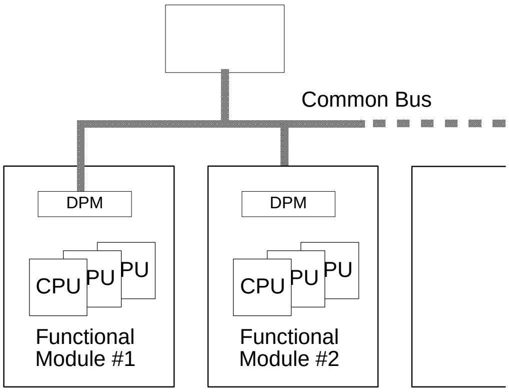

Fig. 5: Parallelism on the functions level: The vehicle YAMABICO [42] (DPM: Dual Port Memory)

High extensibility (a functional module of YAMABICO can be easily removed, replaced or added) and dynamic reconfiguration (the functional modules of HILARE are dynamically linked) characterize such architectures in general.

3.6 Behaviors Level

A behavior is a relatively simple sensor-effector connection which makes the robot react to a given sensor input like a stimulus-response reflex. The global behavior of the robot is a result of the interactions between these independent behaviors (wander, explore, avoid collisions), providing the controller with a high degree of parallelism.

For example, the subsumption architecture introduced in [7, 8] represents one method to arbitrate the different behaviors. The behaviors work in parallel but have different priorities. The high-level behaviors can inhibit or subsume the behaviors of the lower levels. Brooks successfully applied this subsumption architecture to different mobile robots with three behaviors: avoid collision, wander, explore (see Fig. 6).

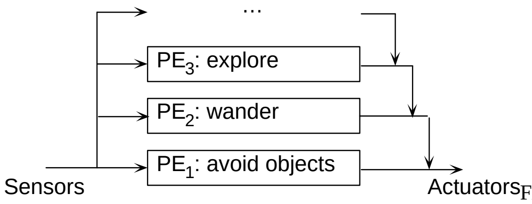

Fig. 6: Parallelism on the behavior level: The subsumption architecture similar to similar to [8]

A subsumption architecture for a manipulator has been implemented on 24 PEs [9]. An arm (including hand) was controlled with 15 independent behaviors (extend, stop, open, deposit, …) running concurrently on a set of eight loosely coupled microprocessors [16]. But for more complex cases, such as airplane control, difficulties arise due to the conflicts between the behaviors [35].

In [20], the robot ROBBIE has a subsumption-like control architecture, where the priorities between the levels vary. Several methods are proposed to avoid the deadlock problem for a behavior-based architecture, such as adding learning capacities [67] or adding an adaptative level [65]. Another behavior-based architecture, based on very simple behaviors called ”Schemas”, are presented in [3, 47]. A manipulator based on these Schemas is described in [12].The overall complexity emerges from the parallel actions of these independent simple behaviors.

The behaviors level has the advantage of being easily parallelized (for instance, one processor per a level), high robustness, high reactivity (by suppressing the chain sensor-model-planning-action), high extensibility potential (for new competence, one just has to add processors) and the possibility of incremental development. A higher level can be added after testing the functionalities of the lower levels. The main problem is to find the different behaviors and arbitrate them, so that the robot can finally perform the task the user wants.

3.7 Abstraction Level

The control architecture of robots is often divided into levels according to the degree of abstraction of processed data and response time. These hierarchical architectures vary from centralized architectures (tree structures) to layered architectures, with communication capabilities within the layer and between two adjacent layers. In the first case, parallelism can be introduced by using one PE for each node of the tree structure according to the pipeline principle. In the second case, the layers, which have very different response times, are considered to work simultaneously and are quasi-independent from each other, and thus can be implemented on different PEs.

For example, the control architecture of the two-armed mobile robot KAMRO [19] is a conventional hierarchical architecture with several layers. The response times between two neighboring levels are sufficiently different (ratio of one to ten) to consider them to work in parallel, and can thus be easily implemented on different PEs (see Fig. 7).

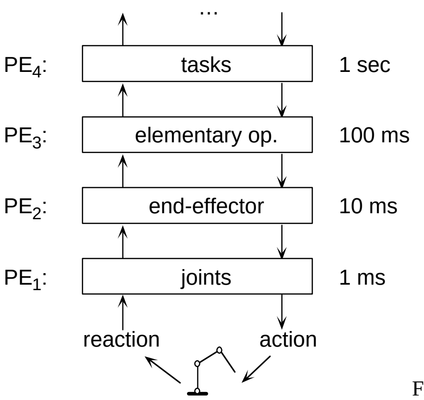

Fig. 7: Parallelism on the abstraction level: A hierarchical control architecture for manufacturing environments

Other hierarchical architectures implemented on multiprocessor systems have been investigated [24, 54]. A shorter response time at the lower levels was reached by adding connections across the architecture [2, 51] which improved the reactive capacity of the architecture. In [15], a three-layered hierarchical architecture ANIMA (Architecture for Natural Intelligence in Machine Applications) which exhibits reflexive behaviors is implemented on five processors, and can be easily extended with other processors. In addition to single hierarchies, a twin hierarchy for navigation and perception [18] and a three-layered architecture with double parallelism [44] were used. The tasks of one layer can be executed simultaneously and synchronously. Different levels work in parallel and asynchronously. Intercommunication problems between the different layers of a hierarchical robot controller are analyzed in [14]. Special architecture and development environments well suited to hierarchical control systems are described in [41, 1].

Although hierarchical structures in general offer high efficiency and can optimize problems, they have to deal with communication problems, the computational bottlenecks, the difficulty of integrating additional sensors, the reaction capacity (messages have to go through several layers before reaching the actuator) and the robustness (due to the pipeline principle, if a bug occurs in a level, the whole structure breaks down).

3.8 Algorithm Level

At this level, the single algorithms of a robot system are parallelized. The aim is to speed up the algorithms which need a huge amount of computation in order to satisfy the required real-time constraints. Previous work is flourishing in this area, especially in the following four fields: image processing, motion planning, kinematics and dynamics.

On the one hand, image processing techniques offer well parallelized algorithms and appropriate hardware. A good overview on parallel robot vision algorithms is provided in [13]. On the other hand, motion planning algorithms have long execution times and are a critical point for closing the control loop made up of sensing, planning, and acting. A review of parallel processing approaches to motion planning is given in [37].

In Robot joint control, i.e. in kinematics and dynamics computation, there are the most severe time constraints of robot control architectures. The computational power of a single PE is not sufficient to control a manipulator with several DOF. A survey of parallel processing approaches to robot kinematics is given in [40]. Parallel approaches to dynamics are given in [27, 69, 28]. Additionally, there are very specific architectures combining CORDIC processor arrays and DSPs, e.g., in [64].

Additionally, general computing architectures, which are independent of the algorithms to be tested, which have powerful communication systems using message passing, are developed in [30]. In [1, 41], two modular architectures, using tightly and loosely coupled subsystems are developed.

Results obtained by parallelizing algorithms vary. It depends on the degree of dependency among the equations. Image processing problems can be broken down quite well by dividing the image into smaller independent blocks, whereas kinematics or dynamics algorithms contain coupled equations, which lead to a communication overhead when parallelizing.

4. CONCLUSION

One promising method to master the complexity of a system such as a robot consists of breaking down the system into independent subsystems. These subsystems can then be easily mapped onto parallel processing elements.

In our first step, we recalled the requirements for robot control architectures, especially from the parallel processing viewpoint. Then, we presented the eight levels at which this system partitioning occurs in current robot applications: multirobot level, robot level, kinematics level, control level, functions level, behaviors level, abstraction level and algorithm level.

As a conclusion, one can say that there are no thoroughly parallelized architectures available for robot control. For the given approaches, most of the following statements are valid:

- Only separate areas have been regarded for parallel processing. These areas can now be easily distinguished by the parallelism levels.
- The approaches are only scalable within one level. For example, in the kinematics level, only joints can be easily added, adding functions may result in a complete re-design of the architecture.
- Some levels often occur in a mixed form, e.g., component / functions level or abstraction / control level, but this is not necessarily the case

The presented work may not serve as an orthogonal classification scheme for parallel robot control, but it is certainly useful for making the (potential) parallelism in existing control architectures more distinct. Additionally, the different levels of parallelism can help to increase parallel processing in future robot control architectures. This again will lead to scalable architectures, shorter response times, and easier programming of the robot systems.

ACKNOWLEDGEMENTS

This research work was partially supported by the Deutsche Forschungsgemeinschaft (DFG) with a stipend by the "Graduiertenkolleg" of the Computer Science Department, University of Karlsruhe. The work was performed at the Institute for Real-Time Computer Systems and Robotics, Prof. Dr.-Ing. U. Rembold and Prof. Dr.-Ing. R. Dillmann. Many thanks to Christian Wurll, Petra Bohner, and Michelle Specht for proofreading.

REFERENCES

For each reference, a list of keywords is given. Most of the keywords indicate the level of parallelization. Multiple levels indicate that parallel processing is performed at more than one level. As additional keywords, we use *introduction* and *survey*.

[1] Al-Mouhamed M.: ”Multiprocessor system for realtime robotics applications”, Microprocessors and Microsystems, 1990, June. *Keywords:* algorithm level, abstraction level.

[2] Albus J. S., McCain H. G., Lumia R.: ”NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM)”, NIST Tech. Rep. 1235, 1989. *Keywords:* abstraction level.

[3] Arkin R. C.: ”Motor Schema Based Navigation for a Mobile Robot: An Approach to Programming by Behavior”, IEEE Int. Conf. on Robotics and Automation, 1987. *Keywords:* behavior level.

[4] Azarm K., Schmidt G.: ”Decentralized Motion Planning for Multiple Mobile Robots”, Intelligent Autonomous Systems, U. Rembold et al. (Eds.),IOS Press, pp. 302-309, 1995. *Keywords:* multirobot level.

[5] Bar-on D., Gershon D.: ”A Multi-Processor Robot Controller”, Comp. Euro. Proc. Computers in Design, Manufacturing and Production, Prix Erry, pp. 88 - 93, France, 1993, 24-27 May. *Keywords:* robot level.

[6] Bohner P.: ”A multi-agent approach with distributed fuzzy logic control”, Intelligent Robots and Systems, 1994. *Keywords:* kinematics level.

[7] Brooks R., Connell J., Flynn A.: ”A synchronous distributed control system for a mobile robot”, MIT Artificial Intelligent Lab, USA, 1986. *Keywords:* behavior level.

[8] Brooks R., Connell J., Flynn A.: ”A mobile Robot with onboard Parallel Processor and large Workspace Arm”, Nat. Conf. on Artificial Intelligence, pp. 1096-1100, Philadelphia, 1986, Aug.. *Keywords:* behavior level.

[9] Brooks R. A.: ”A Hardware Retargetable Distributed Layered Architecture for mobile Robot Control”, IEEE Int. Conf. on Robotics and Automation, pp. 106-110, 1987. *Keywords:* behavior level.

[10] Brussel: ”The Vision matching the Problem”, First European Conf. on Holonic Manufacturing Systems, Hannover, Germany, 1994, 1 Dec.. *Keywords:* multirobot level.

[11] Camargo R. F., Chatila R., Alami R. A.: ”A Distributed Evolvable Control Architecture for Mobile Robots”, ICAR, Int. Conf. on Advanced Robotics, pp. 1646-1649, Pisa, Italy, 1991, June. *Keywords:* functions level, multirobot level.

[12] Cameron J. M., Mackenzie D. C., Ward K. R., Arkin R. C., Book W. J.: ”Reactive Control for Mobile Manipulation”, IEEE Int. Conf. on Robotics and Automation, pp. 228-235, 1993, May. *Keywords:* behavior level.

[13] Chaudhary V., Aggarwal J. K.: ”Parallelism in computer vision: A review”, in: ”Parallel algorithms for machine intelligence and vision”, Springer-Verlag, New York, 1990. *Keywords:* algorithm level.

[14] Chen N., Parker G. A.: ”Design of a Robot Control System Architecture”, Microprocessors and Microsystems, vol. 18, 1994, July/ August. *Keywords:* abstraction level, control level, multirobot level.

[15] Collins T. R., Arkin R. C., Henshaw M.: ”Integration of Reactive Navigation with a Flexible Parallel Hardware Architecture”, IEEE Int. Conf. on Robotics and Automation, vol. 1, Atlanta, Georgia, 1993, May. *Keywords:* abstraction level.

[16] Connel J. H.: ”A Behavior-Based Arm Controller”, IEEE Trans. on Robotics and Automation, vol. 5, no. 6, pp. 784-791, 1989, Dec.. *Keywords:* robot level, behavior level.

[17] Cordes S., Berns K., Dillmann R.: ”Steuerungsarchitektur der sechsbeinigen Laufmachine LAURON”, Autonome Mobile Systeme, 9. Fachgesp. an der TU München, G. Schmidt, 1993, Oct.. *Keywords:* kinematics level, abstraction level.

[18] Crowley J. L., Causse O.: ”Layers of Control in Autonomous Navigation”, Mechatronics & Robotics, vol. I, IOS Press, 1991. *Keywords:* abstraction level.

[19] Damm M.: ”Zweiarm-Koordination kraftgeregelter Manipulatoren mit situationsabhängiger Lastverteilung”, VDI Verlag, Reihe 8, no. 519, 1995. *Keywords:* robot level, abstraction level.

[20] Dario P., Ribechini F., Genovese V., Sandini G.: ”Instinctive behaviors and personalities in societies of cellular robots”, IEEE Int. Conf. on Robotics and Automation, pp. 1927, Sacramento, California, 1991. *Keywords:* Behavior level.

[21] Dilts D. M., Boy N. P., Whoirms H. H.: ”The Evolution of Control Architectures for Automated Manufacturing Systems”, Journal of Manufacturing Systems, vol. 10, 1991. *Keywords:* survey.

[22] Doty K. L., Van Aken R. E.: ”Swarm Robot Material Handling Paradigm for a Manufacturing Workcell”, IEEE. Int. Conf. on Robotics and Automation, vol. 1, pp. 778-782, Atlanta, Georgia, 1993, May. *Keywords:* multirobot level.

[23] Ejiri A., Watanabe I., Okabayashi K., Hashima M., Tatewaki M., Aoki T., Maruyama T.: ”Satellite Berthing Experiment with a Two-Armed Space Robot”, IEEE Int. Conf. on Robotics and Automation, pp. 3480-3487, 1994. *Keywords:* robot level.

[24] Everett H. R., Gilbreath G. A.: ”ROBART II: A Robtic Security Testbed”, Naval Ocean Sys. Center. Tech. Doc. 1450, San Diego, 1989, January. *Keywords:* abstraction level.

[25] Fatikow S.: ”Intelligent Robot Hand Control System Using a Tailorable Parallel Computer Concept”, IEEE Int. Conf. on Robots and Systems, München, Germany, 1994. *Keywords:* control level.

[26] Fayek R. E., Liscano R., Karam G. M.: ”A System Architecture for a Mobile Robot Based on Activities and a Blackboard Control Unit”, IEEE Int. Conf. on Robotics and Automation, vol. 2, pp. 267-274, 1993. *Keywords:* abstraction level, functions level.

[27] Fijany A., Bejczy A.: ”Parallel Computation Systems for Robotics: Algorithms and Architectures”, World Scientific, 1992. *Keywords:* algorithm level.

[28] Fleming P. L.: ”Parallel processing in control: The transputer and other architectures”, IEEE control engineering series. *Keywords:* algorithm level, (control level), survey.

[29] Fleury S., Herrb M., Chatila R.: ”Design of a Modular Architecture for Autonomous Robot”, Int. Conf. on Robotics and Automation, pp. 3508, 1994. *Keywords:* functions level, (multirobot level).

[30] Gaglianello R., Katseff H.: ”A Distributed Computing Environment for Robotics”, IEEE Int. Conf. on Robotics and Automation, pp. 1890-1896, 1986. *Keywords:* algorithm level.

[31] Gauthier D., Freedman P., Carayannis G., Malowany A.: ”Interprocess Communication for Distributed Robotics”, IEEE Journal of Robotics and Automation, pp. 493-504, 1987, Dec.. *Keywords:* multirobot level.

[32] Goto Y., Stenz A.: ”The CMU System for Mobile Robot Navigation”, IEEE Int. Conf. on Robotics and Automation, pp. 99-105, 1887, Apr.. *Keywords:* functions level, robot level.

[33] Graham James H.: ”Special Computer Architectures for Robotics: Tutorial and Survey”, IEEE Trans. on Robotics and Automation, vol. 5, no. 5, 1989. *Keywords:* kinematics level, algorithm level, control level, survey.

[34] Harmon S. Y.: ”Implementation of complex robot subsystems on distributed computing resources”, NATO ASI series, vol. F33. *Keywords:* functions level.

[35] Hartley R., Pipitone F.: ”Experiments with Subsumption Architecture”, IEEE Int. Conf. on Robotics and Automation, vol. 2, pp. 1652-1653, 1991. *Keywords:* behavior level.

[36] Henderson T. C, Hansen C., Bhanu B.: ”A Framework for distributed sensing and control”, in 9th IJCAI, Los Angeles, California, USA, 1985. *Keywords:* function level.

[37] Henrich D.: ”A review of parallel processing approaches to motion planning”, IEEE Int. Conf. on Robotics and Automation, vol. 4, pp. 3289-3294, 1996. *Keywords:* algorithm level.

[38] Hörmann A.: ”Steuerung und Systemarchitektur von fortgeschrittenen autonomen Systemen”, Robotersysteme 5, Springer-Verlag, pp. 173-185, 1989. *Keywords:* survey.

[39] Kappey D.: ”Autonome Mehrkamerasysteme für die flexible Fertigung”, Dissertation, Karlsruhe University, Institute for Real-Time Computer Systems and Robotics, Germany, 1995. *Keywords:* robot level.

[40] Karl J., Henrich D.: ”A review of parallel processing approaches to robot kinematics and Jacobian”, submitted to IEEE Int. Conf. on Robotics and Automation, Albuquerque, New Mexico, USA, 1997. *Keywords:* algorithm level.

[41] Kazanzides P.: ”A Multiprocessor System for Real-Time Robotic Control”, Information Sciences, vol. 44, 225- 247, 1988. *Keywords:* algorithm level, abstraction level.

[42] Kimoto K., Yuta S.: ”An Architecture and Real Time Operating System of Autonomous Mobile Robot for Distributed and Incremental Development”, IOS, Rembold U. et al (Eds), IOS PRess, pp. 198-205, 1995. *Keywords:* functions level.

[43] Kumar V., Grama A., Gupta A., Karypis G.: ”Introduction to parallel computing - design and analysis of algorithms”, The Benjamin/Cummings Publishing Company, Inc., 1994. *Keywords:* introduction.

[44] Levi P., Muscholl M., Bräunl T.: ”Cooperative mobile Robots Stuttgart: Architecture and Tasks”, IOS 310-317, 1995. *Keywords:* abstraction level, multirobot level.

[45] Liscano Ramori: ”Using a Blackboard to integrate Multiples Activities and Achieve Strategic Reasoning for Mobile-Robot Navigation”, IEEE Expert, 1995. *Keywords:* functions level.

[46] Lüth T. C., Längle T.: ”Task Description, Decomposition, and Allocation in a Distributed Autonomous Multi-Agent Robot System”, Intelligent Robots and Systems 94, vol. 3, pp. 1516-1523, 1994. *Keywords:* functions level.

[47] Lyons D. M., Arbib M. A.: ”A Formal Model of Computation for Sensory-Based Robotics”, IEEE Trans. Rob. Aut., vol. 5, no. 3, pp. 280-293, 1989, June. *Keywords:* behavior level.

[48] Magnussen B.: ”A Parallel Control Computer Structure for Complex High Speed Applications”, ASCC., First Asian Control Conf., Tokyo, 1994. *Keywords:* control level.

[49] Magnussen B. B.: ”Infrastruktur für Steuerungs- und Regelungssysteme von robotischen Miniatur- und Mikrogreifern”, VDI Verlag, Reihe 8, no. 567, 1996. *Keywords:* control level, robot level.

[50] Mataric M. J.: ”Minimizing Complexity in Controlling a Mobile Robot Population”, IEEE Int. Conf. on Robotics and Automation, pp. 830-83, Nice, France, 1992. *Keywords:* multirobot level.

[51] Meyestel A.: ”Multiresolutional Feedforward/Feedback Loops”, IEEE Int. Symo. intelligent Control, pp. 85-90, 1991, Aug.. *Keywords:* abstraction level.

[52] Mignot R., Rawden A., Bengoa A.: ”Potential Industry Applications”, First European Conf. on Holonic Manufacturing Systems, Hannover, Germany, 1994, 1.Dec.. *Keywords:* multirobot level.

[53] Mitsumoto N., Fukuda T., Arai F., Tadashi H., Idogaki T.: ”Self-organizing Multiple Robotic System (A Population Control through Biologically Inspired Immune Network Architecture)”, IEEE Int. Conf. on Robotics and Automation, pp. 1614-1619, Minneapolis, USA, 1996, Apr.. *Keywords:* multirobot level, (behavior level).

[54] Moravec H. P.: ”The Stanford Cart and the CMU Rover”, Proc. of the IEEE, vol. 71, no. 7, pp. 872-884, 1983, July. *Keywords:* abstraction level.

[55] Nassal U., Damm M., Lüth T.: ”Mobile Manipulation: A Mobile Platform Supporting a Manipulator System for an Autonomous Robot”, RI/SME Fifth World Conf. on Robotics Research, Cambridge, 1994, Sept.. *Keywords:* robot level.

[56] Noreils F. R., Chatila R. G: ”Control of Mobile Robot Actions”, IEEE Int. Conf on Robotics and Automation, pp. 701-707, 1989. *Keywords:* functions level.

[57] Paredis C. J. J., Brown H. B., Khosla P. K.: ”A Rapidly Deployable Manipulator System”, IEEE Int. Conf. on Robotics and Automation, vol. 2, pp. 1434-1439, 1996. *Keywords:* kinematics level, robot level.

[58] Shin K., Epstein M.: ”Intertask Communications in an Integrated Multirobot System”, Jour. of Robotics and Automation, 1987, April. *Keywords:* multirobot level.

[59] Sousa J. B., Pereira F. L.: ”A General control Architecture for multiple Vehicles”, IEEE. Int. Conf. on Robotics and Automation, pp. 692-697, 1996, April. *Keywords:* multirobot level, (robot level, functions level).

[60] Stilwell D. J., Bay J. S.: ”Toward the Development of a Material Transport System using Swarms of Ant-like Robots”, IEEE Int. Conf. on Robotics and Automation, vol. 1, pp. 766-771, Atlanta, Georigia, 1993, May. *Keywords:* multirobot level.

[61] Sugano S., Kato I.: ”WABOT: autonomous robot with dexterous Finger-Arm: Finger-Arm Coordination control in Keyboard Performance”, IEEE Int. Conf. on Robotics and Automation, pp. 90-97, 1987. *Keywords:* kinematics level, abstraction level.

[62] Tigli J. Y., Occello M., Thomas M. C.: ”A Reactiv Multi-Agents System As Mobile Robot Controller”, Intelligent Robots and Systems, pp. 2008-2014, Yokohama. Japan, 1993, July. *Keywords:* function level.

[63] Vidal T., Ghallab M., Alami R.: ”Incremental Mission Allocation to a large Team of Robots”, IEEE ICARO, pp. 1620-1625, 1996. *Keywords:* multirobot level.

[64] Walker I. D., Cavallaro J. R.: ”Parallel VLSI Architectures for Real-Time Kinematics of Redundant Robots”, IEEE Int. Conf. on Robotcs and Automation, vol. 1, pp. 870-877, 1993. *Keywords:* algorithm level.

[65] Watanabe M., Onoguchi K., Kweon I., Kuno Y.: ”Architecture of Behavior-based Mobile Robot in Dynamic”, Int. Conf. on Robotics and Automation, pp. 2711- 2718, 1992. *Keywords:* behavior level.

[66] Whitcomb L. L.: ”Robot Control in a Message Passing Environment: Theoretical Question and Preliminary Experiments”, Int. Conf. on Robotics and Automation, pp. 1198- 1203, 1990. *Keywords:* control level, kinematics level.

[67] Zelinsky A.: ”Unsing an Augmentable Ressource to Robustly and Purposefully Navigate a Robot”, IEEE Int. Conf. on Robotics and Automation, 1995. *Keywords:* behavior level.

[68] Zheng Z., Lecocq H., Jordant R.: ”Adaptive Manipulator Control: A parallel Implementation on a Network of Microprocessors”, IEEE Int. Conf. on Robotics and Automation, vol. 1, pp. 878-882, 1993, May. *Keywords:* kinematics level.

[69] Zoyama A. B.: ”Modelling and Simulation of Robot Manipulators: a parallel Processing approach”, World Scientific. *Keywords:* algorithm level.