DOI: 10.17587/mau.22.200-207

P. S. Sorokoumov, petr.sorokoumov@gmail.com,
NRC "Kurchatov Institute", Moscow, Russian Federation

Corresponding author: **Sorokoumov Petr S.,** Research Engineering, NRC "Kurchatov Institute", Moscow, Russian Federation, e-mail:petr.sorokoumov@gmail.com

Accepted on January 12, 2021

Formalization of Tasks for Robotic Manipulators: Review and Prospects

Abstract

This overview of the problems formulations for robotic manipulators at different abstraction levels can be used to find the causes of troubles with some types of control systems. For many variants of manipulators, for example, biomorphic ones, it is not yet possible to achieve the required quality and universality. Nevertheless these tasks are solvable, which is proved by the natural movement control systems of biological organisms. One of the reasons of the difficulties is the complexity of the formalization of motion control, which prevents the development of universal approaches. The existing formalizations were separated by functional level to facilitate analysis. The high-level problems (the division of complex motor tasks into stages) are successfully solved by general planners or logical inference procedures. The middle-level problems (the trajectory tracing according to an abstract motor task) are so far solved less efficiently. Some existing tools, as linguistic methods, can greatly facilitate solution, but require significant and very laborious formalization of conditions. Inverse problems of kinematics and dynamics, conjugation of trajectory sections and direct control of the manipulator motors with error handling are further stages of processing; the quality of known solutions is usually acceptable. Based on the data collected, it can be argued that the development of methods for solving medium-level problems, i.e. constructing the trajectory of the robot according to the description of the action, is the most important domain for the successful creation of new types of manipulator control systems.

Keywords*: robotic arm, inverse kinematics problem, motion definition language*

For citation:

Sorokoumov P. S. Formalization of Tasks for Robotic Manipulators: Review and Prospects, *Mekhatronika, Avtomatizatsiya, Upravlenie,* 2021, vol. 22, no. 4, pp. 200—207.

DOI: 10.17587/mau.22.200-207

УДК 004.896

DOI: 10.17587/mau.22.200-207

П. С. Сорокоумов, инженер-исследователь, petr.sorokoumov@gmail.com
НИЦ "Курчатовский институт", Москва, Россия

Формализация заданий для роботов-манипуляторов: обзор и перспективы развития

Выполнен обзор методов формализации задач управления роботами-манипуляторами, возникающих на разных уровнях представления. Для многих видов манипуляторов, например биоморфных, достичь нужного качества управления в сочетании с универсальностью пока не удается, хотя возникающие при этом задачи принципиально решаемы, что доказывается успешной работой систем контроля движения биологических организмов. Одной из причин затруднений является сложность формализации задач управления движением, поэтому целесообразно проанализировать наиболее распространенные подходы к постановке этих задач в целях поиска перспективных вариантов развития.

Анализ существующих методов решения проведен отдельно по каждому функциональному уровню. Наиболее высокоуровневая из рассматриваемых подзадач — разделение комплексных заданий на этапы — обычно успешно реализуется общими средствами декомпозиции сложных процессов, например методами планирования или логического вывода. Обработка подзадач среднего уровня представления — построение траектории по двигательному заданию — осуществляется пока менее эффективно. Показано, что имеющиеся средства, например лингвистические описания

движений, могут значительно облегчить работу с этим уровнем, но требуют значительной и трудоемкой формализации. Дальнейшие стадии обработки — решение обратных задач кинематики и динамики, сопряжение участков траекторий, а также непосредственное управление двигателями манипулятора с обработкой ошибок — решаются имеющимися средствами с приемлемым качеством.

Исходя из собранных данных можно утверждать, что развитие методов решения среднеуровневых задач, т.е. построение траектории движения манипулятора по описанию требуемого от него действия, является наиболее важным для успешного создания перспективных типов манипуляторов, способных двигаться точно и разнообразно.

Ключевые слова: *робот-манипулятор, обратная задача кинематики, язык определения движений*

Introduction

Robotic manipulators are now widely used, and they usually work not in isolation, but as part of complex systems. Ideally, the formalization methods of motor tasks should be well suited for both the robot and the human (developer). For that they should combine rigor and unambiguity with simplicity of human perception, versatility, and ease of making changes. On practice these requirements are often too hard. Therefore often too primitive or, conversely, overcomplicated approaches are used. It is necessary to analyze the methods used at different stages of the process of setting tasks for the robot manipulator to find better approaches. The results of this review can be useful both for choosing methods of practical implementation of control systems and for planning further research. Since the successful operation of the manipulator requires the solution of a huge number of diverse tasks, it is necessary to consider their existing classifications and decompositions before the analysis..

Hierarchical structure of the manipulator control task

The complex nature of the problem of manipulator control has long been known. So, in the classical work [1, p. 6—10] the following main stages are distinguished:

1. Solving the problems of kinematics and dynamics of the manipulator, i.e. a mathematical description of the geometry of its motion, forces and moments acting on it;
2. Tracing of trajectories and control of movement along them, that is, the use of knowledge about kinematics, robot dynamics and sensory information for smooth movement of the working tool in compliance with the restrictions;
3. Programming tasks for the manipulator as sequences of trajectories;
4. Setting tasks for the manipulator, in particular, the use of machine intelligence for this.

Other suggested hierarchies differ from the one given mostly only in fine details or naming of stages. For example, in [2] for a robot interacting with humans trajectory smoothing is considered as a separate processing stage in order to reduce the stress arising in people with sudden movements of the manipulator.

Decomposition of a manipulator control task into hierarchical levels is very important in practice. Often in new systems only some parts are created anew, while others are used ready-made [3]. The analysis of control systems, for example in security audits, is also often carried out separately at levels [4]. Below in this work some proposed methods for solving problems are described for each of the levels. Special attention paid to the using them as an interface tool for the developer of the system as a whole, allowing to choose the required behavior of each level. In practice, adjacent levels sometimes may merge; also if obvious errors are recognized, it is possible to repeat processing with new input data. At the end of the article, control systems that work on many stages are briefly described. Conclusions contain a summary of the review.

The gripping problem is not addressed in this work, because it is too complicated an a domain. In general an exact solution can not be found because of the difficulties in modeling of friction and deformations of a captured object [5] but there are approximate solutions for many practically important cases.

Statement of problems of kinematics and dynamics

Quite often kinematic modeling of the manipulator uses the Denavit-Hartenberg representation. It connects the Cartesian coordinates of the links with the generalized coordinates that determine the position of the robot's degrees of freedom. To take into account the dynamics the Lagrange equation [1] can be used. The transition from Cartesian coordinates to generalized ones, or solving the inverse kinematics (IK) problem, can be performed by various methods. An exact solution of the IK problem exists for all manipulators with a small number of links (two or three) and for many models with up to six links. For

more complex systems the solution is usually found numerically. There are many rather full reviews of methods for solving inverse problems of kinematics and dynamics both in general and special cases [6, 7] and the conclusions generally coincide: the available methods most often solve IK problem quite well, difficulties are encountered mainly when working with dynamic effects and when interacting with obstacles in the environment. There is a large number of ready-made software tools for solving IK problem, including free and open source solutions, for example MoveIt! [8] or RoBoy [9]. The inverse problem of dynamics in practice is often solved by them also by adjusting the control signals transmitted to the manipulator according to the sensor data. Because of the mentioned difficulties in obstacle processing it is sometimes preferable do not solve the IK problem directly. In particular, to move the manipulator in a close environment, it was proposed to use neural networks [10,11] trained on a sample collected from its sequential positions when moving along permitted trajectories. This method is capable to some extent of ensuring the continuity of the solution of inverse problems, since the sample includes only examples from admissible (i.e., continuous) trajectories, but it is not possible to strictly guarantee the quality of solutions because of the uncertainties inherent in neural network approximations. Therefore, to ensure safety, the trajectories generated by such systems must be later checked for constraint compliance and continuity.

Another option for using a neural network approach for low-level control of multi-legged limbs is the ENSO (Evolution of Network Symmetry and mOdularity) architecture [12]. It was originally proposed to increase the reliability of adaptive control of multi-legged platforms [13]. Each limb in ENSO is controlled by a separate neural network module connected with the rest (Fig. 1). Initially, the synaptic coefficients of all modules are the same but during the learning process they receive random small deviations which are considered as symmetry violations. There are many ways to train such a network; the authors of the original work used a genetic algorithm, but it is possible to apply arbitrary methods of reinforcement learning.

The value of the ENSO approach is that the created system is able to learn purposeful movement, as experimentally confirmed by the authors of the original work. Stable synchronizations of movements of different degrees of freedom in an initially symmetric system can be obtained. This property can be useful for controlling manipulators with excessive degrees of freedom — for example, simulating human limbs.

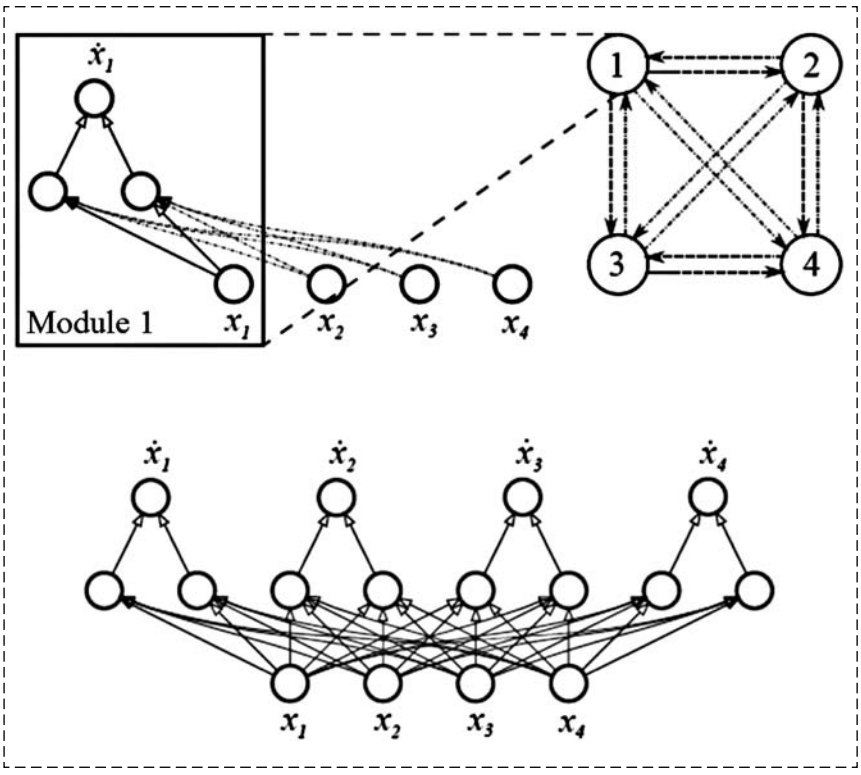


Fig. 1. An example of neural network architecture of ENSO modules for a four-legged robot [12]: one module, connections between modules and a complete network

It is obvious that the modern methods of this level are numerous and well developed, but for many applied problems direct interface to them by explicitly indicating the desired link position turns out to be too low-level and therefore undesirable. These methods are often used in interfaces indirectly.

Trajectory motion controls

The most common concept of constructing a trajectory from segments of a set of simplest lines (straight lines or arcs) connecting points located on the trajectory. In this case, it is required to match the segments with the provision of acceptable smoothness of the overall curve; it, firstly, must be smooth to a certain order and, secondly, not include high-frequency vibrations. Implementations of this concept differ mainly in the properties of the resulting mates. It

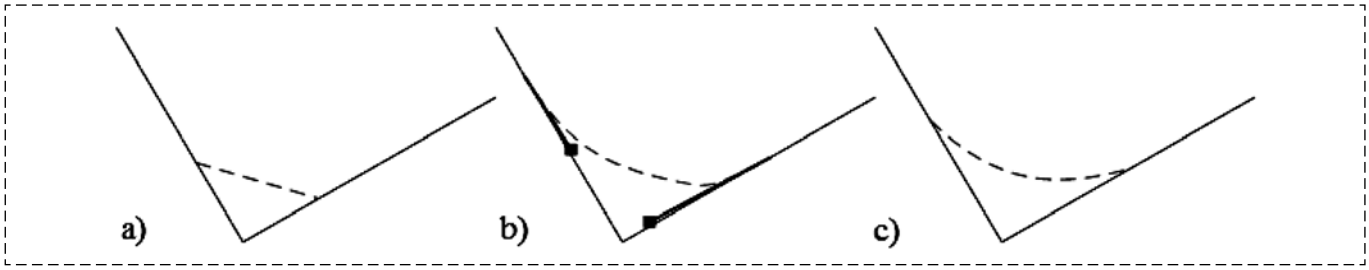


Fig. 2. Examples of geometric primitives used for conjugation of trajectory segments:
a — straight line; *b* — NURBs curve [16]; *c* — hyperbola [15]

was proposed to use various geometric primitives for conjugation, for example, quadratic optimization [14], conical sections [15], NURBs [16] (Fig. 2). The choice of a specific option is usually justified by solving some optimization problem or by the compliance with some restrictions on the smoothness of the trajectory.

One of the most common approaches to the conjugation problem is fuzzy logic [17,18]. It is especially often used when there are significant nonlinearities in the joint, for example, when there are flexible sections [19] or springs. At the same time, for a two-tier system the problem of fuzzy control is solved in a completely straightforward and accurate manner. There is even a simple hardware implementation [20]. This method does not work for systems with a large number of links.

There are also approaches that to a greater extent use the physical properties of the manipulator. For example, in [21] the energy consumption by motors leading to their heating minimized numerically.

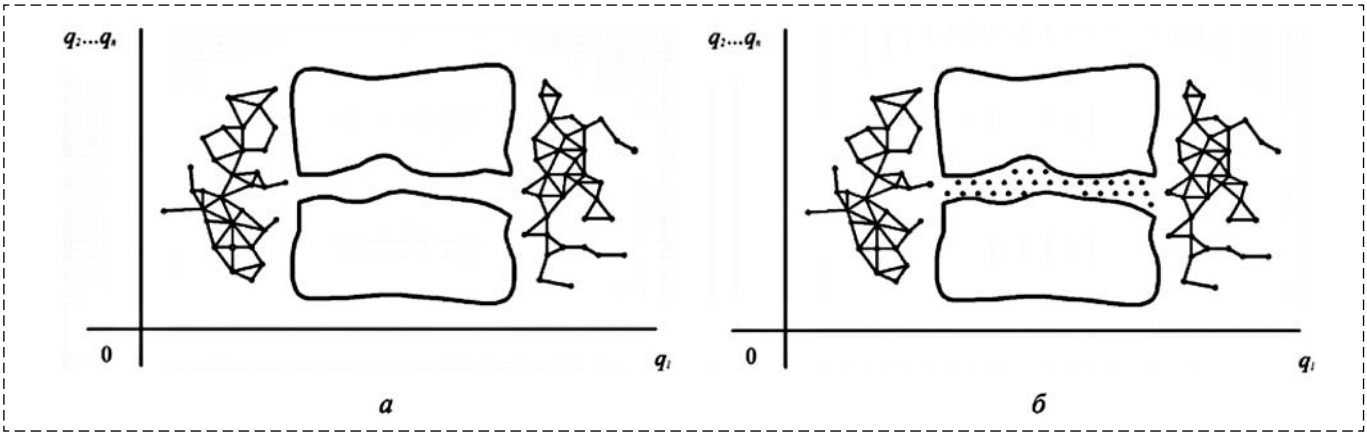
Most of the proposed tools do not take into account the specific requirements for the manipulator associated with the need to hold objects. Among the exceptions are the trajectory tracing for working with flexible objects [22] and the construction of trajectories for two jointly acting manipulators for grabbing one rigid object [23].

To control and correct movements, various methods can be used. Their applicability depends on the feedback signals available on this manipulator and on the requirements for the temporal and spatial characteristics of regulation. Information for correction can be obtained from various types of encoders or from external sensor devices, like a video camera [24].

Motion control based on data on the real and desired position of the manipulator can be formalized as an optimization problem. When describing the dynamics of a robot using the Lagrange equation, it is possible to use any common types of controllers, for example, a PID controller or a controller based on a computable torque. In this case, the PID controller can be used directly with a non-linear model of the system; the controller for the calculated torque requires preliminary linearization. In [25] it is proposed to take into account the uncertainties of the parameters of the robot to correct the solution obtained by the controller of the computable moment using additional optimization. This process formalized as a differential game with a quadratic functional of quality; simulation showed that the robustness of the resulting system with respect to the uncertainties of the parameters of the robot's motors increased. Adaptive control (the change in controller parameters over time) was also extensively used [26, 27] but since this approach has problems with stability, its application to this area remained limited. The review shows that this problem has been studied very well, and the considered tools complement well the means for solving IK and ID problems. It is highly desirable to be able to explicitly describe the conjugations in individual but practically important cases as for fragile objects or near obstacles.

Tools for constructing the trajectory of the manipulator

This stage is not always implemented in practice: if the number of different required trajectories is small (as in many industrial manipulators designed to perform a small number of operations), then the trajectory is often set simply by a list of points. This list is formed either analytically or by holding the manipulator along the desired path in manual or weakly automated mode. If these capabilities are not enough, then more complex tools are used, for example, planning. One of the classical approaches is to create a finite connected graph that covers the state space of the manipulator (Fig. 3). Graph nodes correspond to admissible states, and an edge between a pair of neighboring nodes is drawn only if the transition between the



а *б*

Fig. 3. Discretization of the working space of the robot manipulator by the combined method [28]:
a — random filling of large free areas with points with a partition into polyhedra; *b* — the addition of discretization with points of a narrow zone located at the nodes of a regular lattice

corresponding states is safe for the given robot, i.e. does not lead to a collision with obstacles, other links of this manipulator, or to the parameters exceeding the permissible limits. If the geometry of the feasible regions is fully known then methods of optimal space partitioning can be applied, for example, by constructing a Voronoi diagram. In practice, these conditions are often not met; therefore, non-optimal partitions by regular lattices or their hybrid with a stochastic approach can be used [28]. On the graph obtained by any of the indicated methods, the problem of finding the shortest path from the initial state to the desired one is solved; the final plan becomes the required trajectory of movement.

In addition to those described, there were also proposed tracing tools that are more often used in other branches of robotics as the presentation of targets as sinks, and obstacles as sources of potential fields [29]. Many ready-made implementations of scheduling methods are contained, for example, in the MoveIt! [8] package.

Another option for constructing a trajectory, which makes it possible to more fully take into account the capabilities of specific robots, is described by linguistic models of permissible movements. Thus, methods were proposed based on the language of defining movements [30—32] or (as their development) on the technology of modular movements [33]. The language for describing movements is a formal language according to Chomsky, described by context-free grammar. The symbols of this language are sets of actions on the controlled system at a given time. The language interpreter program plays the role of an intermediary between the discrete language description of the system's actions and continuous signals to executable devices. In other words, the control process is described as the execution of a sequence of control actions that are symbols of a certain "alphabet". At each moment of time only one symbol is executed. Complex movements are described with character strings. A formal language called motion definition language (MDL) can be constructed from the valid sets of strings. These publications and subsequent works of the authors describe numerous examples of applications of this approach both to manipulations and to other control problems.

The advantages of this model include the versatility of presentation and ease of implementation on a particular selected system. Individual parts of the trajectories in the state space are represented by separate small changes of several degrees of freedom at once. They are combined according to the laws fixed in the formal grammar, forming integral movements. In some cases, it is possible to reduce such systems to finite state machines [34]. Character sets can be quite small, making it easier to work with. The disadvantages of this approach include the need for manual assignment of grammar for a specific task, because it is very difficult to automatically generate adequate inference rules for characters described in such a form. Translating characters from representing one manipulator to another is also nontrivial.

In [35] another linguistic representation of the tasks of manipulating objects is proposed. It is a set of diverse actions, consisting of functions, each of which determines its trajectory from the initial to the final state. In this case, the domain of each action is not all possible states of the robot, but only those obeying some known a priori formal constraints. For

example, the following five primitives are used as a set of heterogeneous actions suitable for describing typical manipulation tasks for a mobile manipulator:

- transit, i.e. movement of the robot without load;
- rigid transfer, i.e. moving with the held object;
- pushing an object with a closed manipulator;
- approach / retreat of the manipulator in open view for gripping or releasing an object;
- picking up an object lying on the floor.

Further, if the trajectory is considered as a function that transfers the robot from the initial state to the final state, then the problem of path tracing can be posed, and each stage of the path will correspond to one of the actions. For a known robot and for each of the primitives of this method (DAMA — diverse action manipulation algorithm) the domain and the tracing method can be defined, i.e. they are in principle determined. However, in these calculations, in addition to the parameters of robots, it is inevitable to take into account the properties of manipulated objects. This concept is limited, for example, by the fact that a plate lying on the floor will be captured by the pickup action, and the one standing on the table — by approach action; perhaps the pickup action was introduced due to the difficulties in solving this particular problem by the more common action. Although, in general, this method is of great interest, its applicability is therefore limited.

The paper [36] also proposes a joint use of the DAMA method for motion decomposition and the ENSO architecture for training a control system. The set of primitives differs from that described in the original work [35], which again shows the limitations of the approach to dividing motion into components in DAMA. Nevertheless, the final simulated system was able to successfully learn how to solve manipulation problems, although exact information about stability is not given.

Some types of linguistic models use simpler abstractions to describe the motor tasks — the description of the trajectories of movement of manipulation objects in the form of alphabet characters [37]; the movement of the robot outside the contact with the controlled system is not described. This allows the developer to focus on the most important aspects of manipulation, providing solutions to auxiliary problems at the lower hierarchical levels. However, taking into account in such systems the desired orientation of the robots' grippers, as well as the synchronization of movements between several simultaneously operating manipulators, is very complicated, requiring the use of poorly developed formalisms, for example, parallel grammars.

In some particularly critical cases (for example, in the maintenance systems of nuclear power plants), approaches are used that leave the choice of trajectories for most free motions to the operator [38]. In this case, only individual fine manipulation operations are subject to automation, the commands for starting which are also given by the operator. In general, we can say that the means of this level are very developed and have significant potential, but of all the possible options, as a rule, only planning methods in the state space are widely used. The reason for this, apparently, can be considered the significant requirements for the formalization of the description of movements characteristic of linguistic models, which lead to complex models with difficult to measure, often unstable parameters.

Methods of manipulation decomposition

In general, the methods used to solve this stage of the problem do not differ from the tools for decomposing complex problems used in any section of artificial intelligence. Rule-based systems, expert systems, planning and other formalisms can be applied. A detailed review of training methods for robotic manipulators using the listed and other approaches is given in [39].

Manipulation problems are classic examples for demonstrating inference from its inception. Given an accurate and complete description of the situation, this approach is still the most visual and computationally simple. The initial state of the manipulator and the environment is represented in it by a set of true statements; permissible state changes — by the rules for forming new true statements from the existing ones. Recently, approaches have been actively developed that allow working with factors that are difficult to take into account in the classical formulation of the problem. For example, the dynamic nature of the environment can be taken into account using the means of fuzzy logic [40] for which the authors use the description of the problem in the specialized FLOPER language [41]. To take into account possible changes in the environment, facts are placed in the so-called lattice — a special algebraic not completely ordered structure that allows the simultaneous presence of mutually exclusive facts in the knowledge system.

The depth-first search from the initial state is another approach to the planning that builds the trajectory of movement explicitly. The set of actions permissible for the manipulator can be ini-

Summary of the survey

Method	Hierarchical level	Features	Examples
Classical IK and ID problems	1	Well developed and used frequently, but there may be problems with continuity of solutions in complex environments with many obstacles	[8]
ENSO and other neural network approaches	1	Able to learn complex movements, including synchronous, but poorly interpreted	[12]
Regulatory control of trajectory passing	1—2	Applicability strongly depends on the operating conditions of the system, requires careful testing	[45]
Planning on covering graphs	2	Well-known and frequently used method that poorly adapts to the particular manipulator	[28]
Motion Definition Language (MDL)	2—3	It takes into account the specifics of a particular manipulator well, but requires significant formalization of the task and is not universal	[30—32]
Diverse action manipulation algorithm (DAMA)	2—3	Similar to the previous	[35]
Logic systems, incl. rule-based	3	Decisions are easy to interpret	[41]
Copying movements	2	Universal approach, easy to prototype, but requires operator control	[43]
Playing back recorded movements	2	Universal approach, easy to prototype, but not adapted to changing conditions	[44]

tially specified as a finite list (for example, a set of DAMA primitives) to facilitate interpretation.

Simpler solutions are also used, for example, simulating the states of the robot using an explicitly defined finite automaton, while the transition between states is carried out after performing the movements with the receipt of confirmation of the successful manipulation from the operator [42]. In general, the tools used at this stage have no specific features in comparison with those used for similar tasks (for example, planning) in other subject areas.

Manipulator control without explicit hierarchical levels

Simpler and straightforward but widely used in practice methods fall into this category. These are various kinds of copying controllers for humanoid robots whose task is to repeat the reference movements of a person. They can be used for complete copying of human movements if the robot's limb is arranged similarly to a human. Reproducing the movements of only some parts of the hand, for example, the hand [43] is also possible.

There are also systems that reproduce recorded forced movements (movement players). Their applicability is limited due to the need to constantly monitor the operating conditions of the manipulator. When these conditions are violated its behavior becomes unpredictable. These methods are often applied if it is too difficult to formally describe the problem or in especially critical systems that require an operator for safety, for example, in the space [44] and nuclear [38] industries.

Conclusion

From the review, we can conclude that a very large arsenal of tools has been accumulated for all the main manipulation problems. The main findings of the review are summarized in Table.

An analysis of the survey results shows that the areas of application of different approaches and the problems they solve differ significantly. For complete automation of manipulation, common software packages have ready-made tools of all levels. From the development prospects the linguistic approaches (MDL and DAMA) are of great interest, since they provide an interpretable formal connection between high-level tasks (level 3) and direct control (level 1) by specifying a formal language for the operator. At the same time, higher-level and lower-level problems of automatic control are acceptable to existing off-the-shelf systems. This makes further research of linguistic models of manipulator control problems very promising.

References

1. **Fu K. S., Gonzalez R. C., Lee C. S. G.** Robotics: control, sensing, vision, and intelligence, New York, McGraw-Hill, 1987.

2. **Zhao R.** Trajectory planning and control for robot manipulations, Toulouse III, Université Paul Sabatier, 2015.

3. **Krechetov I. V., Poselsky I. A.** The manipulator tasks planning and robotic sorting facility management, *The Concepts of*

sustainable development of science in modern conditions. Articles of the International Scientific and Practical Conference, 2018, pp. 162—173.

4. **Kartashev V. A.** et al. Problems of manipulation robot control to ensure the movements safety, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2015, vol. 16, no. 1, pp. 24—28.

5. **Monkman G. J.** et al. Introduction to Prehension Technology. Robot Grippers, Wiley-VCH, 2006, pp. 1—17.

6. **Aristidou A.** et al. Inverse Kinematics Techniques in Computer Graphics: A Survey, *Comput. Graph. Forum*, John Wiley & Sons, Ltd, 2018, vol. 37, no. 6, pp. 35—58.

7. **Chan Y. P.** et al. A Survey on Inverse Dynamics Solvers for Cable-Driven Parallel Robots, *Australasian Conference on Robotics and Automation*, ACRA, 2017.

8. **Coleman D.** et al**.** Reducing the Barrier to Entry of Complex Robotic Software: a MoveIt! Case Study, arXiv e-prints, 2014, pp. arXiv:1404.3785.

9. **D'Souza A., Vijayakumar S., Schaal S.** Learning inverse kinematics, *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics,* Maui, HI, 2001, vol. 1, pp. 298—303.

10. **Voinov I. V.** et al. Control system for a robotic manipulator using neural network algorithms for limiting the working area of the gripper, *Bulletin of the South Ural State University. Series: Computer technologies, control, electronics*, 2017, vol. 17, no. 4, pp. 29—36.

11. **Smirnov P. A., Yakovlev R. N.** Solution of direct and inverse problems of kinematics in the positioning system of manipulator links, *Mekhatronika, Avtomatizatsiya, Upravlenie,* 2019, vol. 12, pp. 732—739.

12. **Valsalam V. K.** et al. Constructing Controllers for Physical Multilegged Robots using the ENSO Neuroevolution Approach, *Evol. Intell.*, 2012, vol. 5, no. 1, pp. 1—12.

13. **Valsalam V., Miikkulainen R.** Evolving symmetric and modular neural networks for distributed control, *Proceedings of the 11th Annual Genetic and Evolutionary Computation Conference, GECCO-200,* 2009, pp. 731—738.

14. **Rossi R.** et al. Trajectory Generation for Unmanned Aerial Manipulators Through Quadratic Programming, *IEEE Robot. Autom. Lett.,* 2017, vol. 2, no. 2, pp. 389—396.

15. **Dijk N. J. M. Van.** Generic trajectory generation for industrial manipulators, Eindhoven University of Technology, 2006.

16. **Steuben J., Steele J., Turner C.** NURBs for Robot Manipulator Trajectory Generation, *Proceedings of the ASME Design Engineering Technical Conference,* 2011, vol. 6.

17. **Yi S. Y., Chung M. J.** A robust fuzzy logic controller for robot manipulators with uncertainties, *IEEE Trans. Syst. Man, Cybern.* Part B, 1997, vol. 27, no. 4, pp. 706—713.

18. **López M., Arteaga-Pérez M.** Fuzzy Logic Control of a Robot Manipulator in 3D Based on Visual Servoing, *IFAC Proc.,* 2011, vol. 18.

19. **Rostami Kandroodi M.** et al. Control of Flexible Joint Manipulator via Reduced Rule-Based Fuzzy Control with Experimental Validation, *Int. Sch. Res. Netw. ISRN Artif. Intell. Artic. ID,* 2012, vol. 309687.

20. **Banerjee S., Woo P. Y.** Fuzzy logic control of robot manipulator, *Proceedings of IEEE International Conference on Control and Applications,* 1993, pp. 87—88 vol. 1.

21. **Guilbert M., Wieber P., Joly L.** Optimal Trajectory Generation for Manipulator Robots under Thermal Constraints, *IEEE-RSJ International Conference on Intelligent Robots & Systems,* Beijing, 2006.

22. **Nakamura Y., Igarashi H.** Manipulator trajectory generation for flexible object handling, *2008 10th IEEE International Workshop on Advanced Motion Control,* 2008, pp. 143—148.

23. **Pajak I.** Real-Time Trajectory Generation Methods for Cooperating Mobile Manipulators Subject to State and Control Constraints, *J. Intell. Robot. Syst. Theory Appl. Journal of Intelligent & Robotic Systems,* 2019, vol. 93, no. 3—4, pp. 649—668.

24. **López M., Pérez M., Leite A.** Modelado de Sistemas de Visión en 2D y 3D, *un Enfoque Hacia el Control de Robots Manipulador*es, 2013, vol. 17, pp. 12—21.

25. **Gurko A. G., Yanchevsky I. V.** Guaranteed motion control of a manipulative robot // Radioelectronics, Informatics, Control. 2014. № 2. P. 174—181.

26. **Slotine J.-J. E., Li Weiping.** Adaptive manipulator control: A case study, *IEEE Trans. Automat. Contr.,* 1988, vol. 33, no. 11, pp. 995—1003.

27. **Slotine J.-J. E., Li W.** Applied Nonlinear Control. Prentice-Hall, 1991.

28. **Kozhevnikov M. M.** et al. Synthesis of trajectories for assembly and welding robots-manipulators in a working environment with obstacles, *Robotics and mechatronics,* 2015, vol. 3 (13), pp. 36—42.

29. **Tsuji T., Morasso P. G., Kaneko M.** Trajectory generation for manipulators based on artificial potential field approach with adjustable temporal behavior, *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS '96*, 1996, vol. 2, pp. 438—443 vol. 2.

30. **Brockett R. W.** Hybrid Models for Motion Control Systems BT — Essays on Control: Perspectives in the Theory and its Applications / ed. Trentelman H. L., Willems J. C. Boston, MA, Birkhäuser Boston, 1993, pp. 29—53.

31. **Brockett R.** Language driven hybrid systems, *Proceedings of 1994 33rd IEEE Conference on Decision and Control,* 1994, vol. 4, pp. 4210—4214.

32. **Liu Z.** et al. Motion Description Language for Trajectory Generation of a Robot Manipulator, 2017.

33. **Liu Z.** et al. A New Type of Industrial Robot Trajectory Generation Component Based on Motion Modularity Technology, *J. Robot*, 2020, vol. 2020.

34. **Gennaro S. D. I.** et al. Reduction of timed hybrid systems, *Discret. Event Dyn. Syst. Theory Appl.,* 1998, vol. 8, no. 4, pp. 343—351.

35. **Barry J. L.** Manipulation with Diverse Actions, Massachusetts Institute of Technology, 2013, 201 p.

36. **Engel E. A.** Method of intelligent computing for configuration management of a mobile robot, *Bulletin of NEFU*, 2015, vol. 3 (47), pp. 127—137.

37. **Sorokoumov P. S.** The fuzzy language models for synchronizing the multi-limbic manipulator movements*, IX International Scientific and Practical Conference "Integrated models and soft computing in artificial intelligence",* Kolomna, 2019.

38. **Kuvshinnikov V. S., Kovshov E. E.** The use of a neural map for auxiliary control of a portal manipulator, *Cloud Sci.*, 2018, vol. 5, no. 2, pp. 310—324.

39. **Kroemer O., Niekum S., Konidaris G.** A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms, *ArXiv,* 2019, vol. abs/1907.0.

40. **Morcillo P.** et al. Declarative Traces into Fuzzy Computed Answers, 2011, 170—185 p.

41. **Medina J., Ojeda-Aciego M., Vojtáš P.** Multi-Adjoint Logic Programming with Continuous Semantics, *Lecture Notes in Computer Science*, 2001, vol. 2173, pp. 351—364.

42. **Shuai I., Yushchenko A. S.** Dialogue control system of a robot based on the theory of finite automata, *Mekhatronika, Avtomatizatsiya, Upravlenie,* 2019, vol. 20, no. 11. pp. 686—695.

43. **Kofman J.** et al. Teleoperation of a robot manipulator from 3D human hand-arm motion, *Optomechatronic Syst.,* IV, 2003, vol. 5264, pp. 257.

44. **da Fonseca I., Pontuschka M., Lima G.** Kinematics for Spacecraft-Type Robotic Manipulators. 2019.

45. **Siciliano B., Khatib O.** ed. Springer Handbook of Robotics, Berlin, Heidelberg, Springer Berlin Heidelberg, 2008.