Article

Parallel Computational Algorithm for Object-Oriented Modeling of Manipulation Robots

Oleg Krakhmalev ¹, Sergey Korchagin ¹ , Ekaterina Pleshakova ^{1,*}, Petr Nikitin ¹ , Oksana Tsibizova ², Irina Sycheva ³, Kang Liang ⁴, Denis Serdechnyy ⁵, Sergey Gataullin ¹ and Nikita Krakhmalev ⁶

¹ Department of Data Analysis and Machine Learning, Financial University under the Government of the Russian Federation, 4-th Veshnyakovsky Passage, 4, 109456 Moscow, Russia; olegkr64@mail.ru (O.K.); SAKorchagin@fa.ru (S.K.); pvnikitin@fa.ru (P.N.); STGataullin@fa.ru (S.G.)

² Department of Russian as a Foreign Language and General Theoretical Subjects, Russian State Agrarian University—Moscow Timiryazev Agricultural Academy, 127550 Moscow, Russia; cibizova75@mail.ru

³ Department of the Chair of Special Animal Husbandry, Russian State Agrarian University—Moscow Timiryazev Agricultural Academy, 127550 Moscow, Russia; in_sychewa@mail.ru

⁴ Engineering Training Center, Shanghai Polytechnic University, 2360 Jin Hai Road, Pudong District, Shanghai 201209, China; kangliang@sspu.edu.cn

⁵ Department of Innovation Management, State University of Management, Ryazansky Pr., 99, 109542 Moscow, Russia; DV_Serdechnyy@guu.ru

⁶ Department of Engineering Graphics, Moscow State University of Technology STANKIN, Vadkovsky Lane, 3a, 127055 Moscow, Russia; krakhmalevnik@yandex.ru

* Correspondence: pleshakova_es@mail.ru

Citation: Krakhmalev, O.; Korchagin, S.; Pleshakova, E.; Nikitin, P.; Tsibizova, O.; Sycheva, I.; Liang, K.; Serdechnyy, D.; Gataullin, S.; Krakhmalev, N. Parallel Computational Algorithm for Object-Oriented Modeling of Manipulation Robots. *Mathematics* **2021**, *9*, 2886. <https://doi.org/10.3390/math9222886>

Academic Editors: Simeon Reich and Fabio Caraffini

Received: 13 September 2021

Accepted: 9 November 2021

Published: 12 November 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Abstract: An algorithm for parallel calculations in a dynamic model of manipulation robots obtained by the Lagrange–Euler method is developed. Independent components were identified in the structure of the dynamic model by its decomposition. Using the technology of object-oriented programming, classes corresponding to the structures of the selected components of the dynamic model were described. The algorithmization of parallel computing is based on the independence of the calculation of objects of individual classes and the sequence of matrix operations. The estimation of the execution time of parallel algorithms, the resulting acceleration, and the efficiency of using processors is given.

Keywords: manipulation robots; dynamic model; mathematical modeling; object-oriented approach; computational algorithms; parallel computing

1. Introduction

In this paper, the object of research is manipulation systems considered as components of anthropomorphic robots. Robot manipulation systems are open kinematic chains formed by sequentially connected links with each other. The links are connected by joints, which, as a rule, have one degree of freedom (Figure 1) [1].

Since robots are mechanical devices, well-known methods of theoretical mechanics are used for their modeling. The choice of the method significantly determines the structure of the computational algorithm. These algorithms describe a sequence of calculations of the coefficients of the equations of motion that make up the dynamic model of the robot. The ways of solving these equations are determined by the problem statement [2–6].

If the trajectories of the robot's movement are set according to the condition of the problem, then the necessary forces in the drives can be determined based on the laws of change of the generalized (articulated) coordinates of the robot, considered as a function of time. To do this, it is necessary to solve a system of nonlinear equations. If the forces are given, then by integrating a dynamic model containing ordinary differential equations, generalized coordinates can be determined in the form of time functions [3–8].

Figure 1. 6-DOF robot manipulation system (a) and its kinematic scheme (b).

The development of algorithms for modeling robot manipulation systems that allow the performing of parallel calculations of individual parts of their mathematical models is an urgent task since it allows the solving of control problems of multi-link robots in real-time. This has become especially important with the advent of multiprocessor computing systems. There are known examples of the development of such algorithms conducted earlier [9–15]. However, the study of this issue cannot be considered complete since most of the presented results are not given by universal methods but by particular solutions.

The method considered in the article is based on an object-oriented approach to modeling the dynamics of robot manipulation systems. The developed method makes it possible to increase the efficiency of work related to the compilation of parallel computational algorithms. This is achieved based on the representation of mathematical models in the form of object schemes, in which each object is considered as part of a mathematical model, independent of other parts—objects associated with it in a general scheme [16,17].

To do this, in the structure of data and algorithms used in modeling, basic classes are allocated, on the basis of which other more complex classes are created, from which mathematical models are already directly formed. The original base class adopted a class describing the structure and methods of working with matrices. The concepts of “class” and “object” introduced by us fully correspond to the concepts of the same name used in object-oriented programming, implemented in many high-level programming languages.

Thus, mathematical models can be assembled from objects of base classes as from bricks. The assembly process is similar to the assembly of products from the elements of the LEGO constructor, the complex parts of which consist of a combination of basic parts. To implement the assembly process, a formalism has been developed that defines the rules for the interaction of constructor elements connected to each other, which are objects of the corresponding classes.

If the object is a matrix structure, then the multiplication operation of two objects will be represented by attaching the image of one object to the image of another object on one side (Figure 2). The matrix represented by the first object (Object 1) is multiplied on the right by the matrix represented by the second object (Object 2).

Figure 2. Multiplication of objects.

The operation of adding two or more matrix-type objects will be depicted as shown in Figure 3.

Figure 3. Adding objects: $A + B$ (a), $\sum A_i$ (b).

The subtraction of one object from another object will be depicted (Figure 4).

Figure 4. Subtracting objects.

The operation of assigning the value of one object to another object will be depicted, as shown in Figure 5. In this case, the object Object_1 will be assigned the values of the object Object_2.

Figure 5. Assignment operation.

Multiplication of a matrix-type object by a scalar (Figure 6).

Figure 6. Multiplying an object by a scalar.

The designations of other operations will be entered as needed.

2. Dynamic Model

The dynamic model of robot manipulation systems obtained based on the Lagrange–Euler method has the form [2,3]

$$\mathbf{M}_s \ddot{\mathbf{q}} + \dot{\mathbf{q}}^T \mathbf{C}_s \dot{\mathbf{q}} = Q_s \quad (1)$$

Here $\mathbf{q} = [q_1, \dots, q_n]^T$, $\dot{\mathbf{q}} = [\dot{q}_1, \dots, \dot{q}_n]^T$, $\ddot{\mathbf{q}} = [\ddot{q}_1, \dots, \ddot{q}_n]^T$ —vectors of generalized coordinates and their time derivatives; Q_s —generalized force by s the generalized coordinate (scalar); n —the number of degrees of freedom of the robot's manipulation system. $\mathbf{M}_s$ —matrix-string $(1 \times n)$ inertial characteristics of the manipulating robot

$$\mathbf{M}_s(\mathbf{q}) = \sum_{k=1}^n [m_j^{sk}], \quad m_j^{sk} = \text{tr} \left[\frac{\partial \mathbf{A}_{0k}}{\partial q_s} \mathbf{H}_k \frac{\partial \mathbf{A}_{0k}^T}{\partial q_j} \right], \quad j = (1, \dots, n), \quad (2)$$

C_s —matrix ($n \times n$) centrifugal and Coriolis effects

$$C_s(\mathbf{q}) = \sum_{k=1}^n [c_{ij}^{sk}], c_{ij}^{sk} = \text{tr} \left[\frac{\partial \mathbf{A}_{0k}}{\partial q_s} \mathbf{H}_k \frac{\partial^2 \mathbf{A}_{0k}^T}{\partial q_j \partial q_j} \right], j = (1, \dots, n), \quad (3)$$

Here, tr —the operation of calculating the sum of the diagonal elements of the matrix. $\mathbf{H}_k$ —matrix (4×4) inertia of the k -th link of the manipulation system, considered as a solid. $\mathbf{A}_{0k}$ —matrix (4×4) transformations of homogeneous coordinates from a coordinate system, connected to the k -th link, to a fixed coordinate system connected to the base of the robot.

$$Q_{Ds} = Q_s - Q_{Gs} - Q_{Fs} \quad (4)$$

where Q_{Ds} —the force developed by the drive in the s -th joint (hinge); Q_{Gs} —the generalized force corresponding to the s -th generalized coordinate, corresponding to the action of gravity; Q_{Fs} —the generalized force corresponding to the s -th generalized coordinate, corresponding to the action of external forces.

$$Q_{Gs} = \sum_{k=s}^n \text{tr} \left[\mathbf{F}_{Gk} \frac{\partial \mathbf{A}_{0k}^T}{\partial q_s} \right] \quad (5)$$

where $\mathbf{F}_{Gk}$ —matrix (4×4) containing the gravity of the k -th link.

$$Q_{Fs} = \sum_{k=s}^n \text{tr} \left[\mathbf{F}_{Rk} \frac{\partial \mathbf{A}_{0k}^T}{\partial q_s} + \mu_s \mathbf{F}_{Mk} \mathbf{A}_{0s}^T \right] \quad (6)$$

where $\mathbf{F}_{Rk}$ and $\mathbf{F}_{Mk}$ —matrix (4×4) containing projections of the main vector and the main moment of external forces, μ_s —ratio, $\mu_s = 1$, respectively, if the s -th joint is rotational and $\mu_s = 0$ if it is translational.

The methodology of composing dynamic models of robot manipulation systems based on their geometric and inertial models is considered in detail in the textbook [18].

The object scheme of this dynamic model can be represented as (Figure 7).

Figure 7. Object diagram of the dynamic model of manipulation systems of robots

On the presented object diagram, Q_s —the scalar value corresponding to the resulting generalized force along with the s -th generalized (hinge) coordinate; Q_{Ds} —the scalar value corresponding to the force developed by the drive in the s -th joint (hinge); Q_{Gs} —the scalar value corresponding to the generalized force Q_{Gs} ; Q_{Fs} —the scalar value corresponding to the generalized force Q_{Fs} . M_s —the object corresponding to the matrix-string $\mathbf{M}_s$; C_s —the object corresponding to the matrix $\mathbf{C}_s$; Dq , DDq and Dq^T —objects that model vectors $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$ and $\dot{\mathbf{q}}^T$, accordingly.

Classes that define the structure of the corresponding objects can be created using object-oriented programming languages, for example, C++, and must describe their corresponding objects following Expressions (1)–(6) [19–24].

3. Parallel Computing Algorithm

In a computational algorithm that implements modeling based on an object scheme (Figure 7), it is possible to distinguish several levels (tiers) at which the components of this object scheme are calculated.

At the first (I) level, objects corresponding to the inertia matrices of the links are calculated, $\mathbf{H}_k$, components of the so-called inertial model of manipulation systems (Figure 8).

Figure 8. The inertial model. I-st level of the algorithm.

At the second (II) level, objects corresponding to the transformation matrices of homogeneous coordinates are calculated, $\mathbf{A}_{0k}$, and their partial derivatives are related to the geometric model of manipulation systems (Figure 9).

Figure 9. Geometric model. II-nd level of the algorithm.

The objects calculated at the second (II) level depend on the parameters that change during the modeling process; in addition, when calculating partial differential matrices, it is necessary to exclude the calculation of elements of these matrices that are identically equal to zero. In accordance with this, the number of N_1 calculated partial differential matrices $\partial \mathbf{A}_{0k} / \partial q_s$ ($k, s = 1, \dots, n$) will be defined by the expression

$$N_1 = \frac{n}{2}(n+1) \quad (7)$$

and the number of N_2 matrices of the second derivatives ($\partial^2 \mathbf{A}_{0k} / \partial q_s^2$ and $\partial^2 \mathbf{A}_{0k} / (\partial q_i \partial q_j)$) by the expression

$$N_2 = \sum_{k=1}^n \frac{k}{2}(k+1) \quad (8)$$

At the third (III) level, calculations are performed based on object schemes corresponding to various elements of the matrices $\mathbf{M}_s$ and $\mathbf{C}_s$ ((M_s) and (C_s)), performed in accordance with Expressions (2) and (3), and scalar values Q_{Gs} and Q_{Fs} , performed in accordance with (5) and (6) (Figure 10).

Figure 10. Level III of the algorithm.

At the fourth (IV) level, based on objects Ms, Cs, QGs, and QFs, corresponding to the matrices $\mathbf{M}_s$ and $\mathbf{C}_s$ (Figure 11), and generalized forces Q_{Gs} and Q_{Fs} (Figure 12), the object schema is being calculated (Figure 7) by the corresponding dynamic model (1).

Figure 11. Object representation of matrices $\mathbf{M}_s$ and $\mathbf{C}_s$.

Figure 12. Object representation of generalized forces.

The hierarchical structure of the dynamic model parallel computing algorithm (1) is divided into several levels and, corresponding to the object scheme (Figure 7), without modeling the external load, it can be illustrated on a general computational scheme (Figure 13).

Figure 13. Levels of the dynamic model parallel computing algorithm.

The inertia model calculated at the first level of the algorithm reflects the distribution of masses within the links of the manipulation system under study, the corresponding mathematical model is a set of inertia matrices of the links H_k , ($k = 1, \dots, n$).

At the second level, a geometric model of the manipulation system is formed. Its mathematical model is a set of transformation matrices of homogeneous coordinates A_{0k} , ($k = 1, \dots, n$) supplemented by a set of partial derivatives of these matrices.

The definition of these two submodels in the structure of the dynamic model of robot simulation systems is of great methodological importance since it allows us to identify parts in the data structure and algorithms of the dynamic model that have a certain applied meaning, the calculation of which can be performed independently of each other.

Further parallelization of calculations performed at the appropriate levels of the considered algorithm for parallel calculations of the dynamic model of robot manipulation systems can be performed based on algorithms for parallel calculations of matrix operations.

4. Matrix Computation Algorithms

Consider the graph of the computational process of the product of a matrix (4×4) by a vector (4×1) (Figure 14), based on the ratio

$$w_i = \sum_{j=1}^4 a_{ij}v_j, \quad i = \{1, \dots, 4\}. \quad (9)$$

Figure 14. Graph of the computational process of a matrix (4×4) multiplication by a vector (4×1).

When calculating the product of a matrix of dimension $n \times n$ by the corresponding vector, the number of necessary scalar addition and multiplication operations can be estimated by

$$T_1 = 2n^2 \quad (10)$$

Analysis of the computational process graph (Figure 8) shows that parallel methods for solving the problem of the product of a matrix by a vector can be obtained by using parallel summation algorithms, in this case, a cascade summation algorithm is used. In addition, the way parallel computing is organized will depend on the number of processors available for use.

It should be noted that the sequences of operations performed for multiplying individual n rows of the matrix by a vector are independent and can be performed in parallel. The operations of multiplying each row by a vector include n independent operations of element-by-element multiplication, and can also be performed in parallel. Thus, the maximum required number of processors is determined by the value

$$p = n^2 \quad (11)$$

The execution time of the parallel algorithm, related to the execution time of a single scalar operation, when using p processors is determined by the execution time of the multiplication operation and the execution time of the cascade summation algorithm

$$T_p = 1 + \log_2 n \quad (12)$$

Performance indicators of the parallel algorithm, respectively,

$$S_p = \frac{T_1}{T_p} = \frac{2n^2}{1 + \log_2 n} \quad (13)$$

where S_p —acceleration for p processors, obtained by using a parallel algorithm

$$E_p = \frac{S_p}{p} = \frac{2}{1 + \log_2 n} \quad (14)$$

where E_p —processor utilization efficiency.

The information graph (Figure 14) for the corresponding computational process of multiplying the i -th of a matrix row (4×4) by a vector (4×1) built based on four processors is shown in Figure 15.

Figure 15. Information graph of the computational process of the i -th row multiplication of the matrix (4×4) by the vector (4×1)

The sequence of calculating the elements of the w_i vector of the result will be as follows: $p_{i1} = a_{i1} \times v_1$, $p_{i2} = a_{i2} \times v_2$, $p_{i3} = a_{i3} \times v_3$, $p_{i4} = a_{i4} \times v_4$; $p_{i1} = p_{i1} + p_{i2}$, $p_{i3} = p_{i3} + p_{i4}$; $p_{i1} = p_{i1} + p_{i3}$; $w_i = p_{i1}$. In the presented algorithm, the equality symbol is used to denote the assignment operation of the value of the calculated expression located on the right side of the variable located on the left.

The problem of multiplying rectangular matrices is determined by the relation

$$c_{ij} = \sum_{l=1}^k a_{il}b_{lj}, \quad i = \{1, \dots, n\}, \quad j = \{1, \dots, m\}. \quad (15)$$

In the case of the multiplication of square matrices of dimension $n \times n$, the number of scalar multiplication and addition operations is proportional to n^3 , and the value can be used to evaluate them

$$T_1 = 2n^3 \quad (16)$$

The sequences of operations performed for multiplying individual rows of the left matrix by the corresponding column of the right matrix are independent and can be performed in parallel. The algorithm of parallel calculation of matrix multiplication can be built based on the considered computational algorithm of matrix multiplication by a vector. Accordingly, the required number of processors for maximum parallelization of the computational algorithm will be determined by the value

$$p = n^3 \quad (17)$$

The information graph corresponding to the computational process of matrix multiplication (4×4), built based on 64 processors, is shown in Figure 16.

Figure 16. Information graph of the computational process of matrix (4×4) multiplication based on 64 processors.

The sequence of calculating the elements of the c_{ij} matrix of the result of multiplying two matrices corresponds to the previously presented algorithm for multiplying a matrix by a vector. The process is implemented in parallel in 16 groups of 4 processors each.

The execution time of the parallel algorithm, as before, will be determined by Expression (12). The resulting acceleration for p processors will increase

$$S_p = \frac{T_1}{T_p} = \frac{2n^3}{1 + \log_2 n} \quad (18)$$

and the efficiency of using processors under (14) will not change.

Reducing the number of processors used will require rebuilding the information graph (Figure 15), reflecting the computational process of matrix multiplication. This will result in an increase in the calculation execution time.

5. Conclusions

The algorithm of parallel calculations of the dynamic model (1) considered in the article, built based on an object-oriented approach, has four levels of parallelization (Figure 13) and can be supplemented with an initial (zero) level corresponding to data initialization.

The number of independently calculated operations at level III is maximum. At this level, when the number of processors used decreases, the greatest slowdown will be observed.

At level IV, the information graph of the computational process of multiplying a matrix ($n \times n$) by a vector ($n \times 1$) for the dynamic model (1) should be built based on n processors, by analogy with the information graph built based on four processors (Figure 15).

The considered algorithm for parallel calculations of the dynamic model (1) of manipulation robots based on an object-oriented approach can be implemented using object schemes (Figures 7–13). Such schemes can be created from objects corresponding to the geometric and inertial models, the manipulation system under study, as well as from objects corresponding to the external load model.

The use of parallel computational algorithms in modeling tasks requires computing systems to have a sufficient number of processors. This requirement can be implemented based on multi-chip reconfigurable computing systems created based on the use of programmable logic integrated circuits. Such computing systems can adapt to the structure of the problem being solved and provide the implementation of complex multi-level parallel algorithms [25–31].

Author Contributions: Data curation, I.S. and S.G.; Formal analysis, P.N.; Investigation, K.L.; Conceptualization and Methodology, O.K.; Writing—original draft, O.K., S.K., E.P., D.S., O.T. and N.K.; Software, N.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Siciliano, B.; Sciavicco, L.; Villani, L.; Oriolo, G. *Robotics*; Springer: Berlin/Heidelberg, Germany, 2009. [\[CrossRef\]](#)
2. Vukobratović, M.; Stokić, D. *Control of Manipulation Robots*; Springer: Berlin/Heidelberg, Germany, 1982. [\[CrossRef\]](#)
3. Vukobratović, M.; Stokić, D.; Kirčanski, N. *Non-Adaptive and Adaptive Control of Manipulation Robots*; Springer Science and Business Media LLC: Berlin/Heidelberg, Germany, 1985.
4. Chernousko, F.L.; Bolotnik, N.N.; Gradetsky, V.G. *Manipulation Robots. Dynamics, Control, and Optimization*; CRC Press: Boca Raton, FL, USA, 1993.
5. Wu, J.; Wang, J.; You, Z. An overview of dynamic parameter identification of robots. *Robot. Comput. Integr. Manuf.* **2010**, *26*, 414–419. [\[CrossRef\]](#)
6. Kuznetsova, A.; Maleva, T.; Soloviev, V. Detecting Apples in Orchards Using YOLOv3 and YOLOv5 in General and Close-Up Images. In *International Symposium on Neural Networks*; Springer: Cham, Switzerland, 2020; pp. 233–243.
7. Soloviev, V.; Titov, N.; Smirnova, E. Coking Coal Railway Transportation Forecasting Using Ensembles of ElasticNet, LightGBM, and Facebook Prophet. In *International Conference on Machine Learning, Optimization, and Data Science*; Springer: Cham, Switzerland, 2020; pp. 181–190.
8. Shirokanov, A.S.; Andriyanov, N.A.; Ilyasova, N.Y. Development of vector algorithm using CUDA technology for three-dimensional retinal laser coagulation process modelling. *Comp. Opt.* **2021**, *45*, 427–437. [\[CrossRef\]](#)
9. Fisette, P.; Péterkenne, J. Contribution to parallel and vector computation in multibody dynamics. *Parallel Comput.* **1998**, *24*, 717–728. [\[CrossRef\]](#)
10. Zomaya, A.Y. Parallel processing for robot dynamics computations. *Parallel Comput.* **1995**, *21*, 649–668. [\[CrossRef\]](#)
11. Belousov, I.R. The use of the method of symbolic transformations for the formation of parallel computing algorithms in the problems of kinematics and dynamics of robots. *Rep. IPM M.V. Keldysh RAS* **1993**, *5*, 19–93. (In Russian)
12. Malashonok, G.I.; Avetisyan, A.I.; Valeev, Y.D.; Zuev, M.S. Parallel Algorithms of Computer Algebra. *Proc. ISP RAS* **2004**, *8*, 169–180. (In Russian)
13. Vnukov, A.A.; Lisenkov, M.A. The development of parallel data processing algorithms in multiprocessor systems using the example of solving the problems of the dynamics of industrial robots. *Bull. RUDN Univ. Ser. Eng. Res.* **2010**, *4*, 60–72. (In Russian)
14. Zhang, X.; Sørensen, R.; Iversen, M.R.; Li, H. Computationally efficient dynamic modeling of robot manipulators with multiple flexible-links using acceleration-based discrete time transfer matrix method. *Robot. Comput. Integr. Manuf.* **2018**, *49*, 181–193. [\[CrossRef\]](#)

15. Korchagin, S.A.; Terin, D.V.; Klinaev, Y.V.; Romanchuk, S.P. Simulation of Current-Voltage Characteristics of Conglomerate of Nonlinear Semiconductor Nanocomposites. In *2018 International Conference on Actual Problems of Electron Devices Engineering (APEDE)*; IEEE: Piscataway Township, NJ, USA, 2018; pp. 397–399.
16. Krakhmalev, O. *Designing Object Diagrams and the Method of Structural Mutations in Models of Robots' Manipulation Systems*; Springer: Berlin/Heidelberg, Germany, 2019; Volume 154, pp. 209–221.
17. Krakhmalev, O.N. Use of Structural Mutations in Object-Oriented Mathematical Models of Robot Manipulation Systems. *Math. Model. Comput. Simul.* **2020**, *12*, 90–98. [[CrossRef](#)]
18. Krakhmalev, O.N. *Modeling of Robot Manipulation Systems*; IPR Media: Saratov, Russia, 2018; Available online: <https://search.rsl.ru/ru/search#q=9785448601460> (accessed on 12 September 2021).
19. Shapira, Y. *Mathematical Objects in C++: Computational Tools in A Unified Object-Oriented Approach*; CRC Press: Boca Raton, FL, USA, 2017.
20. Moiseev, G. Forecasting oil tanker shipping market in crisis periods: Exponential smoothing model application. *Asian J. Shipp. Logist.* **2021**, *37*, 239–244. [[CrossRef](#)]
21. Dogadina, E.; Smirnov, M.; Osipov, A.; Suvorov, S. Evaluation of the Forms of Education of High School Students Using a Hybrid Model Based on Various Optimization Methods and a Neural Network. *Informatics* **2021**, *8*, 46. [[CrossRef](#)]
22. Korchagin, S.; Romanova, E.; Serdechnyy, D.; Nikitin, P.; Dolgov, V.; Feklin, V. Mathematical Modeling of Layered Nanocomposite of Fractal Structure. *Mathematics* **2021**, *9*, 1541. [[CrossRef](#)]
23. Krakhmalev, O.N.; Bleyshmidt, L.I. Determination of dynamic accuracy of manipulation systems of robots with elastic hinges. *J. Mach. Manuf. Reliab.* **2014**, *43*, 22–28. [[CrossRef](#)]
24. Krakhmalev, O.N.; Petreshin, D.I.; Fedonin, O.N. Mathematical models for base calibration in industrial robots. *Russ. Eng. Res.* **2017**, *37*, 995–1000. [[CrossRef](#)]
25. Kalyaev, I.A.; Levin, I.I.; Semernikov, E.A.; Shmoilov, V.I. *Reconfigurable Multipipeline Computing Structures*; Nova Science Publishers: New York, NY, USA, 2012.
26. Dordopulo, A.; Levin, I.I.; Doronchenko, Y.I.; Raskladkin, M.K. *High-Performance Reconfigurable Computer Systems Based on Virtex FPGAs*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 349–362.
27. Irwansyah, A.; Ibraheem, O.W.; Hagemeyer, J.; Porrmann, M.; Rueckert, U. FPGA-based multi-robot tracking. *J. Parallel Distrib. Comput.* **2017**, *107*, 146–161. [[CrossRef](#)]
28. Cumplido, R.; Gokhale, M.; Feregrino, C.; Huebner, M. Guest Editorial: Special Issue on Reconfigurable Computing and FPGA Technology. *J. Parallel Distrib. Comput.* **2019**, *133*, 359–361. [[CrossRef](#)]
29. Liang, K.; Karpenko, A.P. A Modified Particle Swarm Algorithm for Solving Group Robotics Problem. *Adv. Neuroergon. Cogn. Eng.* **2020**, *1127*, 205–217. [[CrossRef](#)]
30. Krakhmalev, N.O.; Korostelyov, D.A. Solutions of the inverse kinematic problem for manipulation robots based on the genetic algorithm. *IOP Conf. Ser. Mater. Sci. Eng.* **2020**, *747*, 012117. [[CrossRef](#)]
31. Krakhmalev, O.; Krakhmalev, N.; Gataullin, S.; Makarenko, I.; Nikitin, P.; Serdechnyy, D.; Liang, K.; Korchagin, S. Mathematics Model for 6-DOF Joints Manipulation Robots. *Mathematics* **2021**, *9*, 2828. [[CrossRef](#)]