Mivar-based route planning simulation model for obstacle-aware autonomous agricultural machinery

Anton Kotsenko[1], *Oleg* Varlamov[1*], Alexey *Denisov*[1], *Alexander* Matsnev [1], and *Boris* Goryachkin[1]

[1]Bauman Moscow State Technical University, 2-ya Baumanskaya Street, 5/1, Moscow, 105005, Russian Federation

Abstract. Autonomous robot navigation is increasingly becoming an important task that requires solutions. This paper explores the practical application of logical artificial intelligence to address the problem of route planning, using the example of a computer game. Within the scope of this work, a knowledge base model was created, and a new version of the mivar reasoner was developed for integration with Unity. This reasoner allows the activation of logical rules with linear complexity. As a result, a system was developed that processes user input for position and moves an autonomous agent in a virtual environment according to the rules. This work confirms the feasibility of using mivar technologies to improve control systems of autonomous robots in the area of agriculture. The study also emphasizes the adaptability of mivar networks in dynamic environments, demonstrating their ability to effectively process changes in real-time. This research shows enhanced decision-making capabilities and reliable navigation strategies for autonomous agents, setting a precedent for future developments in autonomous digital solutions for agriculture. The results obtained open new prospects for the advancement of technologies in the field of autonomous navigation.

1 Introduction

Currently, autonomous agents are widely used in various fields: robotics, autonomous vehicles, and the gaming industry. These robots must have the ability to make decisions and navigate their environment to achieve their goals. Expert systems are one of the key tools for controlling the behavior of such autonomous robots. This paper discusses the application of mivar expert systems in the context of managing autonomous agents in a computer game.

Mivar technologies have high information processing and decision-making speeds, allowing autonomous agents to quickly respond to dynamic changes in the virtual environment and make real-time decisions. The use of the mivar approach opens up new

* Corresponding author: ovarlamov@gmail.com

possibilities for creating complex virtual worlds where autonomous agents can effectively plan routes and interact correctly with the environment in various scenarios.

The virtual environment serves as a platform for modeling and testing the behavior of autonomous robots in a virtual setting. It can be implemented as a two-dimensional or three-dimensional simulation, where agents interact with surrounding objects according to specified rules and goals. Navigation of autonomous agents in the virtual environment includes route planning, movement control, and responding to changing environmental conditions.

The development of mivar technologies in logical artificial intelligence has been going on for quite some time. These technologies have proven effective in solving tasks within the framework of production networks, as they enable finding solutions with linear computational complexity [1]. Mivar technologies are widely used for creating expert systems in various fields, such as the management of educational programs at universities [2], the detailed description of knowledge in a scientific discipline [3], the detection of bank check fraud [4], intelligent plant care systems [5], decision-making on the safety of thermolabile blood components [6], optimization of the process of preparing fresh frozen plasma for transfusion [7], diabetes diagnosis [8], automated assembly [9], mechanical engineering [10], and many other areas. Additionally, mivar technologies have found their place in the creation of hybrid artificial intelligence (AI) systems. They can be implemented in a wide range of activities, such as creating a brief overview of judicial practice [11], detecting energy theft in smart grids using explainable attention maps [12], using metagraphs to represent data sets [13] to overcome limitations [14] and improve existing knowledge bases [15], for tasks of analysis and classification [16] of equivalent logical operations [17], and solving first-order logical equations with exhaustive search for solutions [18].

The hybrid AI approach with the application of mivar technologies includes neural network methods. Neural networks can be used for sentiment analysis based on text and audio data [19], for processing media information [20] and its optimal encoding [21], for working with satellite images [22] and text queries [23]. Hybrid AI can also be applied in the development of polymer microstructures [24] and composites [25], in modeling heat exchangers [26], as well as in the formation of intelligent technological units [27]. The use of mivar technologies in conjunction with a neural network approach simplifies the process of analyzing LiDAR data for the task of finding trees and estimating their diameter [28], and for measuring active gases affecting the climate at carbon landfills [29]. Mivar technologies are capable not only of decision-making but also of performing intelligent analysis of pulsed EPR for recognizing 3D objects by an optical location system [30]. One of the most promising areas of application of the mivar approach and hybrid AI is the development of robotics [31], as well as the navigation systems of robotic complexes [32], and the creation of intelligent vehicle control systems [33]. One of the complex tasks in this area is pathfinding [34] with planning the shortest route for a robotic complex [35], which is especially important when moving autonomous transport on public roads, where many conditions need to be taken into account [36].

2 Description of the mivar knowledge base

In the implemented mivar model, there are three types of parameters. The first type is needed for planning the agent's route (Figure 1). The second type consists of parameters that describe the state of each point in the environment, in this case, passability (Figure 2). The third type is the type of vehicle (Figure 3).

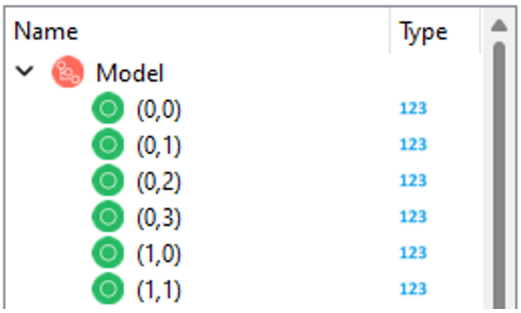


Fig. 1. Example of parameters for modeling agent movement.

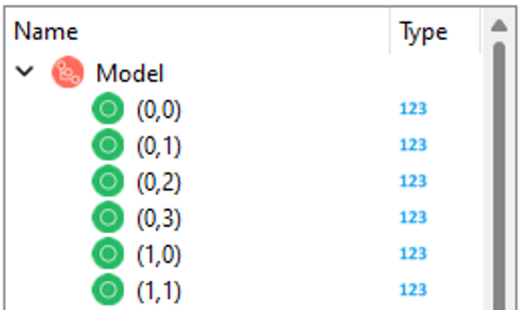


Fig. 2. Example of parameters for describing the passability of the environment.

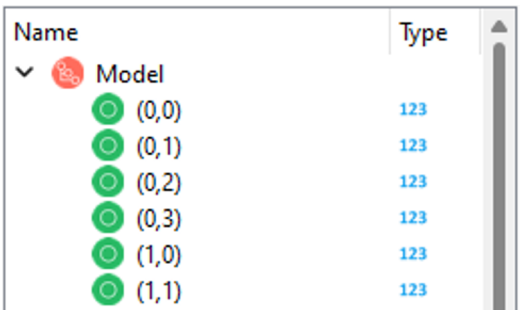


Fig. 3. Parameters for describing the passability of transport.

There is only one relation in the model. It is written as a complex relation in JavaScript (Figure 4). There are 12 transitions between adjacent points in space: 6 in one direction and 6 in the other. Accordingly, transitions are possible movement options from one point to another. For example, an all-terrain vehicle can move through fields of road, mud, and swamp.

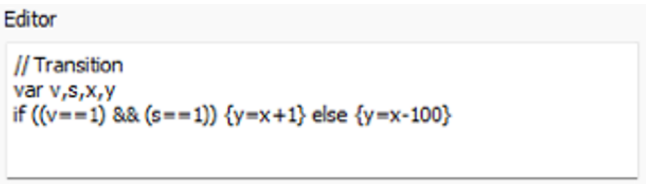


Fig. 4. Relationship script in JavaScript programming language.

3 Examples of knowledge base operation scenarios

If all parameters (S_{(x,x)_1_Dirt}) are assigned a value of "1", and the parameter (Vehicle_1_Dirt) is also assigned a value of "1", then by selecting the start and end points of the path (assigning the parameter (0,0) a value of "0" and checking the box next to the end point), we obtain the movement of an off-road vehicle across a field of dirt without movement restrictions (Figure 5).

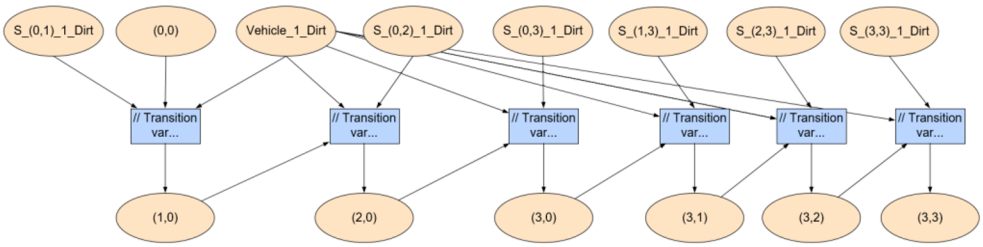


Fig. 5. An example of an SUV moving through a mud field.

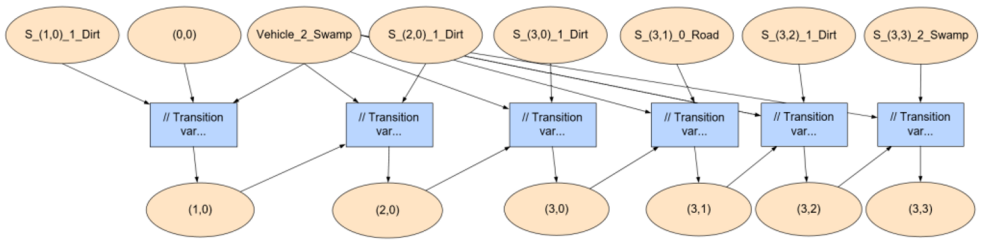


Fig. 6. An all-terrain vehicle moving through a mixed field;

In this case, the final cell (3,3) is reached in 6 steps, as indicated by the value of this parameter. On the path from start to finish, the all-terrain vehicle traversed the following fields: dirt (1,0), dirt (2,0), dirt (3,0), road (3,1), dirt (3,2), swamp (3,3) (Figure 6). Accordingly, these three experiments demonstrated the operation of the KESMI with the mivar model of a two-dimensional space, taking into account the surface condition. The developed system works correctly.

4 Development of a mivar intelligence

During the research, a logical mivar core was developed, capable of finding the required parameters based on given rules. In the work, the core is used for loading the model and planning the route. Navigation is carried out based on the passability of each cell and the type of vehicle used. The model solution is organized into three stages: loading the model, achieving all the required parameters considering known parameters and rules, and calculating the parameters to find the values of the required parameters.

The C# programming language was chosen for this purpose. This choice is due to the fact that the Unity3D modeling environment uses this language for developing three-dimensional models of the environment. Additionally, this language is relevant and used in

a large share of modern projects, has a precise update schedule, and is supported by Microsoft, ensuring its reliability and demand.

The model loading is implemented for the model format used in KESMI. This format implies saving rules, relations, and parameters, with connections established based on identifiers. The model saving format is created based on XML, and working with this markup language is supported within the standard C# library.

The algorithm for achieving all required parameters. Within the core, there is a work cycle during which new parameters are achieved. During each subsequent iteration of the cycle, the core tries to determine all the rules that can be executed based on the parameters known at that moment. Then, each output parameter that can be achieved within the selected rules will be marked as achieved.

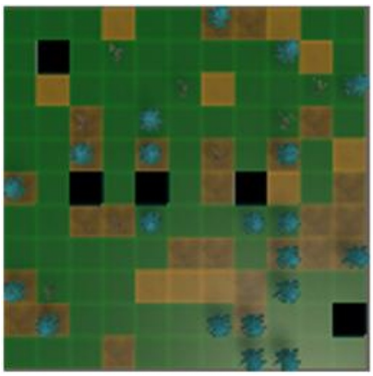
This work cycle will continue until either all required parameters are marked as achieved, or there are no more rules that can be computed in this iteration and have not been computed before. Thus, the core will either be able to achieve all required parameters and build an algorithm for their computation, or it will throw an exception indicating that a solution cannot be found for the given parameters. The parameter computation follows the found solution algorithm. If the required parameters were achieved during the second stage, all the rules used for this will be computed.

The algorithm for selecting rules and parameters in the second stage of work. After loading the model into memory, a data model will be created in which the rules contain identifiers of input and output parameters. The parameters, in turn, contain identifiers of the rules for which the selected parameter is an input. Thus, the rules “know” which parameters can be computed when they are used, and the parameters “know” the rules dependent on them. This allows organizing the computations according to the previously described cycle.

5 Machine vision model

Technical vision was implemented for specific usage conditions and can serve as a basis for further developments. It is based on a neural network implemented using the U-net architecture. The input data for this network is a 128x128 image depicting a field within which the agent moves. The field, in turn, consists of square cells, with a size of 11x11 cells. Each cell can have one of four passability states. Thus, the technical vision system must recognize individual cells and determine the state of each.

The neural network was trained in the Google Colab environment for 10 epochs with a batch size of 64 and a learning rate of $5*10^{-5}$. As a result of the initial training stage, an accuracy of 98 percent was achieved, which is an acceptable accuracy for this stage of model training. The neural network is based on the U-net architecture with an additional activation output layer that converts the probabilities of detecting four states into one of the desired states. A data set generator was developed based on the first version of the environment for training the neural network. The main goal of creating the generator is to automatically obtain a set of training and validation data in the form of images and their corresponding state matrices. In the work, 500 images were generated, and the field state matrices were recorded in a JSON file. An example of one image and its state matrix is shown in Figure 7.




```
{
  "name": "image_matrix_0",
  "matrix": [
    [0, 0, 0, 2, 0, 0, 0, -1, -1, 0, 0],
    [2, -1, 0, 0, 0, 0, -1, -1, 0, 0, -1],
    [-1, -1, 0, 0, 1, 1, 1, 2, -1, 0, 0],
    [0, 0, 0, 0, 0, 2, 2, 0, -1, 2, -1],
    [0, 0, 2, -1, -1, 0, 0, -1, -1, 2, 2],
    [-1, 0, -1, 0, -1, 0, 2, -1, 1, 0, 2],
    [0, 0, -1, 0, -1, 0, -1, 0, -1, 0, 1],
    [0, 0, -1, 0, -1, 0, 0, 0, -1, 2, 0],
    [0, 1, 0, 0, 0, -1, 1, 0, 0, -1, -1],
    [0, -1, 0, -1, 0, 0, 0, 0, -1, 1, 0],
    [0, 0, 0, 1, 0, 0, -1, 2, 1, 0, 0]
  ]
},
```


Fig. 7. Example of the dataset generator in operation.

6 Game design

An important aspect of this work is the creation of an attractive interactive virtual environment, designed in accordance with generally accepted game design principles. The virtual environment should demonstrate the operation of the technical vision system integrated with the mivar navigation system of the autonomous agent. The game is designed in a minimalist graphic style with a map divided into tiles—small square fragments of the same size. The setting is an abstract forest thicket where logging operations are carried out.

The idea of the game environment is to exploit the mechanics of moving an agent controlled by the user. Accordingly, the gameplay involves the player determining the end point of the agent's movement and selecting the appropriate vehicle for the final route. These vehicles will have different speeds and passability on certain types of surfaces.

Car – A high-speed vehicle capable of moving only on one surface – road (grass). Off-road vehicle – A medium-speed vehicle capable of moving on grass and dirt (sand). All-terrain vehicle – A slowest vehicle capable of moving on grass, dirt, and swamp.

Thus, according to this description, the project is presented in the point-and-click genre. The player's ultimate goal is to collect a certain amount of cargo within a limited time.

For a more advanced demonstration of the terrain recognition and agent navigation technologies implemented in the game based on mivar technologies, the terrain will dynamically change during the game session. For this, wind and rain effects are added to the game, causing contextual events such as trees falling from one cell to another and the appearance of puddles.

7 Selecting a modeling environment

A game engine (modeling environment) is software designed for developing computer games and other interactive programs that process real-time graphics. The list of the most popular ones includes: Unreal Engine by Epic Games, CryEngine by Crytek, Unity by Unity Technologies, GameMaker by Opera Gaming, Godot by The Godot Foundation, Panda3D by Disney, Construct by Scirra, and others.

Many of them are aimed at developing simple and small projects and do not offer advanced tools for working with artificial intelligence. Therefore, Unreal Engine, Unity, and Godot will be considered as the most accessible and multifunctional engines. Based on the empirical method of hierarchy analysis, a ranked list of game engines was obtained:

Unity, Unreal Engine, Godot. Unity is a high-performance and intuitive development environment that is relatively easy to master and has extensive documentation. This environment was used in the development of the software system.

8 Development of software system

In accordance with the project plan, work on visual design has been completed, specifically: fragments of surfaces such as grass (road), dirt (sand), and water (swamp) have been created; three types of vehicles and their models have been defined—car, off-road vehicle, and all-terrain vehicle; several models of trees, as well as secondary objects such as stones and logs, have been selected. Through several iterations, the game map and the angle of the virtual camera have been determined (Figure 8).

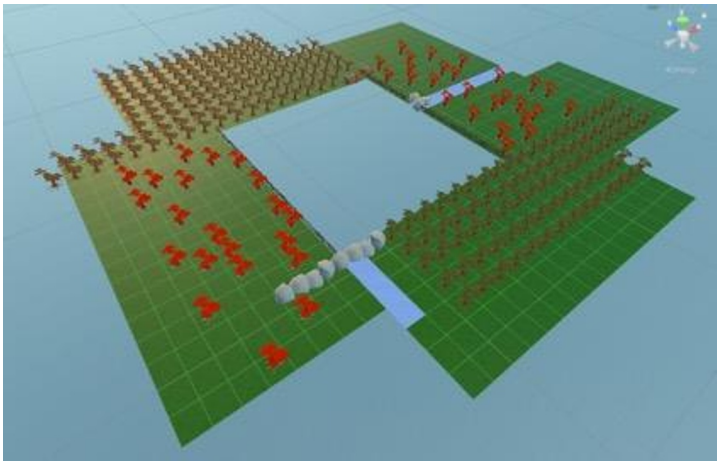


Fig. 8. View of the game scene in the editor at an angle.

To demonstrate the implemented technologies and increase user interest, a script for procedural generation of an interactive map segment measuring 11x11 tiles was written. The generation script details some features of constructing this segment, namely: random uniform distribution of all types of surfaces and trees; partially random appearance of bridges over the river (swamp); the appearance of the agent on the grass; contextual events—randomly distributed tree falls (not on bridges and not on other trees) and the appearance of puddles.

Next, a script for moving the agent controlled by the user was created, and the appearance of the cursor was determined (Figure 9). This script incorporates terrain recognition and agent navigation technologies based on mivar technologies. The player uses the mouse to indicate the end point (tile), and the navigation system, based on the map processed by technical vision, builds the shortest route accessible for the specified type of vehicle. If the route cannot be built under the current conditions, the game notifies the player with a signal (cursor symbol with prohibition).

Fig. 9. Player movement on the map using the point-and-click method.

The player's ultimate goal is to collect cargo within the allotted time. The cargo will consist of bundles of logs prepared according to the game's context. These will be highlighted with special interface elements. Next, we move on to the aspects of developing a cohesive user interface. A separate scene with the main menu was prepared for the game, and all interface elements necessary for the following actions were defined (Figure 10): obtaining information about the current task; a timer indicating the time within which the task must be completed; buttons for restarting and returning to the main menu; buttons for selecting the vehicle with descriptions of each.

Fig. 10. All user interface elements.

The condition for victory is to collect all the cargo before the timer runs out. If the player fails to do so, they lose. Background melodies have been added to the main menu and game level to provide pleasant sound accompaniment. There are also sounds of rain, wind, falling trees, and cargo collection. After refining all game aspects, the final product was tested by the developer and several independent users.

9 Conclusion

The article implements a system for route planning using mivar technologies and a system for recognizing the passability types of areas. As a result of the work, the application of the mivar approach for controlling agents in a single-player computer game is demonstrated. This allows for a higher level of autonomy and intelligence in robotic systems, which also significantly impacts the development of artificial intelligence in mechanical engineering..

References

1. O. Varlamov, Big Data Research **25**, 100241 (2021) https://doi.org/10.1016/j.bdr.2021.100241
2. T. Guzeva, A. Parsheva, V. Babin, at al., Lecture Notes in Networks and Systems **509**, 651–659 (2023) https://doi.org/10.1007/978-3-031-11058-0_65
3. T. Guzeva, S. Egorov, K. Smetankin, at al., Lecture Notes in Networks and Systems **509**, 643–650 (2023) https://doi.org/10.1007/978-3-031-11058-0_64
4. U. Grigorev, Y. Shashkin, A. Ploutenko, et al., *Bank Checks Fraud Detection Based on the Analysis of Event Trends in Data-Flow Systems, in Proceedings of the* International Conference on Internet of Things, Big Data and Security, IoTBDS Proceedings, vol. 2023-April, pp. 97-104 (2023) https://doi.org/10.5220/0011727300003482
5. D. V. Aladin, E. V. Aladina, D. A. Chuvikov, at al., IOP Conference Series: Earth and Environmental Science **954(1)**, 012004 (2022) https://doi.org/10.1088/1755-1315/954/1/012004
6. O. O. Varlamov, D. A. Chuvikov, V. N. Lemondzhava, at al., Biomedical Engineering **55(5)**, 355–359 (2022) https://doi.org/10.1007/s10527-022-10135-0
7. V. N. Lemondzhava, T. Yu. Lemondzhava, A. G. Gudkov, et al., AIP Conference Proceedings **2605** (2023) https://doi.org/10.1063/5.0110400
8. D. Shamaev, Lecture Notes in Networks and Systems **597**, 519-528 (2023) https://doi.org/10.1007/978-3-031-21438-7_41
9. A. Andreev, A. Kotsenko, R. Kim, at al., E3S Web of Conferences **549**, 08008 (2024) https://doi.org/10.1051/e3sconf/202454908008
10. A. Volkov, O. Varlamov, Journal of Physics: Conference Series **2131(3)**, 032003 (2021) https://doi.org/10.1088/1742-6596/2131/3/032003
11. M. O. Taran, G. I. Revunkov, Y. E. Gapanyuk, Studies in Computational Intelligence **1064**, 466-475 (2023) https://doi.org/10.1007/978-3-031-19032-2_48
12. D. O. Ishkov, V. I. Terekhov, K. S. Myshenkov, *Energy Theft Detection in Smart Grids via Explainable Attention Maps*, in Proceedings of the 2023 5th International Youth Conference on Radio Electronics, Electrical and Power Engineering, REEPE 2023 (2023) https://doi.org/10.1109/REEPE57272.2023.10086919
13. Y. E. Gapanyuk, Pattern Recognition and Image Analysis **33(3)**, 300-305 (2023) https://doi.org/10.1134/S1054661823030124
14. D. Fedyukin, D. Gromozdov, Y. Gapanyuk, Studies in Systems, Decision and Control **457**, 207-218 (2023) https://doi.org/10.1007/978-3-031- 22938-1_14
15. R. Tsarev, S. H. Azizam, A. Sablinskii, et al., Lecture Notes in Networks and Systems **723**, 209-216 (2023) https://doi.org/10.1007/978-3-031-35317-8_18
16. D. A. Gurianov, K. S. Myshenkov, V. I. Terekhov, *Software Development Methodologies: Analysis and Classification*, in Proceedings of the 2023 5th

International Youth Conference on Radio Electronics, Electrical and Power Engineering, REEPE 2023 (2023) https://doi.org/10.1109/REEPE57272.2023.10086852

17. V. S. Popov, *Equivalence of logical operations and other operations in Python programming language*, in Proceedings of the 2023 5th International Youth Conference on Radio Electronics, Electrical and Power Engineering, REEPE 2023 (2023) https://doi.org/10.1109/REEPE57272.2023.10086928
18. V. S. Popov, *First-Order Logical Equations with Parameter and their Exhaustive Search Solutions*, in Proceedings of the 2023 5th International Youth Conference on Radio Electronics, Electrical and Power Engineering, REEPE 2023 (2023) https://doi.org/10.1109/REEPE57272.2023.10086751
19. Y. Verina, D. Tolstoukhov, D., Egorov, et al., AIP Conference Proceedings **2819** (2023) https://doi.org/10.1063/5.0137948
20. A. Kanev, M. Nazarov, D. Uskov, V. Terentyev, *Research of Different Neural Network Architectures for Audio and Video Denoising*, in Proceedings of the 2023 5th International Youth Conference on Radio Electronics, Electrical and Power Engineering, REEPE 2023 (2023) https://doi.org/10.1109/REEPE57272.2023.10086862
21. M. V. Belodedov, R. V. Fonkants, R. R. Safin, *Development of an Algorithm for Optimal Encoding of WAV Files Using Genetic Algorithms*, in Proceedings of the 2023 5th International Youth Conference on Radio Electronics, Electrical and Power Engineering, REEPE 2023 (2023) https://doi.org/10.1109/REEPE57272.2023.10086837
22. A. O. Kanev, A. V. Tarasov, A. N. Shikhov, et al., Sovremennye Problemy Distantsionnogo Zondirovaniya Zemli iz Kosmosa **20(3)**, 136-151 (2023) https://doi.org/10.21046/2070-7401-2023-20-3-136-151
23. R. Kim, A. Kotsenko, A. Andreev, at al., E3S Web of Conferences **515**, 03016 (2024) https://doi.org/10.1051/e3sconf/202451503016
24. R. Kim, A. Kotsenko, A. Andreev, at al., E3S Web of Conferences **549**, 08009 (2024) https://doi.org/10.1051/e3sconf/202454908009
25. D. S. Tikhonova, A. Y. Abramochkin, A. S. Borodulin, et al., Polymer Science - Series D **16(2)**, 307-312 (2023) https://doi.org/10.1134/S1995421223020442
26. A. Valishin, N. Beriachvili, E3S Web of Conferences **376** (2023) https://doi.org/10.1051/e3sconf/202337601041
27. N. A. Lavrov, S. I. Khutsieva, V. A. Shananin, Chemical and Petroleum Engineering **58(11-12)**, 917-924 (2023) https://doi.org/10.1007/s10556-023-01183-8
28. V. V. Bukhtoyarov, V. S. Tynchenko, V. A. Nelyub, et al., Electronics (Switzerland) **12(1)** (2023) https://doi.org/10.3390/electronics12010215
29. I. A. Grishin, V. I. Terekhov, *Procedure for Locating Trees and Estimating Diameters Using LiDAR Data*, in Proceedings of the 2023 5th International Youth Conference on Radio Electronics, Electrical and Power Engineering, REEPE (2023) https://doi.org/10.1109/REEPE57272.2023.10086843
30. V. V. Dyachenko, V. A. Devisilov, V. G. Shemanin, Ecology and Industry of Russia **27(6)**, 30-35 (2023) https://doi.org/10.18412/1816-0395-2023-6-30-35
31. L. Labunets, Proceedings of SPIE - The International Society for Optical Engineering **12564** (2023) https://doi.org/10.1117/12.2669242
32. O. Varlamov, D. A. Aladin, New Generation of Rules-based Approach: Mivar-based

Intelligent Planning of Robot Actions (MIPRA) and Brains for Autonomous Robots. Machine Intelligence Research (2024) https://doi.org/10.1007/s11633-023-1473-1

33. A. D. Volgina, D. S. Kirillov, A. N. Kravtsov, et al., *The Robot-Guide for Indoor Navigation*, in Proceedings of the 2023 5th International Youth Conference on Radio Electronics, Electrical and Power Engineering, REEPE 2023 (2023) https://doi.org/10.1109/REEPE57272.2023.10086707
34. D. Aladin, A. Kotsenko, R. Kim, at al., E3S Web of Conferences **549**, 08007 (2024) https://doi.org/10.1051/e3sconf/202454908007
35. A. Kotsenko, A. Andreev, R. Kim, at al., E3S Web of Conferences **515**, 04018 (2024) https://doi.org/10.1051/e3sconf/202451504018
36. M. Potashnikov, V. Shishkina, A. Muravev, A. Kartashov, E3S Web of Conferences **402** (2023) https://doi.org/10.1051/e3sconf/2023402