Evaluation of the Effectiveness of Network-Centric Control of Mobile Agents in a Dynamic Environment Using Neural Networks

Vadym Mukhin, Valerii Zavgorodnii, Anna Zavgorodnya, Oleksandr Yarovyi, Lesia Baranovska and Oleg Mukhin

National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Prospect Beresteiskyi, Kyiv, 03056, Ukraine

Abstract

This paper examines the network-centric approach for enhancing the efficiency of mobile agent control. The primary distinction of this approach is the use of a "wandering center." If communication with the active center is lost, a new center is appointed to continue control, ensuring the system's survivability and improving the overall efficiency of mobile agent management.

An experimental study was conducted to evaluate the parameters and effectiveness of network-centric control of mobile agents. The study utilized a developed model of mobile agent interaction based on network-centric control, along with a simulation model of agent behavior in a dynamic environment. Additionally, a neural network was created that accurately predicts the probability of hitting the target (%) under changing dynamics. The use of this neural network also helped identify input parameters with minimal impact on the outcome.

Keywords

Mobile agent, network-centric control, dynamic environment, neural networks, simulation model

1. Introduction

Today, most existing mobile agents are controlled manually using remote controls that operate on radio channels. However, this manual control poses several challenges, including the need for specialized operator training, limited operational range, and dependency on weather conditions [1, 2].

A mobile agent can be a software or hardware entity capable of performing various tasks within a network or on a device, such as carrying information, conducting computations, and interacting with other agents or the environment [3, 4].

Controlling mobile agents requires qualified specialists. For example, in the U.S. military, experienced Air Force pilots undergo a full year of training to become proficient mobile agent operators – a task that, in some cases, can be more demanding than piloting an aircraft. Operator errors and mechanical failures account for most mobile agent accidents [5, 6].

A new technological trend involves the development of mobile agents equipped with an even number of rotors that rotate diagonally in opposite directions. Mobile agents are just one component within a complex, multi-functional system [7, 8]. Unlike manned aircraft, operating mobile agents requires additional support system components, including the agent itself, the operator's workstation, software, data lines, and other elements necessary to achieve mission objectives. Current development trends favor compact mobile agents, with a focus on simplicity of control, reliability, and maneuverability [9, 10].

ITS-2023: Information Technologies and Security, November 30, 2023, Kyiv, Ukraine
v.mukhin@kpi.ua (V. Mukhin); zavgorodniivalerii@gmail.com (V. Zavgorodnii); annzavgorodnya@gmail.com (A. Zavgorodnya); getem13@ukr.net (O. Yarovyi); lesia@baranovsky.org (L. Baranovska); o.mukhin01@gmail.com (O. Mukhin)
0000-0002-1206-9131 (V. Mukhin); 0000-0002-8347-7183 (V. Zavgorodnii); 0000-0001-8523-1761 (A. Zavgorodnya); 0000-0002-3889-5730 (O. Yarovyi); 0000-0003-0024-8180 (L. Baranovska); 0009-0005-5301-8276 (O. Mukhin)
© 2023 Copyright for this paper by its authors.
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR-WS.org/Vol-3887/paper16.pdf

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

In model aviation and professional applications such as the civil sector, agriculture, military, law enforcement, and other fields, mobile agents are in high demand. Selecting the optimal models and control systems is crucial for effectively monitoring ground-based targets [11, 12].

Thus, pilotless mobile agents present a more efficient and economical alternative to manned aircraft for many tasks [13–15].

The main challenge in controlling mobile agents lies in the reliance on centralized control systems, where the control center serves as a vulnerable element; if communication with it is lost, further control becomes impossible. In contrast, decentralized control faces issues with coordinating and circulating large volumes of information, leading to slower response times. A promising alternative is a network-centric approach, which allows control to be transferred to an alternative center when needed [16–18].

An analysis of recent scientific literature reveals that, although modern information technologies have advanced considerably, the network-centric approach to control remains underdeveloped compared to centralized and decentralized methods.

2. A network-centric approach to enhancing the efficiency of mobile agent control systems

Three main approaches to the control of mobile agents can be identified: centralized, decentralized, and network-centric. In centralized control, a single command center issues control signals to all mobile agents. If this center is disabled or compromised, all mobile agents lose connectivity and cannot be controlled.

Decentralized control, while reducing dependence on a single command point, has the drawback of coordination challenges. For example, if mobile agents need to rapidly reconfigure, a large amount of information must be transmitted to them, significantly reducing the overall speed of the control system [19, 20].

The network-centric approach is designed to address the limitations of the previous models. This approach integrates all forces and resources into a single information system, enabling control objectives to be met even in dynamic, complex environments that are subject to unpredictable interference. Such interference can be irregular, intermittent, and variable, yet network-centric control is still effective under these conditions [21, 22].

Consider a control network that a subset of mobile agents follows. Among the entire group of mobile agents, approximately 10% are selected as nodes that hold partial control information (Fig. 1). These nodes, equipped with control functions, form a network, and from them, one is designated as the primary control center [23, 24].

Only the node mobile agents (NMA) are coordinated directly by the control center; these nodes, in turn, control the remaining mobile agents (MA). This setup minimizes communication interference since individual control of each node could overwhelm communication channels. For example, if the command is to change direction quickly, adjusting the coordinates for each mobile agent individually would require significant time. Furthermore, if there is only one primary control agent and it suddenly stops responding, is damaged, or is compromised, all mobile agents would lose direction, become vulnerable to interception, or cease to operate [25–27].

A network-centric system consists of a primary control mobile agent (MMA) that continuously sends commands to other mobile agents, setting parameters such as movement direction [28, 29]. Alongside commands, it also relays information from the control center to the node mobile agents. If the primary control agent fails for any reason, copies of all information, including control data for all mobile agents, are preserved within the node mobile agents. In this way, an adversary would need to disable every node agent to compromise the entire network – a challenging task since the enemy cannot readily identify which agents' function as nodes. This network organization enhances the resilience of the control system.

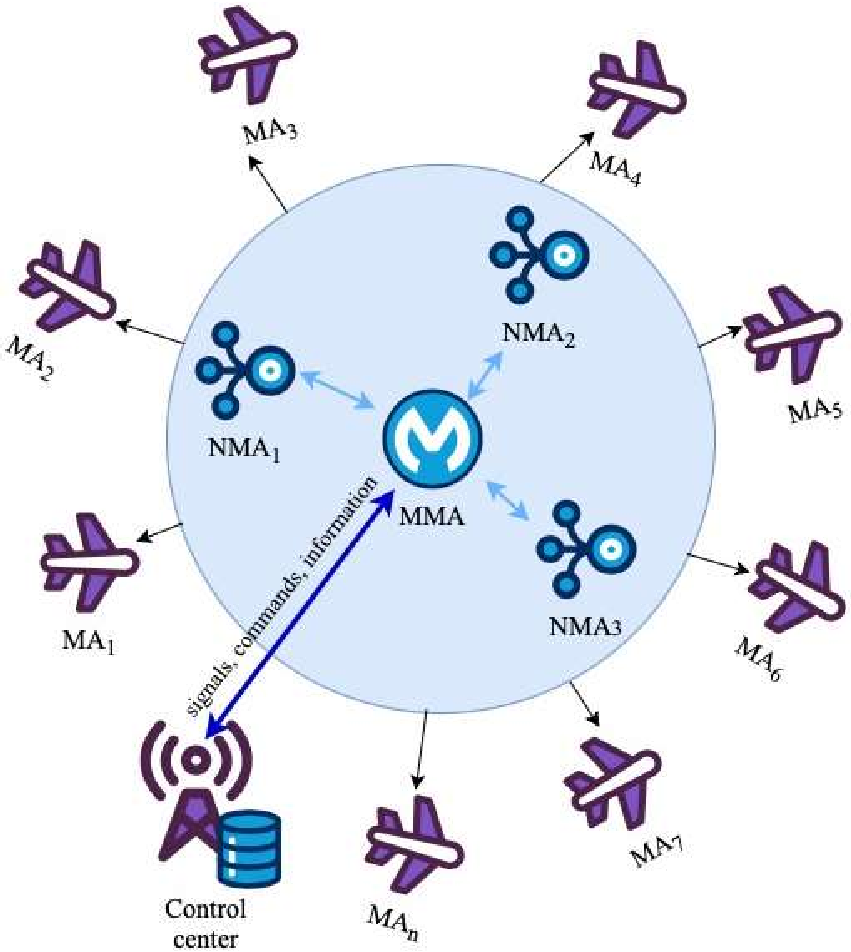


Figure 1: Mobile agent control system based on network-centric control

A key feature of a network-centric system is its "wandering center." If the primary control mobile agent is identified and disabled by an adversary, the stability of the system remains unaffected, as control can quickly be reassigned. A new control agent is selected from the remaining active node agents, allowing for seamless continuity. This flexibility means that control can be handed over to another node at any moment, ensuring sustained operation. In this network-centric mobile agent control system, two types of information are managed:

- control: this includes global coordinates transmitted from the control center to the main control agent.
- command: this includes the coordinates transmitted by the control agent to node agents, which then relay local coordinates to other agents. Thus, mobile agents navigate based on local rather than global coordinates, as provided by the node agents.

In summary, a key strength of the network-centric approach is the use of a roving control center. If connections to the current center are lost, a new center is assigned to maintain control. This design ensures system survivability, thereby greatly enhancing the overall effectiveness of mobile agent control.

3. Evaluation of the Effectiveness of Network-Centric Control for Mobile Agents in a Dynamic Environment

Based on the network-centric control algorithm model for mobile agents in dynamic environments and the simulation model of interacting mobile agents, it is essential to conduct experimental studies to assess parameters and evaluate the effectiveness of network-centric control.

To validate the scientific and methodological framework developed, we performed mathematical modeling to analyze the effectiveness of network-centric control for mobile agents in a dynamic environment.

The software tools used in this study include:

- Simulations were performed on a personal computer with an Intel® Celeron® processor (3.2 GHz) and 16 GB of RAM running on Microsoft Windows 10.
- Microsoft SQL Server 2022 was utilized as the database management system.
- Microsoft Visual Studio 2022 Community Edition served as the development environment. Development was conducted in C# using an object-oriented approach.
- Experiments in the simulation environment included setting initial parameters, performing intermediate calculations, and visualizing results with Windows Presentation Foundation.

The minimum required percentage of node mobile agents % (nodePercentage) needed to reach the target was calculated as follows. A target is considered "hit" if both of the following conditions are met:

- The percentage of mobile agents that reached the target (targetHitPercentage) must be at least equal to the required percentage of mobile agents needed to hit the target (targetHitPercentage).
- Among the mobile agents that reached the target, there must be at least one node mobile agent (targetReachedCount > 0).

To evaluate these conditions, 100 experimental trials were conducted, using the following parameters:

- Number of mobile agents (numberOfAgents): 100.
- Minimum percentage of mobile agents required to hit the target (targetHitPercentage): 5%.
- Percentage of node mobile agents (nodePercentage): ranging from 1% to 20% of the total number of mobile agents.
- Initial (startLatitude, startLongitude) and target (targetLatitude, targetLongitude) geographic coordinates for node mobile agents.
- Percentage of mobile agents that lost communication (lossPercentage), randomly generated between 50% and 90%. For each agent that lost communication, the geographic coordinates were set to NULL.

The experimental results were used to plot graphs illustrating the relationship between the percentage of node mobile agents % (nodePercentage) and the probability of hitting the target (targetHitPercentage) in each experiment (Fig. 2).

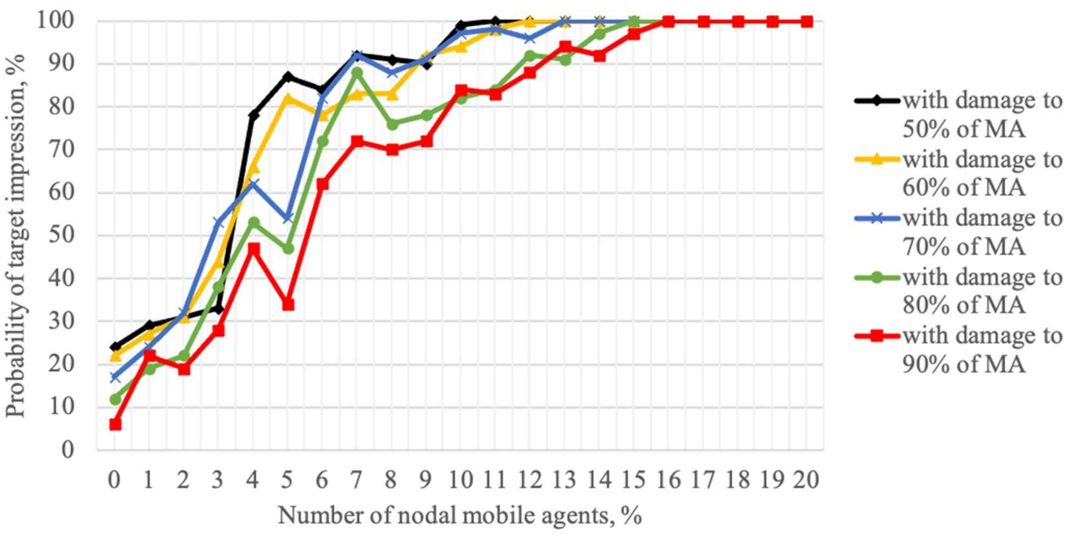


Figure 2: Relationship between the percentage of node mobile agents and the probability of hitting the target

Based on the experimental results, the average effectiveness of target hits with varying numbers of node mobile agents was analyzed (see Table 1 and Fig. 3).

Table 1
Target-Hitting Effectiveness with Varying Numbers of Node Mobile Agents

Number of nodal mobile agents, %	Number of mobile agents with which communication was lost, %				
	up to 50%	up to 60%	up to 70%	up to 80%	up to 90%
0	24	22	17	12	6
1	29	27	24	19	22
2	31	31	32	22	19
3	33	44	53	38	28
4	78	66	62	53	47
5	87	82	54	47	34
6	84	78	82	72	62
7	92	83	92	88	72
8	91	83	88	76	70
9	90	92	91	78	72
10	99	94	97	82	84
11	100	98	98	84	83
12	100	100	96	92	88
13	100	100	100	91	94
14	100	100	100	97	92
15	100	100	100	100	97
16	100	100	100	100	100
17	100	100	100	100	100
18	100	100	100	100	100
19	100	100	100	100	100
20	100	100	100	100	100

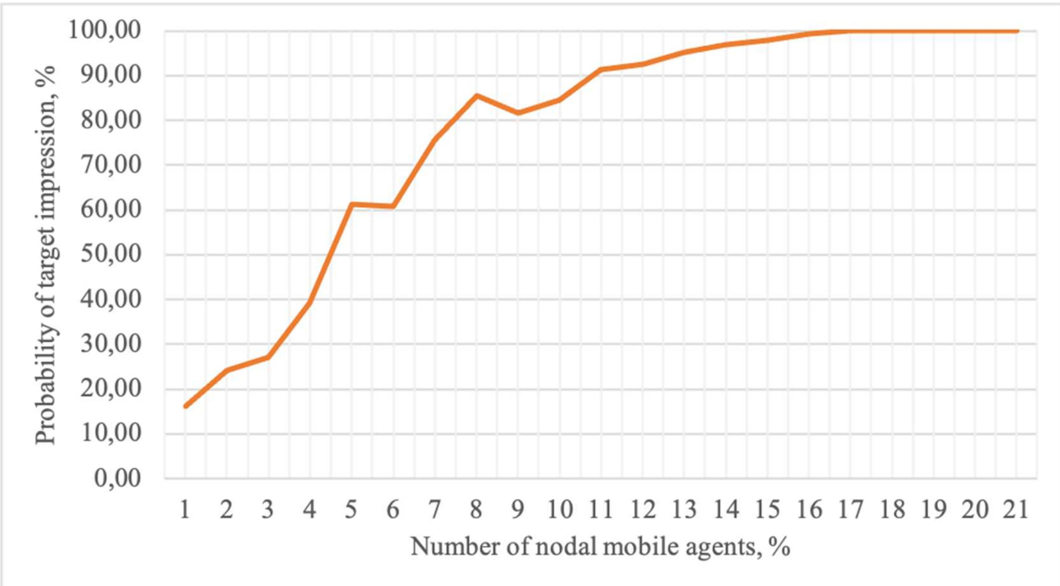


Figure 3: Average Effectiveness of Target Hits with Varying Numbers of Node Mobile Agents

Figure 3 indicates the average effectiveness of target hits based on the percentage of node mobile agents:

- 16% or more node mobile agents: 100% effectiveness.
- 12–15% node mobile agents: 95% effectiveness.
- 10–11% node mobile agents: 90% effectiveness.
- 7–9% node mobile agents: 80% effectiveness.
- 0–6% node mobile agents: less than 80% effectiveness.

From the above, it can be concluded that to achieve effective target damage (80% and above), the minimum required number of node mobile agents (nodePercentage) is between 7% and 9%. Therefore, using more than 9% of node mobile agents out of the total number of mobile agents is not advisable.

4. Evaluation of the Effectiveness of Network-Centric Control for Mobile Agents in a Dynamic Environment Using Neural Networks

Evaluating the effectiveness of network-centric control for mobile agents in a dynamic environment, using neural networks, plays a crucial role in determining the suitability of this approach for various tasks. The main stages of this method can be described as follows:

1. Training Data Preparation: collect and prepare a large dataset representing different scenarios and dynamic environmental conditions.
2. Creating a Dataset for Network Training: split the collected data into training and test sets to evaluate the network's adaptability to new situations.
3. Designing and Configuring the Neural Network:
   - develop a neural network architecture suited for controlling mobile agents in dynamic environments.
   - set the parameters and functions to optimize training.
4. Neural Network Training: use the training set to teach the network to recognize and solve agent control tasks in various environments.
5. Validation and Performance Evaluation:
   - use the test set to assess the network's accuracy and efficiency under real-world conditions.
   - analyze the results in the context of specific tasks and environmental constraints.
6. Improvement and Adaptation:
   - use the results to refine the network architecture and its parameters.
   - adapt the network to changes in environmental dynamics and control tasks.

The process of training and creating the neural network was carried out using the Deductor Studio Academic analytical platform, which provides highly efficient tools for data analysis and processing. A critical step in this process was utilizing a specially created and carefully prepared dataset to train the network.

The dataset included information from solving 1,000 different problems, which played a key role in enhancing the adaptability and accuracy of the neural network. This approach to data preparation contributed significantly to the network's ability to solve diverse tasks and ensured its high efficiency across various contexts.

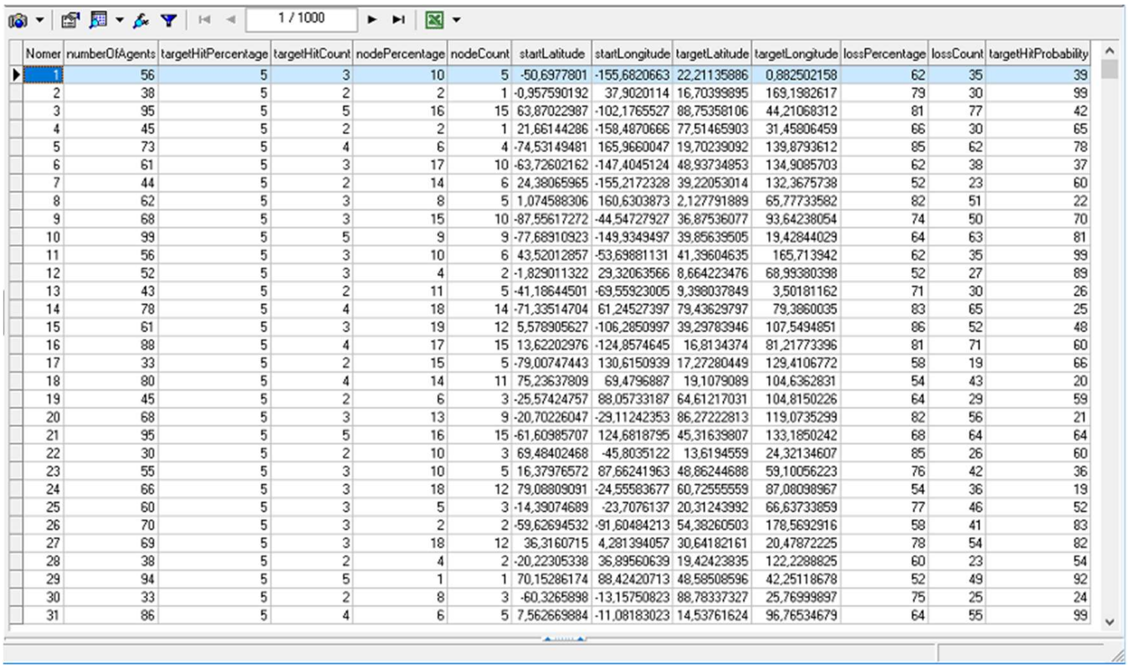
For the parameters listed in Table 2, a mechanism for estimating the probability of target damage (%) based on the neural network can be applied.

Table 2
The main parameters of the task

Parameter	Description
number	Mobile agent number
numberOfAgents	Number of mobile agents, units

targetHitPercentage	The number of mobile agents that will be enough to hit the target, %
targetHitCount	The number of mobile agents that will be enough to hit the target, units.
nodePercentage	Number of nodal mobile agents, %
nodeCount	Number of nodal mobile agents, units
startLatitude	Initial geographic coordinates of the mobile agent (latitude), degrees.
startLongitude	Initial geographic coordinates of the mobile agent (longitude), degrees.
targetLatitude	Final geographical coordinates of the mobile agent (latitude), degrees.
targetLongitude	Final geographic coordinates of the mobile agent (longitude), degrees.
lossPercentage	Number of mobile agents with which communication was lost, %
lossCount	The number of mobile agents with which communication was lost, units.
targetHitProbability	Probability of hitting the target, %

The parameters from Table 2 were provided as input data for building a neural network (Fig. 4).



Nomer	numberOfAgents	targetHitPercentage	targetHitCount	nodePercentage	nodeCount	startLatitude	startLongitude	targetLatitude	targetLongitude	lossPercentage	lossCount	targetHitProbability
1	56	5	3	10	5	-50,6977801	-155,6820663	22,21135886	0,882502158	62	35	39
2	38	5	2	2	1	-0,957590192	37,9020114	16,70399895	169,1982617	79	30	99
3	95	5	5	16	15	63,87022987	-102,1765527	88,75358106	44,21068312	81	77	42
4	45	5	2	2	1	21,66144286	-158,4870666	77,51465903	31,45806459	66	30	65
5	73	5	4	6	4	-74,53149481	165,9660047	19,70239092	139,8793612	85	62	78
6	61	5	3	17	10	-63,72602162	-147,4045124	48,93734853	134,9085703	62	38	37
7	44	5	2	14	6	24,38065965	-155,2172328	39,22053014	132,3675738	52	23	60
8	62	5	3	8	5	1,074588306	160,6303873	2,127791889	65,77733582	82	51	22
9	68	5	3	15	10	-87,55617272	-44,54727927	36,87536077	93,64238054	74	50	70
10	99	5	5	9	9	-77,68910923	-149,9349497	39,85639505	19,42844029	64	63	81
11	56	5	3	10	6	43,52012857	-53,69881131	41,39604635	165,713942	62	35	99
12	52	5	3	4	2	-1,829011322	29,32063566	8,664223476	68,99380398	52	27	89
13	43	5	2	11	5	-41,18644501	-69,55923005	9,398037849	3,50181162	71	30	26
14	78	5	4	18	14	-71,33514704	61,24527397	79,43629797	79,3860035	83	65	25
15	61	5	3	19	12	5,578905627	-106,2850997	39,29783946	107,5494851	86	52	48
16	88	5	4	17	15	13,62202976	-124,8574645	16,8134374	81,21773396	81	71	60
17	33	5	2	15	5	-79,00747443	130,6150939	17,27280449	129,4106772	58	19	66
18	80	5	4	14	11	75,23637809	69,4796887	19,1079089	104,6362831	54	43	20
19	45	5	2	6	3	-25,57424757	88,05733187	64,61217031	104,8150226	64	29	59
20	68	5	3	13	9	-20,70226047	-29,11242353	86,27222813	119,0735299	82	56	21
21	95	5	5	16	15	-61,60985707	124,6818795	45,31639807	133,1850242	68	64	64
22	30	5	2	10	3	69,48402468	-45,8035122	13,6194559	24,32134607	85	26	60
23	55	5	3	10	5	16,37976572	87,66241963	48,86244688	59,10056223	76	42	36
24	66	5	3	18	12	79,08809091	-24,55583677	60,72555559	87,08098967	54	36	19
25	60	5	3	5	3	-14,39074689	-23,7076137	20,31243992	66,63733859	77	46	52
26	70	5	3	2	2	-59,62694532	-91,60484213	54,38260503	178,5692916	58	41	83
27	69	5	3	18	12	36,3160715	4,281394057	30,64182161	20,47872225	78	54	82
28	38	5	2	4	2	-20,22305338	36,89560639	19,42423835	122,2288825	60	23	54
29	94	5	5	1	1	70,15286174	88,42420713	48,58508596	42,25118678	52	49	92
30	33	5	2	8	3	-60,3265898	-13,15750823	88,78337327	25,76999897	75	25	24
31	86	5	4	6	5	7,562669884	-11,08183023	14,53761624	96,76534679	64	55	99

Figure 4: Input Data Set for Neural Network Construction

Based on the results of the correlation analysis of the raw data, it was determined that not all fields should be used for neural network training. Only the fields listed in Table 3 are relevant, as the remaining fields have an insignificant impact on the resulting value (Fig. 5).

Table 3
Input data for neural network training

Parameter	Description
number	Mobile agent number
numberOfAgents	Number of mobile agents, units
targetHitPercentage	The number of mobile agents that will be enough to hit the target, %
targetHitCount	The number of mobile agents that will be enough to hit the target, units.
nodePercentage	Number of nodal mobile agents, %
nodeCount	Number of nodal mobile agents, units
lossPercentage	Number of mobile agents with which communication was lost, %
lossCount	The number of mobile agents with which communication was lost, units.
targetHitProbability	Probability of hitting the target, %
targetHitProbability	Probability of hitting the target, %

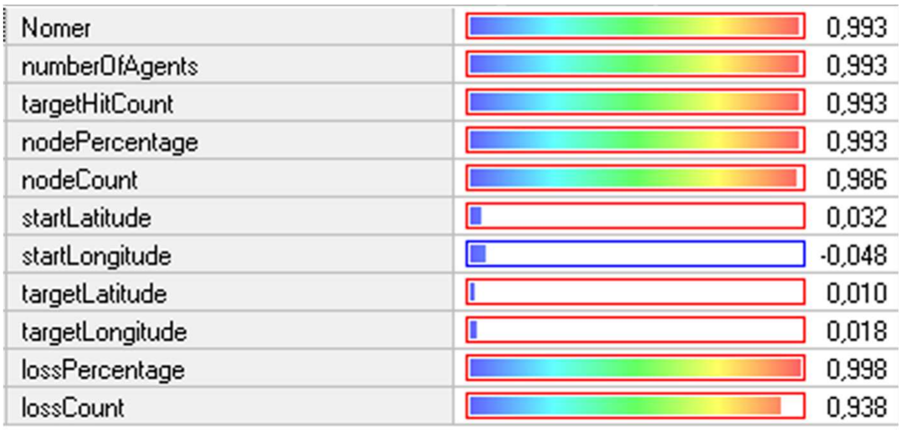


Figure 5: Correlation Analysis of Input Data

The obtained dataset was used to train a multilayer perceptron neural network. The following network structure was established: five layers of neurons, with the first (input) layer containing 7 neurons, three hidden layers containing 7, 3, and 5 neurons respectively, and the fifth (output) layer containing 1 neuron. The activation function used is sigmoid. The architecture of the network – defined by the number of layers, their sizes, and the activation function – determines its ability to solve specific tasks. This framework demonstrates how data moves through the network, from the input to the output layer, with each layer performing calculations using weights and the activation function. The backpropagation method was employed for training.

The schematic representation of the neural network structure is shown in Figure 6, where the line colors indicate the values of the weighting factors. The output of the neural network is the probability of a target being hit (%). By applying the parameters of a different task to the input, the network generates a predictive output for the probability of hitting the target (%).

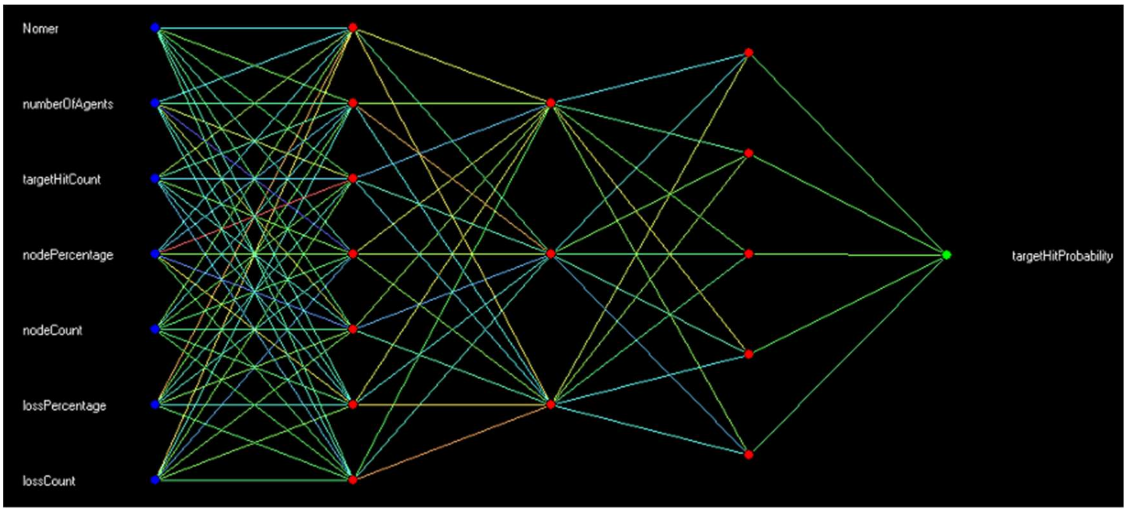


Figure 6: The structure of a neural network

During the training of the neural network, the probability values of hitting the target (%) were obtained. On average, the deviation across 1,000 experiments was 3.8%. This suggests that the neural network trained using this methodology yields results that closely match the efficiency of the mobile agent control system (Table 4).

Table 4
Comparison of the results of the neural network with the original data

numberOfAgents	73	99	87	44	66	89	79	50	91
targetHitPercentage	5	5	5	5	5	5	5	5	5
targetHitCount	4	5	4	2	3	4	4	2	5
nodePercentage	10	11	4	5	15	6	19	15	15
nodeCount	7	11	3	2	10	5	15	8	14
lossPercentage	63	51	69	69	90	63	86	63	73
lossCount	46	50	60	30	59	56	68	31	66
targetHitProbability	74	88	70	84	55	16	57	17	100
Neural network data	77	86	68	83	58	15	60	16	99
Error, %	4.05	2.27	2.86	1.19	5.45	6.25	5.26	5.88	1.00

The structure of the neural network selected for training resulted in the smallest relative calculation error, while errors for other network configurations exceeded 5%. Therefore, the constructed neural network can be effectively used to solve similar problems, as it accurately reflects the results and allows for the estimation of the probability of hitting the target (%) in dynamically changing environments. Additionally, the use of the neural network helped identify input parameters with minimal impact on the outcome.

5. Conclusions

This study reviews the network-centric approach as a method for enhancing the efficiency of mobile agent control systems. The key distinction of this approach, compared to others, is the use of a "wandering center." If communication with the current center is lost, a new center is appointed, ensuring the continuity of the control process. This structure guarantees the system's survivability and, as demonstrated, offers significant potential for improvement, ultimately increasing the overall efficiency of mobile agent control.

The research highlights the effectiveness of network-centric control when compared to centralized control (in which no nodal mobile agents are used). It was found that as the number of nodal mobile agents increases, the effectiveness of hitting the target improves. Notably, effective performance (80% or higher) is achieved with just 9% of the total number of mobile agents functioning as nodal agents, suggesting that exceeding this threshold would not provide additional benefits.

Finally, the results from the mobile agent control system were compared with the output from the neural network training, which estimates the probability of hitting the target with a relative deviation of 3.8%. This confirms that the neural network, trained using the proposed methodology, produces results that closely align with the efficiency of the mobile agent control system.

References

[1] Moutinho A. Modeling and nonlinear control for airship autonomous flight. Instituto Superior Tecnico, Technical University of Lisbon. 2017. P. 55–65.

[2] Fukau T., Kanzawa T., Osuka K. Inverse optimal tracking control of an aerial blimp robot. Proceedings of the 5th International Workshop. Robot Motion and Control. 2015. P. 193–198.

[3] Ko J., Klein D., Fox D., Haehne D. Gaussian Processes and Reinforcement Learning for Identification and Control of an Autonomous Blimp Proc. of the IEEE Int. Conf. Robotics and Automation. 2017. P. 742–747.

[4] V. Zavgorodnii, N. Braykovs'ka, O. Yarovyi, A. Zavgorodnya, V. Liskin, O. Mukhin. The Method of Restoring Parameters of Mobile Agents in a Unified Dynamic Environment Considering Similarity Coefficients. International Journal of Computer Network and Information Security (IJCNIS). Vol.15. No. 4. R. 25-35. 2023. DOI: https://doi.org/10.5815/ijcnis.2023.04.03

[5] Kwon J., Kim J., Seo J. Vector field guided auto-landing control of airship with wind disturbance. Proceedings of the 19th World Congress, Cape Town, South Africa. 2014. P. 1114–1119.

[6] A. Dodonov, V. Mukhin, V. Zavgorodnii, Y. Kornaga, A. Zavgorodnya, O. Mukhin. Method of Parallel Information Object Search in Unified Information Spaces. International Journal of Computer Network and Information Security (IJCNIS). Vol. 13. No. 4. P.1-13, 2021. DOI: https://doi.org/10.5815/ijcnis.2021.04.01

[7] Cortes, J., Martinez, S., Karatas, T., Bullo, F. (2002) Coverage control for mobile sensing networks. IEEE Conference on Robotics and Automation. Arlington, VA, pp. 1327-1332. DOI:https://doi.org/10.48550/arXiv.math/0212212

[8] Dodonov, A., Mukhin, V., Zavgorodnii, V., Kornaga, Ya., Zavgorodnya A. (2021). Method of searching for information objects in unified information space. System research and information technologies. N1. P. 34–46. DOI:https://doi.org/10.20535/SRIT.2308-8893.2021.1.03

[9] Golembo, VA, Bochkaryev, O.Yu., Tsyzh, AM (2006) The task of forming individual zones of responsibility by a team of mobile agents. Visn. National Lviv Polytechnic University. Vol. 573. P. 62-67.

[10] Mukhin, V., Kornaga, Y., Bazaliy, M., Zavgorodnii, V., Krysak, I., Mukhin, O. (2020) Obfuscation Code Techniques Based on Neural Networks Mechanism. IEEE 2nd International Conference on System Analysis & Intelligent Computing (SAIC). Kyiv. Ukraine. P. 1–6. DOI:https://doi.org/10.1109/SAIC51296.2020.9239247

[11] Chikriy A.A., Baranovskaya L.V., Chikriy A.A. (2000) An approach game problem under the failure of controlling devices. J. of Automation and Information Sciences. Vol. 32, No. 5. P. 1–8. DOI:https://doi.org/10.1615/JAutomatInfScien.v32.i5.10

[12] Baranovska L.V. Quasi-linear differential-difference game of approach. Understanding Complex Systems. 2019. P. 505–524. DOI:https://doi.org/10.1007/978-3-319-96755-4_26.

[13] El-Zahar E. Applications of Adaptive multi-step differential transform method to singular perturbation problems arising in science and engineering. Appl. Math. Inf. Sci. 2015. Vol. 9, no. 1. P. 223–232.

[14] Gusynin V., Gusynin A., Tachinina H. The use of differential transformations for solving non-linear boundary value problems. Proceedings of NAU. 2016. No. 4 (69). P. 45–55.

[15] Mukhin, V., Zavgorodnii, V., Kornaga, Y., Baranovska, L. (2021). Algorithm for the Information Space Forming and the Evaluation of Input Objects Search Efficiency. CEUR Workshop Proceedings this link is disabled. 3241. P. 193–204. URL:https://ceur-ws.org/Vol-3241/paper18.pdf

[16] Liu Feng, Guo Wei-Wei. Research and Design of Task Scheduling Method Based on Grid Computing. International Conference on Smart City and Systems Engineering (ICSCSGridE). Changsha, China. 2017. P. 188–192.

[17] Sheikh S., Shahid M., Nagaraju A. A novel dynamic task scheduling strategy for computational grid. International Conference on Intelligent Communication and Computational Techniques (ICCT). 2017. P. 102–107.

[18] Mukhin, V., Zavgorodnii, V., Kornaga, Y., Zavgorodnya, A., Krylov, I., Rybalochka, A., Kornaga, V., Belous, R. (2021). Devising a method to identify an incoming object based on the combination of unified information spaces. Eastern-European Journal of Enterprise Technologies. 3(2 (111). P. 35–44. DOI:https://doi.org/10.15587/1729-4061.2021.229568

[19] Aron R., Chana I. Grid Scheduling Heuristic Methods: State of the Art. International Journal of Computer Information Systems and Industrial Control Applications. 2014. Vol. 6, P. 466–473.

[20] Chauhan P. Decentralized Scheduling Algorithm for DAG Based Tasks on P2P Grid. Journal of Engineering. 2014. P. 1–14.

[21] Canfora G. Formal Tool for Identifying Mobile Malicious Behavior. IEEE Transactions on Software Engineering. 2019. Vol. 45. P. 1230–1252.
[22] How J., Frazolli E., Chowdhary G. Linear Flight Control Techniques. Handbook of Unmanned Aerial Vehicles. Dordrecht; Heidelberg; New York; London: Springer. 2018. P. 529–576.
[23] Lotfi A. Zadeh. 1994. Fuzzy logic, neural networks, and soft computing. Commun. ACM 37, 3 (March 1994), 77–84. DOI:https://doi.org/10.1145/175247.175255
[24] Varley TF, Sporns O, Schaffelhofer S, Scherberger H, Dann B. Information-processing dynamics in neural networks of macaque cerebral cortex reflect cognitive state and behavior. Proc Natl Acad Sci US A. 2023 Jan 10; 120(2): e2207677120. DOI:https://doi.org/10.1073/pnas.2207677120
[25] Paulo Vitor de Campos Souza. Fuzzy neural networks and neuro-fuzzy networks: A review of the main techniques and applications used in the literature. Applied Soft Computing. Vol. 92. 2020. ISSN 1568-4946. DOI:https://doi.org/10.1016/j.asoc.2020.106275
[26] AM Zador. A critique of pure learning and what artificial neural networks can learn from animal brains. Nat Commun 10, 3770 (2019).https://doi.org/10.1038/s41467-019-11786-6
[27] V. Mukhin, V. Zavgorodnii, V. Liskin, S. Syrota, V. Koval, L. Honchar. Classification of Information Objects with Fuzzy Parameters in E-Learning Systems. IEEE 12th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS). Dortmund. Germany. 2023. P. 1189-1193. DOI: https://10.1109/IDAACS58523.2023.10348768
[28] Ghafor K. Multifunctional Models, Including an Artificial Neural Network, to Predict the Compressive Strength of Self-Compacting Concrete. Applied Sciences. 2022; 12(16):8161. DOI:https://doi.org/10.3390/app12168161
[29] Vadym Mukhin; Yaroslav Kornaga; Yurii Bazaka; Ievgen Krylov; Andrii Barabash; Alla Yakovleva; Oleg Mukhin "The Testing Mechanism for Software and Services Based on Mike Cohn's Testing Pyramid Modification," 2021 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Cracow, Poland, 2021, pp. 589-595, doi: 10.1109/IDAACS53288.2021.9660999.