# Software Platform for Designing and Running Artificial Intelligence Competitions with a Visualization Subsystem

D.A. Korostelyov[1], A.O. Radchenko[1], N.S. Silchenko[1], R.A. Krylov[1], P.N. Migal[1]
nigm85@mail.ru|jameslistener@gmail.com|silchenko.nk@gmail.com|oktopy@gmail.com|p.migal@yandex.ru
[1]Bryansk State Technical University, Bryansk, Russia

*The paper describes the solution to the problem of testing the efficiency of new ideas and algorithms for intelligent systems. Simulation of interaction of the corresponding intelligent agents in a competitive form implementing different algorithms is proposed to use as the main approach to the solution. To support this simulation, a specialized software platform is used. The paper describes the platform developed for running competitions in artificial intelligence and its subsystems: a server, a client and visualization. Operational testing of the developed system is also described which helps to evaluate the efficiency of various algorithms of artificial intelligence in relation to the simulation like "Naval Battle".*

***Keywords:*** *artificial intelligence, intellectual agent, artificial intelligence competition.*

## 1. Introduction

At present artificial intelligence technologies are increasingly penetrating into various spheres of human activity. To apply and develop the technologies and methods of artificial intelligence effectively, it is necessary to take into account these trends in the educational process for training modern professionals, as well as in professional activities in various fields: information technologies, economics, finances, design of engineering and control systems, etc.

To form corresponding competences both classical methods of training (study of relevant algorithms), and more progressive approaches based on game or competitive principles are used [1]. For implementing the competitive (game) approach a specialized software platform is usually designed or used, which is a system of multiagent simulation. Each participant loads into this system his algorithm represented as a program that implements a special interface to interact with the system. This program in the system represents one or more intelligent agents interacting with each other as a result of simulation (or several simulations). The rules of interaction and capabilities of intelligent agents are determined by scenarios and the corresponding software interface of the system (platform), and the process of development and simulation becomes competitive.

One of the key components of such platforms is a visualization subsystem of simulations of intelligent agents' interaction, which contributes to a better understanding of the implemented algorithms and results in improving and developing them more effectively. By means of the visualization subsystem it is possible to demonstrate the work of both artificial intelligence algorithms and various data processing algorithms in a more visual and comprehensive form. This aspect has a positive impact on students' understanding of the principles of relevant algorithms.

Such competitions, in addition to training and developing skills in the use and implementation of artificial intelligence methods, also allow to find the most talented among the participants [2]. Usually, to participate in the competition not many skills are necessary, but to win you need to use unique algorithms, universal strategies and much more.

At the moment, there are a considerable number of different platforms for competitions in artificial intelligence [2-5]. However, such platforms are often developed for one specific competition, and after this completion either become open in the form of a sandbox, or are completely forgotten by both developers and participants. Among the most famous platforms are Ants AI Challenge [3], Russian Ai Cup [4], Mini AI Cup [5].

Ants AI Challenge is a Google product [3]. It was a competition of intelligent agents, in which users were given API to interact with the system (Fig. 1). The user developed artificial intelligence, which based on the tasks of the competition, performed certain actions aimed at winning in the most effective way. Among the main advantages of the platform are the following:

1. A unique set of competition rules is implemented.
2. The interactive visualization subsystem displays complete statistics during simulation.
3. The interactive visualization subsystem has an intuitive display of the competition elements represented by a matrix.
4. The visual interactive player uses primitives for display, so its load on the system is minimal.

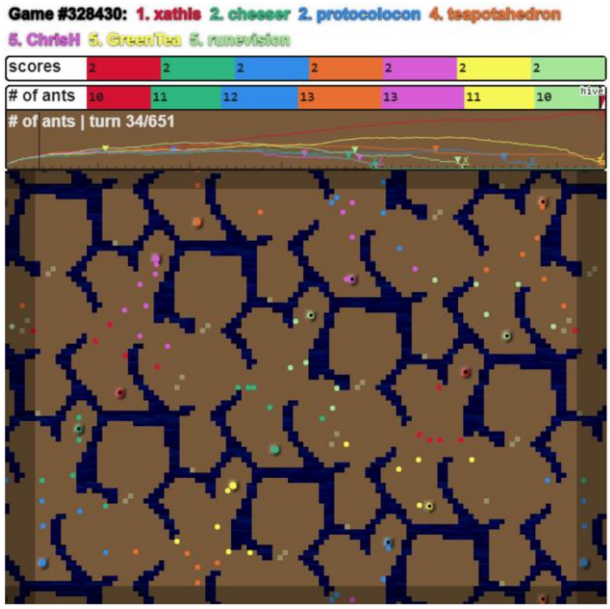


**Fig. 1. Interface of Ants AI Challenge visual player**

As for the disadvantages, they are the following:

1. There is no opportunity to participate in the competition, because it has already been held and completed.
2. The visual interactive player does not have control components.

Russian AI Cup is a competition of artificial intelligence and intelligent agents who fight in the virtual world instead of their owners (Fig. 2) [4]. At all the previous championships more than 40,000 users have registered on the platform, and they have developed more than 150,000 different solutions in total. Each new championship is a new challenge with its own rules, laws and mechanic.

The main advantages of the platform are the following:

1. A variety of competitions and problems of simulation the interaction of intelligent agents.

2. The competitions take place in both two-dimensional and three-dimensional world.

3. There is online access to the visual interactive player of simulations.

**Fig. 2. Interface of Russian AI Cup visual player**

There are some disadvantages as well. They are:

1. For each competition, a new visual display module is used, independent of the previous ones.

2. There is no most part of the statistics. One can view only the main points of the simulation.

3. The visual player has no control components.

4. There is no possibility of step-by-step simulation playback.

Mini AI Cup is a product of Mail.Ru Group, which is an opportunity to prepare for a larger Russian AI Cup competition (Fig. 3). The conditions for competition seem to be simpler [5]. The platform is a set of mini-contests related to artificial intelligence and writing intelligent agents.


**Fig. 3. Interface of Mini AI Cup visual player**

Among the main advantages of the platform are the following:

1. A variety of competitions and problems of simulation the interaction of intelligent agents.

2. There is a control system to play back simulations.

3. Detailed displaying of statistics.

The disadvantages are the following:

1. For each competition, a new visual display module is used, independent of the previous ones.

2. There is no possibility of step-by-step simulation playback.

As one can see from the description above, the existing platforms for artificial intelligence competitions are not invariant to the problems for which it is possible to develop intelligent agents. Also in the existing systems there is a lack of flexibility of the visualization system, which could be quite an effective tool for visual evaluation of algorithms implemented in the form of intelligent agents, provided that it is possible to adapt them to a specific simulation problem. Therefore, the development of a universal platform that provides competitions in artificial intelligence for different scripts and tasks with different visualization methods is an urgent problem.

## 2. Description of building a platform for simulating the interaction of intelligent agents

Artificial intelligence competitions should be as honest, open and universal as possible. To ensure this, server-based simulating of intelligent agents' interaction is used. This guarantees to avoid scenario substitution or other violations of the tournament. The server allows intelligent agents developed by specialists located far from each other to compete. Also, with the help of the server, an unlimited number of tournaments for different groups of participants with unique scenarios can be held. In order to run the tournament according to any scenario developed by the organizer, without additional interventions into the system, as well as to debug the implementation of the algorithms in the form of intelligent agents, a system of simulating the interaction of intelligent agents is necessary. This feature allows to organize a large number of tournaments with a small amount of time for preparing.

In order the participation in the competition should be more visual, and the peculiarities of the algorithms – intuitive, a visualization subsystem of the competition is necessary. Thus, the reasons for agents' winning or losing will be clear not only for the authors of the agents themselves, but also for observers of the competition, who will also be able to view the visual results of the interaction of intelligent agents and evaluate the efficiency of certain implementations of behavioral algorithms. Also, the visualization subsystem allows to track the errors of algorithms and their illogical actions from the point of view of a particular simulation, that improves the efficiency of algorithms determining the behavior of intelligent agents.

It also should be noted that holding competitions on insecure devices is not recommended, so that to avoid the possibility to break the logic of the competition and to win unfairly. Having other users' intelligent agents on one device is also not acceptable, because they can be used to update their intelligent agents.

## 3. Description of software platform

The software platform for designing and holding artificial intelligence competitions with visualization subsystem was implemented in C++ using Visual Studio environment.

The main feature of the platform is interaction with a large number of people (both participants and organizers). For more convenient interaction, client-server architecture was used, which is based on the fact that users have an application by means of which they can interact with the organizers (Fig. 4).

The server subsystem allows to design tournaments for any scenarios and agents developed in accordance with the requirements of the platform. It receives requests from the administrator or from the client application and processes them according to the command sent. The administrator can also receive requests from the organizer for certain actions. The basic commands are to design a tournament, enter the agent or scenario into the database, to register or authorize the user, connect the user to the tournament and run the tournament.

The server subsystem is directly connected with the subsystem of the tournament by artificial intelligence. Referring to it, the server runs the tournament and receives the results that can be provided to administrators, organizers and participants.

The results are obtained on the server, so the organizer himself must choose how to present the results to the participants.

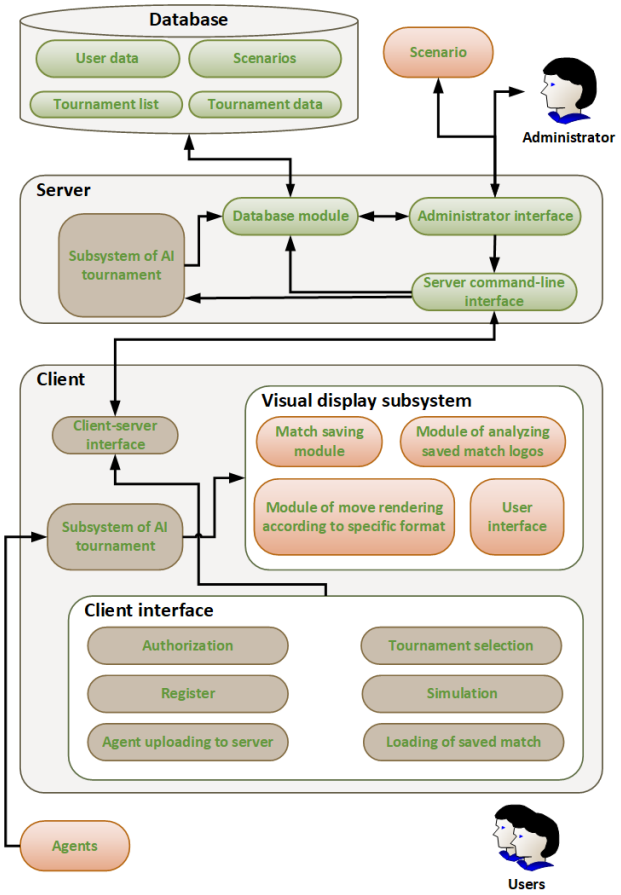


**Fig. 4. Software platform architecture**

The client subsystem allows users to interact with the server using a convenient graphical interface when participating in the tournament. The client subsystem of intelligent agents' interaction is located on the client, and the tournament module is located both on the client and on the server.

Work with the artificial intelligence of running tournament can be initiated in one of two ways: the user tests the agent before sending or the administrator begins to calculate the results of the competition. The subsystem of the competition is located on the client and on the server, which allows to test intelligent agents for users quickly, and to calculate the competition on the server easily.

The subsystem of visual display allows script writers to use their materials to display the actions of intelligent agents, using both primitives and textures (Fig. 5). The visual display subsystem is located on the client. It receives data from the server (scripts, test intelligent agents) or from the organizers (textures, competition rules), and then using the script it generates the simulation and displays it. The generated simulation can be saved and then loaded for further watching and analysis.

There are two ways to start working with the player. The first is to load the saved simulation. The second is agent testing. In the case of agent testing, the subsystem of the tournament is initialized and the competition is held. The tournament subsystem interacts with the scenario, which in its turn interacts with the agent. We get an action log as a result, which is then used by the player for visual display. If we load the saved action log, the simulation is played at the player's request by frames (moves), and the player displays the data on the screen.

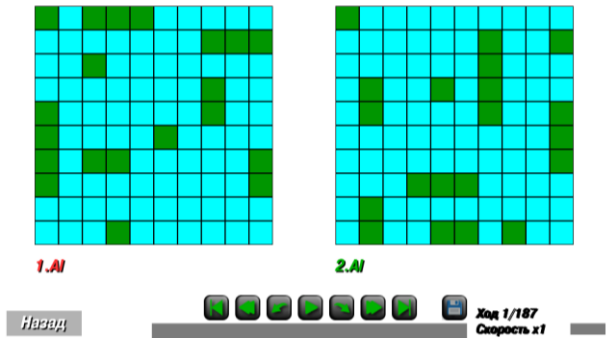


**Fig. 5. Player interface by the example of "Naval Battle" simulation**

To demonstrate the capabilities of subsystems, examples of scenarios and intelligent agents for them were developed, including algorithms visualizing the operation (Fig. 6).


**Fig. 6. Visualization of sorting algorithm operation**

Also, with the help of the developed platform, an experimental study was carried out to evaluate the efficiency of various algorithms of intelligent agents' behavior by the example of "Naval Battle" scenario.

## 4. Description of intelligent agents algorithms under study

The experimental study of the efficiency of artificial intelligence algorithms [6] was carried out on the basis of 3 main options for implementing algorithms of intelligent agent behavior for "Naval Battle" scenario and some of their modifications.

1. Algorithm based on random events for the scenario.
2. Algorithm based on the field analytics.
3. Algorithm based on the analytics of ship positions.

### 4.1. Algorithm based on random events for the scenario

The algorithm of the simplest intelligent agent based on random events performs the following actions:

1. Selection of random coordinates to move.
2. Waiting for the opponent's move.
3. Checking if the simulation is not finished, then go to step 1.

This agent makes an extremely large number of moves due to the lack of checkings, as it can shoot at the same point, or make moves where there is no enemy knowingly (for example, next to the shot ships).

If coordinate validation is additionally made, the intelligent agent will be able to make a lot fewer moves. This is due to the fact that now he does not choose the same coordinates and analyzes his move in accordance with the rules of the game (if the enemy is dead, there isn't definitely any other ship in a radius of one cell; if the enemy is wounded, then there is exactly a continuation of the ship around). Let us consider 3 modifications for this variant of the intelligent agent.

1. Modification 1 is dividing the field. The field is divided into 4 areas of 5x5 cells. Then moves are made in them by turns. In the case of engaging the target, the next move is made in the same field.

2. Modification 2 is random moves at the beginning. It implies ignoring damaged ships in the first 10 moves.

The essence of this modification is that in the first few moves we can find several damaged ships. In some cases, this will allow us to determine in which direction the ship is directed at once, and then finish shooting it. Value 10 for the threshold of the initial moves is chosen based on the fact that the average party lasts 50 moves.

When using this algorithm, during the first 10 moves, if a wounded opponent is found, the rule is applied to him as to the killed: the moves are not made within a radius of one cell from it. Thus, by the end of the tenth move, up to 90% of the playing area can be analyzed.

This algorithm is most effective if the opponent places his ships remotely from each other.

3. Modification 3 is greedy match. If there is a hit, then not adjusting cells are checked firstly, but the cells next but one.

This modification is similar to the second, but has some differences: if the second modification tries to view as much space on the field, the third does it more consistently and throughout the whole game.

The use of modification 3 saves moves on damaging three- and four-deck ships.

### 4.2. Algorithm based on the field analytics

The principle the algorithm operating on the basis of the field analysis [7] is as follows: before the move, the agent checks the presence of damaged ships on the field and acts accordingly to destroy them. If there are no more damaged areas, the algorithm makes predetermined moves. The basic version of the algorithm goes through the field from left to right, from top to bottom.

Let us consider 3 modifications for this variant of the intelligent agent.

1. Modification 1 is diagonal analytics. On average this analysis makes it more likely to find ships from larger to smaller. The essence of the algorithm is that initially the parallels to the diagonal line with an indentation between them equal to the size of the largest ship are checked sequentially. If the largest ship has already been destroyed, reduce the indentation to the currently largest ship.

This algorithm has a great dependence of efficiency on the angle in which the opponent has ships, and from what angle he starts. If ships are evenly spaced, the dependence disappears.

2. Modification 2 is staggered order. It is the most effective if the opponent has placed his single-deck ships on one of the colors of the checker.

3. Modification 3 is the method of "dogtooth". This modification makes the combinations of moves on the field, shown in Fig. 7, so covering the whole field.

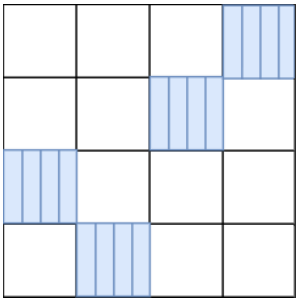

**Fig. 7. "Dogtooth" sequence of moves**

This modification can significantly speed up finding of ships if they are located in the direction of the center. This modification of the algorithm copes with random arrangements in fewer moves, as it does not analyze potentially useless information, unlike the other two modifications.

### 4.3. Algorithm based on the analytics of ship positions

The principle of the algorithm is as follows: before the move, the agent receives a set of data about the field and counts the remaining ships, their quality and quantity [8]. Further, based on this analysis, the agent chooses his move. The basic agent performs the analysis but does not use it. This analysis allows to find large enemy ships more likely (three - and four-deck) and as quickly as possible to get rid of them, opening the space around the destroyed ships.

Let us consider 3 modifications for this variant of the intelligent agent.

1. Modification 1 is looking for neighbors. This modification allows to look for neighbors effectively. Firstly, having found a damaged ship, next but one cells from the ship are checked. In the best case, you can find neighbors or the end of the three-deck ship. In case if there are no two-, three - and four-deck ships left, restrictions are removed. If there are no more places that meet the requirement, the agent tries to finish shooting them, "connecting" neighboring damaged cells. If there are still ships in the field, neighboring cells are checked.

This algorithm works very well with space strategies that try to leave the maximum amount of free space in order to place the ships as close as possible to each other.

2. Modification 2 is "greedy" finishing. This agent calculates whether there are two-, three - or four-deck ships left in the field. If left, the next move is made with the indentation of two, one or zero cells, depending on the remaining ships. The agent tries to finish shooting the ship immediately, starting from the end.

It is the most effective if the opponent puts his ships close to each other. It also makes it easy to finish shooting large ships.

3. Modification 3 is limiting the field depending on the remaining ships. This modification ignores the part of the field depending on the remaining ships. If there is one four-deck ship left, the moves are made with a margin of 3 cells from the edge. Thus, 6x6 field is analyzed first of all. The restriction is ignored when trying to finish the ship. If a four-deck ship is found and destroyed, the indentation is changed to 2 cells. If all three-deck ships are found, indentation is changed to one cell. If there are no more double-deck ships or no cells satisfying the condition, the restrictions are removed completely.

Using this modification can significantly speed up finding ships if they are located in the direction of the center.

## 5. Experiment results

A series of experiments in the form of simulation of intelligent agents' interaction was carried out with each of the algorithms. According to the results of the experiments, statistical data characterizing these algorithms were obtained. The average values of these characteristics are presented in table. 1.

The fastest option in terms of the number of moves was modification 3 of the algorithm, based on the field analytics. This is due to the fact that it has the most versatile analytics that allows to reduce the average number of moves to search.

It should be noted that the time to execute a single move in an analytics-based algorithm without modification is extremely fast. This is due to the fact that the algorithm does not need to calculate any action, and it is enough to focus only on checking the rules of the game and reading information about the next, predetermined move.

| № | Algorithm | Number of moves | Average time (ms) |
|---|---|---|---|
| 1. | Random events. No modifications. | 52.1 | 23.7 |
| 2. | Random events. Modification 1. | 49.6 | 24.6 |
| 3. | Random events. Modification 2. | 47 | 24.3 |
| 4. | Random events. Modification 3. | 55.2 | 27.1 |
| 5. | Field analytics. No modifications. | 77.9 | 14.8 |
| 6. | Field analytics. Modification 1. | 50.7 | 21 |
| 7. | Field analytics. Modification 2. | 51.7 | 21.1 |
| 8. | Field analytics. Modification 3. | 43.3 | 21.1 |
| 9. | Analytics of ship positions. No modifications. | 53.2 | 27.9 |
| 10. | Analytics of ship positions. Modification 1. | 38.6 | 25.5 |
| 11. | Analytics of ship positions. Modification 2. | 51.9 | 28.4 |
| 12. | Analytics of ship positions. Modification 3. | 59.3 | 45.8 |

**Table 1.** Comparison of intelligent agents implementing various algorithms of "Naval Battle" scenario.

## 6. Conclusion

The developed system allows to design scenarios with different conditions, as well as to set their own unique display and run an unlimited number of tournaments according to these scenarios. It can also be used as a learning system due to special scenario tasks in which the user needs to implement the algorithm and send the result to the server for verification.

In addition to the considered scenario of "Naval Battle" type, the system was tested on scenarios such as "War of Viruses", "Snakes", as well as on educational tasks of matrix diagonal display and array sorting in order to demonstrate the work of the corresponding algorithms visually. Templates are also available for the developer to create their own scripts and test agents to them with the necessary recommendations.

The distinctive features of the developed system in comparison with other platforms are its versatility and invariant property, both in the supported scenarios and in the methods of visualization of intelligent agents' interaction. Also, the flexible system of visual display of scenarios allows to develop more effectively and debug algorithms in various fields of artificial intelligence.

The system developed was used for artificial intelligence competition in Bryansk regional IT-festival, held under the guidance of the 29th international conference on computer graphics and machine vision GraphiCon 2019.

## 7. References


[1] Chesani F., Galassi A., Mello P., Trisolini G. (2017) A Game-Based Competition as Instrument for Teaching Artificial Intelligence. In: Esposito F., Basili R., Ferilli S., Lisi F. (eds) AI*IA 2017 Advances in Artificial Intelligence. – AI*IA 2017. Lecture Notes in Computer Science, vol 10640. – Springer, Cham.

[2] F. Lu, K. Yamamoto, L. H. Nomura, S. Mizuno, Y. Lee and R. Thawonmas. Fighting game artificial intelligence competition platform. – 2013 IEEE 2nd Global Conference on Consumer Electronics (GCCE), Tokyo, 2013, pp. 320-323.

[3] Ants AI Challenge. – URL: http://ants.aichallenge.org.

[4] Russian AI Cup – artificial intelligence programming contest. – URL: http://russianaicup.ru.

[5] Ai Cup – artificial intelligence programming contest. – URL: https://aicups.ru/.

[6] Tim Jones, M. AI Application Programming. M.: DMK Press, 2018. – 312 p.

[7] Norwing, P. Artificial Intelligence: A Modern Approach. – UK: Glivice, 2014. – 1408 p.

[8] Riley, D. Abstraction and data structures. Introductory course. – Cambridge, MA; London, UK: MIT Press, 2018. – 310 p.