UDC 004.415.2
DOI https://doi.org/10.32838/2663-5941/2021.5/21

Oleshchenko L.M.
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"

Medvedev M.G.
V.I. Vernadsky Taurida National University

Kobryn D.R.
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"

Sukalo M.L.
National University of Food Technologies

AGENT MODELLING SOFTWARE OF POPULATION BEHAVIOR IN EMERGENCY SITUATIONS

The ability to predict the movement of people is very valuable in many contexts. Panic situations are likely to have been the biggest influence on the development of solutions to model crowd behavior. The paper considers research on the movement of crowds of people in emergencies and analyzes the existing frameworks and software for agent modeling of the crowd. Based on the research on the behavior of people in the crowd and available approaches to modeling the movement of people and the crowd in an emergency, the software requirements were formulated: the ability to configure each of the agents; the ability to change the geometry of the premises; parameterization of the model; correct operation of algorithms with a large number of agents; taking into account the field of view of agents; realization of observance of personal space of the person; implementation of an algorithm to change the trajectory of motion in a collision; realization of the phenomena of crowd panic; calculation of crowd density; the possibility of random distribution of agents; possibility of manual distribution of agents; import and export the result of the program analysis in JSON format. The general description of architecture and the developed software modules of the created software is presented. The created modules include an environment module, an agent module, an agent generation module, a scene editing module, a database interaction module. Describes the architecture of software developed on the Unity platform and how all components interact with each other. A Pathfinding Project technologies were used to develop the software to provide flexible and reliable path finding, the Django framework to create a storage server API, an additional djongo library to integrate with MongoDB database.*

Key words: *software, emergency, agent modelling, Python, Django framework, Unity, MongoDB.*

Introduction. Problem statement. A crowd is a large group of disorganized people that can be described as a temporary gathering of a large number of people who respond almost equally to a particular stimulus. In people's lives, especially in megacities and big cities, there is always a place of movement in large crowds. It can be either a daily morning queue in the subway or an idol concert attended by tens of thousands of people. Thousands of mass events take place every year around the world, during which many people are injured due to the danger of the crowd. This is caused by a certain emergency that causes panic in the crowd. Sometimes the emergency itself poses a much lesser threat than the unpredictable behavior of the crowd. Such situations are influenced by a number of factors such as the geometry of the premises, the number of exits, as well as the cause of the panic, the reaction of the crowd and the effectiveness of the evacuation. The panic, the crowd in general in certain situations is much more dangerous than the emergency that caused it. Emergencies are difficult to prevent, it is possible to change external factors and try to minimize the consequences of irrational human behavior. Everyone in the crowd has own behavior, which is extremely difficult to convey through a model. The complexity of the behavior of the human masses is associated with the presence of behavioral patterns: clustering, queuing, setting routes that people use almost every day on a subconscious level. In order to experiment with different factors of the system, improve the quality of predicting the consequences of an emergency and identify deficiencies in crowded places, a flexible software tool is needed to model the behavior of the crowd with the ability to simulate different types of agents and their environment.

Related research. There is a wide range of programs for simulating the movement of human masses such as: Menge framework, Pedestrian Dynamics, Crowd Dynamics, DI-Guy. After analyzing these products and the projects in which they are used, we can conclude that the creation of systems for predicting the behavior of crowds and individual crowds is of interest to both government agencies and private companies.

The main goal of the article is to create software that allows to model and predict the behavior of the crowd in an emergency.

An overview of existing software products

In recent years, the simulation of crowds and groups of people has become increasingly important not only in the field of design. The products created on the market vary from systems for modeling the flow of people in emergencies and logistics solutions in the design of architecture. To studying people's behavioral patterns, one of the goals of such research was to develop animated realistic programs for urban design, planning, and the game industry. In a familiar environment, each person shows some common behavioral attributes. People always try to find the shortest way to the goal, if it is possible to avoid a change of direction, they will go through the crowd, showing the basic principle of "least effort" where everyone tries to save time and energy. Most habitual and rational behavioral patterns disappear as soon as people are exposed to an emergency or other stimulus or stimulus. In such emergencies, the crowd shows completely different patterns. People who try to get out of a room as fast as possible, for example, start moving much faster, trying to overtake others, and this is why the characteristic formations around bottlenecks appear. These clusters do not allow people to move freely and, thus, the speed of the flow of people in the crowd becomes much slower.

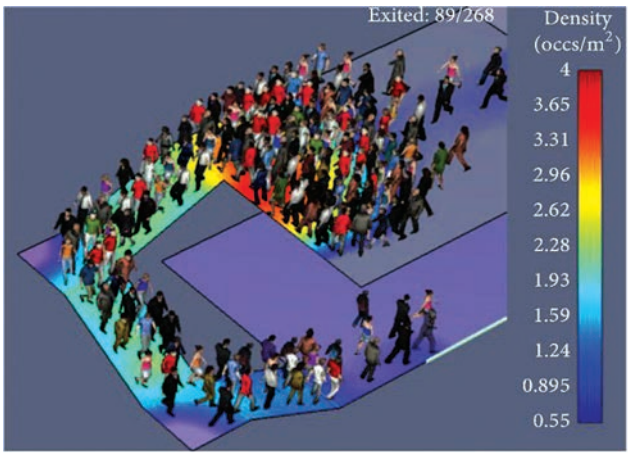

Fig. 1. Characteristic arched formations around the entrance to the stairs [1]

People who entered the building through a certain entrance and do not know its structure, in a panic will try to first reach the same entrance, rather than looking for the nearest emergency exit, which can be closer and safer [1]. In an emergency, people who easily panic show the phenomenon of herd behavior, during which they recklessly try to join the largest group of people.

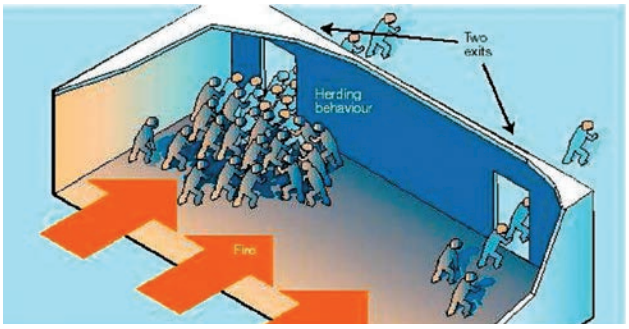

Fig. 2. Demonstration of herd behavior [1]

Such behavioral phenomena in panic usually cause the most injuries and casualties in crowds caused by an emergency. As the nervousness of the crowd increases, each individual pays less attention to other people's comfort zones and tries to achieve his goal, and this sometimes results in violent behavior of the crowd, during which people begin to push, grab others, despite people falling, trampling their. In the process of reviewing the existing available models, it was determined that the situations of modeling emergencies and their consequences are very closely related to the specific state of individual agents. The real crowd is an extremely complex entity in terms of predicting behavior, because every second each person in the crowd makes many decisions, which at the same time are influenced by almost everyone in the crowd, as well as physiological, psychological and social factors of each entity. Mathematical approaches and analytical models are not able to predict the behavior of the crowd with sufficient reliability. If we consider scientific research in the field of pedestrian dynamics, then all the created models are subject to two scales of simulation: microscopic and macroscopic. Microscopic models consider the position of each element of the crowd as a separate discrete part of the simulation. Macroscopic models, on the other hand, perform a simulation as an averaged distributed representation of the entire crowd and consider the crowd in space using variables such as density, flow, and front. Models are usually created on the basis of some phenomenological assumptions, therefore, they are likely to reproduce a fairly similar behavior of crowds with the number of N elements.

Among all the analyzed approaches to solving the problem of crowd modeling, the following main approaches were identified.

The approach of cellular automata. In this approach, the space allocated for behavior modeling is a set of cells that create a lattice with certain rules of transition from state to state. The room is considered as a field of the cellular automaton. Thus, each cell is essentially a finite automaton, the state of which depends on the state of its neighboring cells. For example, a cell with a "busy" state contains a human and the simulation is performed by changing and analyzing the states of a plurality of cells. On the one hand, this method of modeling gives a very fast to calculate $O(N)$, but at the same time simplifies and averages the forces of interaction, which affects the realism of the model.

Approach based on Newtonian mechanics. In this approach, each effect of the environment on the object is reflected by a certain force that tries to change the position of the simulation element. The motion is thus described by Newton's second law. The advantage of this approach is that each element can affect all other elements, thus allowing unlimited complication of the interaction process by adding new vector forces. One of the main disadvantages of such models is that the time complexity of such an approach is $O(N^2)$, which makes it very inefficient with great detail of the process.

Approach based on physical processes of liquids and gases. In such approaches, each element of the crowd is a particle whose state is described by a certain equation of a liquid or gas. Such modeling focuses on the crowd as a whole, not on its components. Models of this type are mainly used in the simulation of dense and large crowds in a short period of time. Thus, the personal qualities of the members of the crowd have almost no effect on the environment, reducing behavioral factors and allow the study of the crowd as a physical process.

An approach based on multiagent methods. This approach describes simple rules of motion and interaction of elements that determine the behavior of each of the simulation objects. Each of the simulation objects responds to any situation independently, based on the described rules for decision making. Because almost every type of behavior can be implemented in different agents, increasing realism and giving complete freedom to implement the behavior of the researcher, this method is considered the most natural for simulation.

There are already quite a lot of full-fledged products on the market of crowd modeling tools, which are mainly created for designing urban architecture with a view to avoiding crowds and modeling crisis situations. They are all commercial products and because of this, access to intellectual property like code and model calculation algorithms is impossible. Despite this, as mentioned at the beginning of the section, the field of crowd prediction in various emergencies is a very popular field of research in the academic field, so we can analyze the developed models of scientific work. In research [2] a model was created to simulate the behavior of the crowd in case of fire and to calculate the optimal evacuation routes of a given room. Submodels were created in the work, which paid attention to the flammability of materials and the spread of fire and smoke throughout the building. In the model [3] the work focused on the concept of perception of the environment by the agent in the scenario of evacuation of the crowd from a certain area. Each agent, through both close verbal communication and remote communication, receives information to supplement the completeness of the map of perception of the area. Each agent subjectively perceives his environment and makes decisions through a progressive decision-making module. This model was also used to emulate the spread of information in crowds during evacuations. The work [4] is interesting in that each of the agents has its own socio-psychological state, which affects how the agent makes decisions and sets priorities. The basic idea of this model is that different people may react differently to the same stimulus depending on their individual characteristics and psychological characteristics.

The NetLogo software is written with the aim of having many opportunities for experts, was also considered. NetLogo allows to influence the simulation with various switches, sliders, buttons and other interface elements [5].

The NetLogo environment allows scientists to study emergent phenomena that occur in systems of elements, but are not inherent in individual elements of the system. NetLogo includes models from various fields of modeling, such as economics, physics, chemistry, psychology, biology. In addition to using existing models, NetLogo allows to create and distribute custom models.

Mesa is a specialized agent modeling framework for the Python language. Created with the aim of becoming the main tool for researchers who want to create agent models using the Python language. It allows to create agent models using basic built-in components or create and manually customize custom models. Mesa also allows to visualize the simulations and analyzes using a graphical browser interface. The

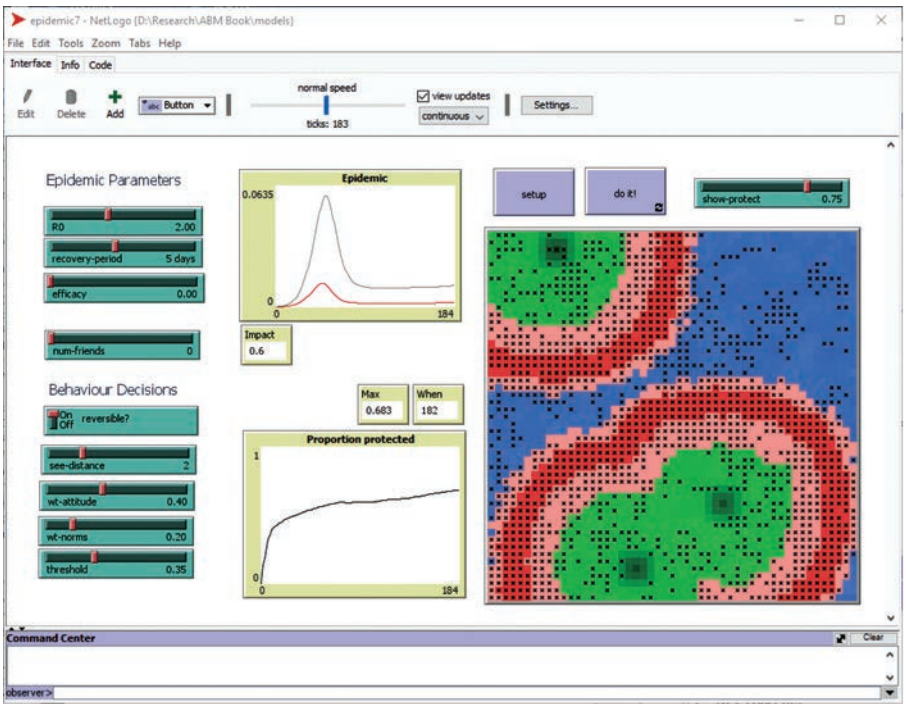

Fig. 3. NetLogo interface [5]

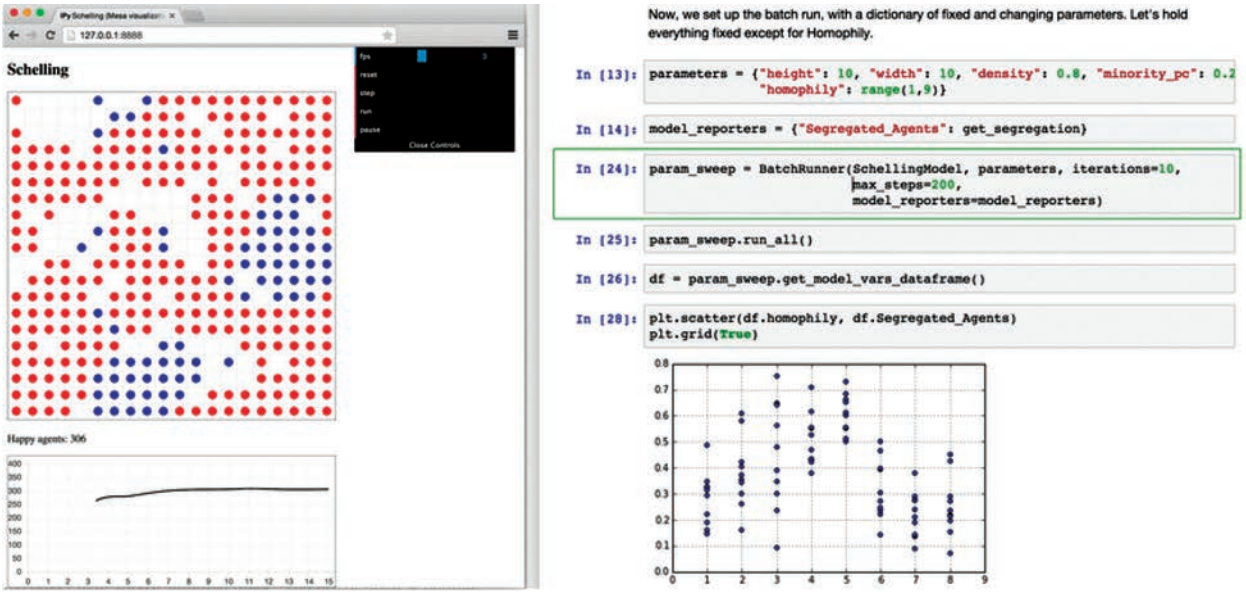

Fig. 4. Example of a model interface created in Mesa [7]

main advantage of Mesa over other agent modeling tools is that Mesa's design is divisible and extensible, thus allowing the creation of a decentralized ecosystem of specialized user-created modules.

To create agents and models, we inherit the agent or model class provided by Mesa. The databases of these classes contain all the basic functions for the interaction of the agent and the model [6].

A Pathfinding Project* is a powerful path search system in Unity. It allows to find the most efficient path for a moving object in a few lines of code. The system supports grid, navigation and hexagonal graphs. The system can automatically generate navigation graphs, so that the developer does not waste time on manual editing of navigation. A* Pathfinding Project is a multi-threaded project, so even calling many paths at once will not affect program performance. The processing of the created paths includes smoothing and a funnel algorithm. Already created paths can be easily modified to suit your needs. The system supports

updating of graphs during the program operation, and also has complete and clear documentation [8]. A* is a mathematical search algorithm in the finding graph the route with the lowest path cost from the initial vertex to the final one, in graphs with positive edge weights. The algorithm is an extension of Dijkstra's algorithm. The difference A* from Dijkstra's algorithm is the use of a heuristic estimate of the distance from the selected vertex to the initial one. The algorithm tries to minimize $f(n)$:

$$f(n)= g(n) + h(n), \quad (1)$$

where n is the next vertex under consideration, $g(n)$ is the cost of the path from the initial vertex to the vertex n, and $h(n)$ is a heuristic function that approximates the cost of the cheapest path from n to the target vertex. An example of a heuristic function for finding a path can be the distance from the vertex to the goal, as the physically smallest possible distance between points.

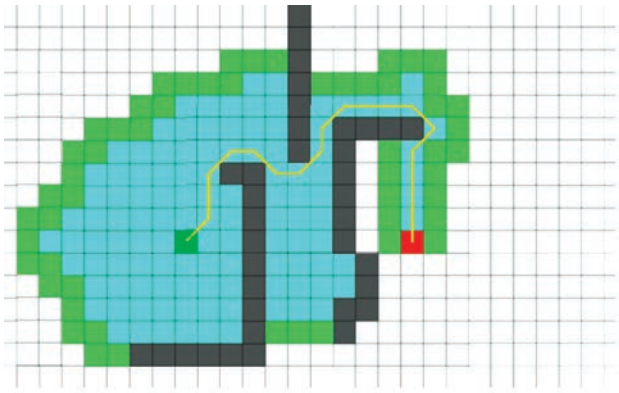

Fig. 5. A*algorithm operation example [8]

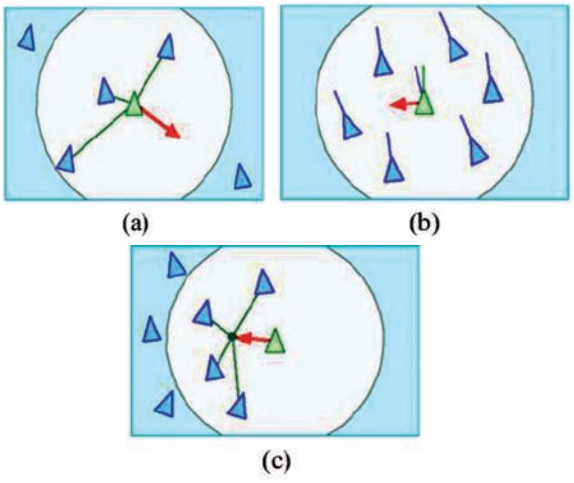

Fig. 6. Using Boids algorithm to describe the agent's behavior in the crowd (a – separation rule, b – alignment rule, c – cohesion rule)

Architecture and modules of the developed software

Cross-platform tool Unity was chosen because of the number of embedded systems focus not on the implementation of graphical, physical and other components, but on the algorithms and solution models for agents. The Python language and the Django framework are selected for the server part used to store the program data. Python has the advantage of writing and debugging code over other programming languages, and Django allows to write web servers with a ready-made graphical CRUD interface in a short amount of time and programmatically access the database using the built-in ORM. The PyCharm IDE, the most popular Python development tool, was chosen as the server development environment. Each of the software modules performs a specific function that ensures the correct interaction of the system. In general, all programs developed on the Unity platform are subject to a similar architecture. This does not prevent the implementation of software templates and other architectural solutions, in order to increase the efficiency and speed of product writing. Each of the modules can exist independently of each other, but somehow interacts with other modules of the system through calls to event functions. Event functions are a structural feature of programs in Unity.

Scripts in Unity are not subject to the usual type, executed in a loop until they fulfill the set goal. Unity passes control of the script by actually performing the functions of the events described in it. Just a function finished running, control is passed back to Unity, and control is passed to the next function. These functions are called event functions because they are called by the Unity controller in response to events between components. Event functions differ from ordinary ones by predefined names, according to their functionality. The most commonly used event functions Update is performed each time before the program cycle (frame) is updated, and Start, which is called before the first frame and is used to initialize the component. In addition, there are many functions created specifically to respond to certain inter-component events. The following modules can be distinguished in the structure of the developed software: environmental module; agent module; agent generation module; scene editing module; module of interaction with the database. Each of these modules performs a separate function and is connected to the others by objects and events in the program scenes.

The Boids algorithm is used to describe the agent's behavior in the crowd (fig. 6). Separation means a change of direction in order to avoid collisions with close agents. Alignment means changing the direction of movement in accordance with the direction of

movement of neighbors. Cohesion means the desire to change position in the direction of the middle position or center of mass of close agents.

Figures 7, 8 and 9 show the editing of the simulation scene of the developed software, the process of modeling and testing of algorithms.

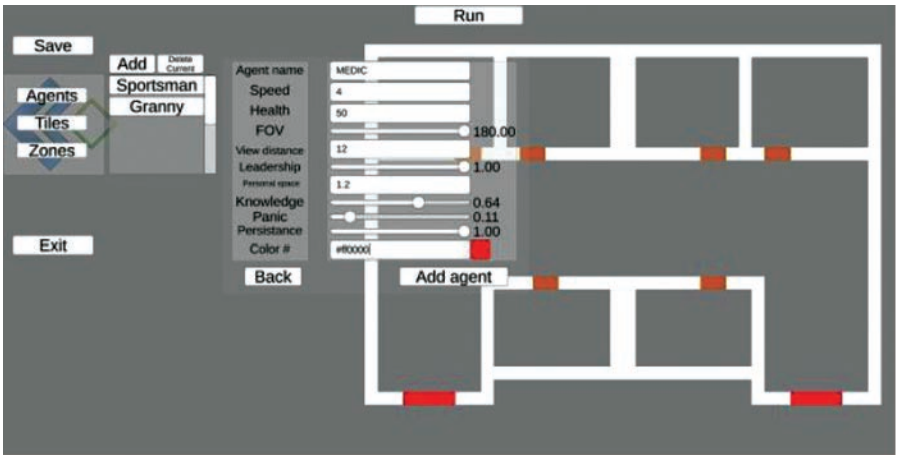

Fig. 7. Editing the simulation scene of the developed software

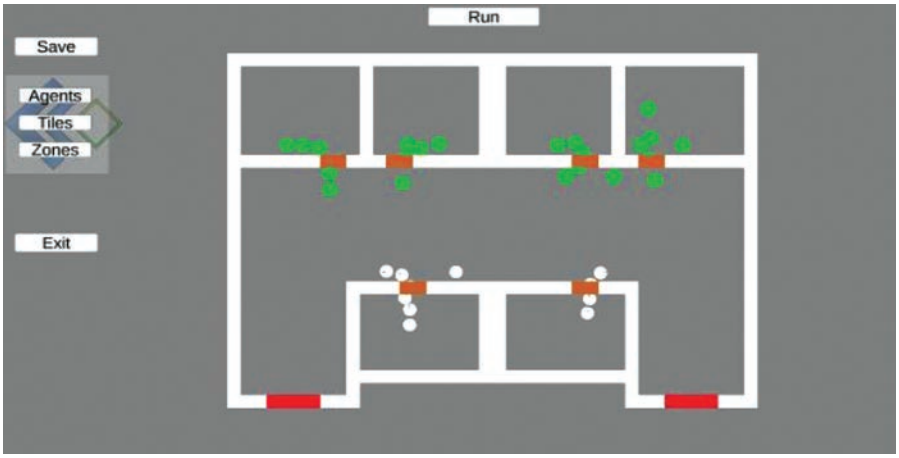

Fig. 8. Modeling process of the developed software

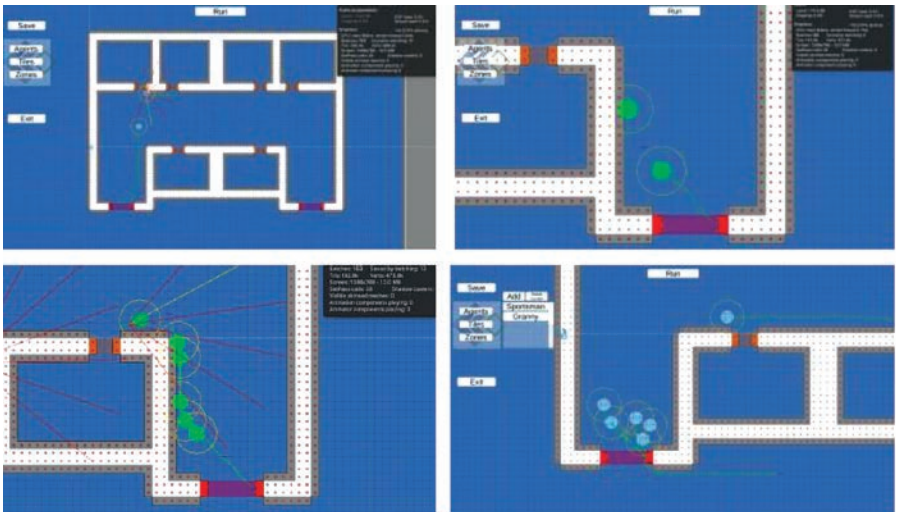

Fig. 9. Testing algorithms

Conclusions and future work. In this research existing software solutions for agent modeling and projects are analyzed. Requirements for the developed software are formed and defined. The developed software allows to model the behavior of people during an emergency situation. The developed software was tested and the results according to which the software meets the requirements were analyzed. Ways to further develop the proposed software are to create a module for viewing simulations, machine analysis of simulations, creating dynamic simulation schedules, introducing new interactive objects of interest to the agent and improving the user interface of the program.

References:

1. Crowd Simulation Modeling Applied to Emergency and Evacuation Simulations. URL: https://arxiv.org/ftp/arxiv/papers/1303/1303.4692.pdf.
2. Agent-Based Evacuation Model Incorporating Fire Scene and Building Geometry. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber =6074190.
3. Agent Perception Modeling for Movement in Crowds. URL: https://www.researchgate.net/publication/235919357_Agent_Perception_Modeling_for_Movement_in_Crowds.
4. Crowd simulation influenced by agent's socio-psychological state. URL: https://arxiv.org/abs/1004.4454.
5. NetLogo. URL: https://ccl.northwestern.edu/netlogo./
6. Utilizing Python for Agent-Based Modeling: The Mesa Framework. URL: https://www.researchgate.net/publication/344675633_Utilizing_Python_for_Agent-Based_Modeling_The_Mesa_Framework.
7. A Mesa implementation of the Schelling segregation model. URL: https://www.researchgate.net/figure/A-Mesa-implementation-of-the-Schelling-segregation-model-being-visualized-in-a-browser_fig1_328774079.
8. A* Pathfinding Project. URL: https://arongranberg.com/astar/front.

Олещенко Л.М., Медведєв М.Г., Кобрин Д.Р., Сукало М.Л. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ АГЕНТНОГО МОДЕЛЮВАННЯ ПОВЕДІНКИ НАСЕЛЕННЯ У НАДЗВИЧАЙНИХ СИТУАЦІЯХ

Можливість прогнозування поведінки населення в надзвичайних ситуаціях цінна в багатьох контекстах. Ситуації, пов'язані з панікою, використовуються для моделювання поведінки натовпу й розроблення програмних рішень для задач евакуації населення під час виникнення надзвичайної ситуації. У статті розглянуто дослідження моделей руху людей у надзвичайних ситуаціях, проаналізовано існуючі фреймворки й програмні засоби для агентного моделювання поведінки натовпу. На основі проведених досліджень про поведінку людей у натовпі й доступних підходів до моделювання руху людей і натовпу в надзвичайній ситуації сформульовано вимоги до програмного забезпечення: можливість конфігурації кожного з агентів; можливість зміни геометрії приміщень; параметризація моделі; коректна робота алгоритмів за великої кількості агентів; урахування поля зору агентів; реалізація дотримання особистого простору людини; реалізація алгоритму для зміни траєкторії руху в разі зіткнення; реалізація феноменів паніки натовпу; розрахунок густини натовпу; можливість випадкового розподілу агентів; можливість ручного розподілу агентів; імпорт та експорт результату аналізу програми у формат JSON. Представлено загальний опис архітектури й розроблених програмних модулів створеного програмного забезпечення. Створені модулі містять модуль навколишнього середовища, модуль агента, модуль генерації агентів, модуль редагування сцени, модуль взаємодії з базою даних. Описано архітектуру програмного забезпечення, розробленого на платформі Unity, і взаємодію основних компонентів. Для розроблення програмного забезпечення використано технології A Pathfinding Project для забезпечення гнучкого й надійного пошуку шляхів, фреймворк Django для створення API серверу збереження даних, бібліотеку djongo для інтеграції з базою даних MongoDB.*

Ключові слова: *програмне забезпечення, надзвичайна ситуація, агентне моделювання, Python, фреймворк Django, Unity, MongoDB.*