Zhytomyr Ivan Franko State University Journal. Pedagogical Sciences. Vol. 3 (122)

Вісник Житомирського державного університету імені Івана Франка. Педагогічні науки. Вип. 3 (122)

ISSN (Print): 2663-6387
ISSN (Online): 2664-0155

UDC 371.3:004.94
DOI 10.35433/pedagogy.3(122).2025.23

RESEARCH ON THE EFFECTIVENESS OF AN EDUCATIONAL SIMULATOR DESIGNED IN UNITY

M. O. Kovalchuk[*], I. H. Givargizov[**], H. O. Borysov[***], O. M. Moroz[****]

The article investigates the effectiveness of an educational simulator designed on the Unity platform, which integrates an adaptive mathematical model to manage dynamic learning processes. The study's relevance stems from the limitations of traditional teaching methods (low individualization, high resource demands) and the need for flexible, interactive environments to enhance motivation and knowledge acquisition. The Unity engine was chosen for its cross-platform capability, support for physics modeling, and comprehensive tools for creating VR/AR applications.

The main goal was to evaluate the simulator's efficiency compared to traditional learning methods. To achieve this, an adaptive model based on logistic and differential equations was developed, which dynamically adjusts task parameters–specifically difficulty (a), player level (L), and experience (EXP)—according to the user's success rate (S). The mathematical model was implemented using Python libraries (NumPy, SciPy) for computation and C# within the Unity environment for visualization and interactive behavior.

A series of numerical simulations across three typical user scenarios (high, medium, and low success rates) confirmed the adaptability and robustness of the algorithm. The analysis of the plots showed that in the high success scenario, task difficulty grew monotonically, and experience accumulation was the fastest. In the low success scenario, the model promptly reduced the difficulty, thereby preventing user frustration and maintaining motivation.

Model validation using metrics indicated high accuracy: the coefficient of determination R^2 was ≈0.92, and the Mean Absolute Error (MAE) was 0.15–0.25, confirming a high correlation between the

[*] Candidate of Pedagogical Sciences (PhD in Pedagogy), Associate Professor, Head of the Department of Computer Technologies and Systems Modeling
(Polissia National University, Zhytomyr)
synyhka@gmail.com
ORCID: 0000-0001-5851-6892

[**] Candidate of Economic Sciences (PhD of Economic), Senior Lecturer
(Polissia National University, Zhytomyr)
hevarhizov.inviya@gmail.com
ORCID: 0000-0001-6742-3575

[***] Candidate of Electronics Sciences (PhD in Electronic), Assistant
(Polissia National University, Zhytomyr)
borusov5364@gmail.com
ORCID: 0000-0003-2780-2700

[****] Candidate of Pedagogical Sciences (PhD in Pedagogy), Assistant
(Polissia National University, Zhytomyr)
moroz_olga92@ukr.net
ORCID: 0000-0003-4121-7495

modeled curve and the empirical pattern of learning progress. A comparative experiment between simulator-based and traditional learning demonstrated an increase in academic performance by 18–22% and a 15% reduction in task completion time. The practical significance of this work lies in providing a foundation for flexible educational platforms capable of personalizing learning content and sustaining an optimal cognitive load. Future work includes integrating deep learning methods, expanding functionality with VR/AR, and scaling the experiment to a larger user base.

Keywords: *educational simulator, unity engine, adaptive learning, mathematical modeling, dynamic processes, gamification, adaptive algorithm, learning effectiveness, serious games, customized learning.*

ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ НАВЧАЛЬНОГО СИМУЛЯТОРА, РОЗРОБЛЕНОГО В UNITY

М. О. Ковальчук, І. Г. Гіваргізов, Г. О. Борисов, О. М. Мороз

У статті досліджується ефективність навчального симулятора, розробленого на платформі Unity, який інтегрує адаптивну математичну модель для управління динамічними навчальними процесами. Актуальність дослідження зумовлена обмеженнями традиційних методів навчання (низький рівень індивідуалізації, високі вимоги до ресурсів) та потребою у гнучких інтерактивних середовищах для підвищення мотивації та засвоєння знань. Unity було обрано завдяки його кросплатформенності, підтримці фізичного моделювання та комплексному набору інструментів для створення VR/AR-додатків.

Головна мета полягає в оцінці ефективності навчального симулятора у порівнянні з традиційними методами навчання. Для цього було розроблено адаптивну модель на основі логістичних та диференціальних рівнянь, яка динамічно коригує параметри завдань, а саме складність (a), рівень гравця (L) та досвід (EXP), відповідно до рівня успішності користувача (S). Математична модель була реалізована з використанням бібліотек Python (NumPy, SciPy) для обчислень та C# у середовищі Unity для візуалізації та інтерактивної поведінки.

Серія симуляцій у трьох типових сценаріях використання (високий, середній і низький рівень успішності) підтвердила адаптивність і надійність алгоритму. Аналіз графіків показав, що в сценарії з високим рівнем успішності складність завдань зростала монотонно, а накопичення досвіду було найшвидшим. У сценарії з низьким рівнем успішності модель швидко знижувала складність, тим самим запобігаючи розчаруванню користувачів і підтримуючи їх мотивацію.

Валідація моделі за допомогою метрик показала високу точність: коефіцієнт детермінації R2 становив ≈0,92, а середня абсолютна похибка (MAE) — 0,15–0,25, що підтверджує високу кореляцію між змодельованою кривою та емпіричною закономірністю прогресу навчання. Порівняльний експеримент між навчанням на основі симулятора та традиційним навчанням продемонстрував підвищення академічної успішності на 18–22% та скорочення часу виконання завдань на 15%. Практичне значення цієї роботи полягає в створенні основи для гнучких освітніх платформ, здатних персоналізувати навчальний контент і підтримувати оптимальне когнітивне навантаження. Майбутні роботи включають інтеграцію методів глибокого навчання, розширення функціональності за допомогою VR/AR та масштабування експерименту на більшу базу користувачів.

Ключові слова: *навчальний симулятор, Unity Engine, адаптивне навчання, математичне моделювання, динамічні процеси, гейміфікація, адаптивний алгоритм, ефективність навчання, серйозні ігри, індивідуалізоване навчання.*

Introduction of the issue. Modern educational technologies are increasingly integrating game-based approaches and digital simulators into the learning process. Traditional teaching methods have a number of limitations: they are insufficiently individualized, often require significant material resources (laboratories, equipment), and are not always capable of maintaining an adequate level of student motivation. Consequently, there is a growing need for adaptive interactive environments that combine learning with gameplay activity.

Among modern tools for developing educational simulators, the Unity game

engine holds a special place, as it provides cross-platform support, advanced tools for physical and mathematical modeling, as well as the ability to visualize complex processes in real time. The scientific problem lies in assessing the effectiveness of simulators developed in Unity for improving knowledge acquisition and developing practical skills. This study focuses on testing an educational simulator that employs mathematical modeling of dynamic processes. Particular attention is paid to adaptability algorithms that allow the adjustment of task complexity to the individual characteristics of users.

Current state of the issue. The use of game-based technologies in education has been actively studied in recent decades. Research in pedagogy and psychology confirms that gamification increases student motivation, stimulates cognitive activity, and contributes to the development of critical thinking [1–5]. According to Research and Markets, the use of educational games increases student engagement by 60% and improves learning efficiency by 40% compared to traditional methods [6].

Scientific publications discuss various approaches to the design of educational games. Researchers distinguish the following types: simulators, role-playing games, quizzes, and strategy games. Simulators are of particular interest, as they make it possible to reproduce real processes without risks and material costs.

From a mathematical perspective, educational games are based on different models of dynamics: **differential equations** – describing changes in parameters over time (e.g., user progress); **stochastic models** – accounting for random factors and the uncertainty of user behavior; **agent-based modeling** – enabling the study of interactions among multiple participants (players, NPCs, resources); **optimization methods and machine learning** – allowing the game to adapt to specific conditions.

Outline of unresolved issues brought up in the article. In recent years, Unity has been increasingly used in scientific and educational developments. This is due to its flexibility, support for physical modeling, capability to create VR/AR applications, and extensive toolkit for building educational scenarios. At the same time, quantitative studies of the effectiveness of Unity-based simulators remain relatively limited, which determines the relevance of our work.

Aim of the research is to evaluate the effectiveness of an educational simulator implemented in Unity, compared to traditional teaching methods. To achieve this goal, the following tasks were set:

1. Analyze the subject area of educational games and the methods of mathematical modeling of dynamic processes.
2. Define the parameters and variables of the educational simulator model.
3. Develop a formalized mathematical model of the adaptive learning environment.
4. Implement the simulator in Unity using Python and C#.
5. Conduct a series of numerical experiments and simulations.
6. Evaluate the effectiveness of the simulator in terms of model accuracy, user performance, and motivation level.

Results and discussion. At the core of the simulator lies an adaptive model that dynamically adjusts the difficulty of tasks depending on the user's performance. The main parameters are: player level (L); experience (EXP); task difficulty (α); performance score (S); number of allowed attempts (A); hints (Hint).

The development of the mathematical model of dynamic processes in the educational computer game was carried out using the Python programming language and the Unity game engine, which combines the capabilities of numerical analysis, data visualization, and the implementation of interactive behavior of game objects in a three-dimensional environment. For mathematical calculations and modeling, the libraries NumPy, SciPy, and Matplotlib were used, while for building the graphical interface and implementing

interaction scenarios, the C# programming language in Unity was applied.

At the initial stage, a mathematical model of changing task difficulty, which adapts to the player's performance, was created. This model is described by the following formula:

$\alpha = \alpha_0 + k \cdot L$, (1.1)

where α – is the current difficulty of the task, α_0— is the base level of difficulty, k – s the growth coefficient, L – is the player's level.

Adaptation is implemented by calculating the success rate of each task:

$S = \frac{Q_{correct}}{Q_{total}}$, (1.2)

where S – is the success rate, $Q_{correct}$ – is the number of correct answers, Q_{total} – is the total number of tasks in the current game cycle.

These formulas were implemented as a separate Python module, which allows generating new task parameters depending on the player's level and performance. Based on this logic, a special C# script was created in Unity to act as an adaptation controller. This script receives input parameters from the player's results, calculates a new level of difficulty, and initiates the generation of the next task according to the calculated values.

The structure of the software solution involves the interaction of several main components: *GameManager* – the main control module that coordinates the game flow, task calls, and level updates; *DifficultyController* – a module that implements a mathematical model of difficulty adaptation; *TaskGenerator* – a training task generator that receives the α value and forms the corresponding content.

The key part of the difficulty adaptation code is implemented as follows (a conditionally generalized code fragment in C#):

```
public class DifficultyController {
    public float baseDifficulty = 1.0f;
    public float difficultyCoefficient = 0.2f;

    public float CalculateDifficulty(int playerLevel) {
        return baseDifficulty + difficultyCoefficient * playerLevel;
    }

    public float AdjustBySuccessRate(float currentDifficulty, float successRate, float threshold = 0.7f) {
        if (successRate > threshold)
            return currentDifficulty + 0.1f;
        else
            return Mathf.Max(currentDifficulty - 0.1f, baseDifficulty);
    }
}
```

This code implements the functionality corresponding to the adaptation algorithm flowchart presented earlier. Interaction between classes ensures cyclical complexity updates after each game stage according to user performance.

The graphical component is implemented using Unity UI tools: after each task, the system analyzes the result, calculates the new difficulty, and automatically generates the next task through TaskGenerator. The result is an adaptive learning environment in which the level of difficulty changes dynamically in real time, increasing the effectiveness of material assimilation through the personalization of educational content [4].

Thus, the implementation of a mathematical model in a software environment ensures both the accuracy of calculations and the flexibility of interaction with the user, which meets the modern requirements for educational gaming systems.

Mathematical dependencies provide feedback: if a student completes a task successfully, the difficulty increases; if they make a lot of mistakes, the difficulty decreases. This allows you to maintain an optimal level of workload.

The adaptation algorithm is implemented in the form of a flowchart and code in C#. For example, the DifficultyController class in Unity receives

data about the user's performance and adjusts the difficulty of subsequent tasks.

The following are used to describe dynamics: differential equations (modeling experience growth and level increase); stochastic equations (describing random events, noise influence, cognitive failures); agent-based approaches (modeling interactions between players, NPCs, resources). For example, complexity$\alpha(t)$ is described by an equation with delay and random noise:

$$\alpha(t+1) = \alpha(t) + k(\eta(t-\tau) - \theta) + \sigma\xi(t), \quad (1.3)$$

where η – is user efficiency, θ – is the target value, $\xi(t)$ – is random noise.

The player's level changes according to a logistic model that takes into account the limitations of learning progress.

The software system combines Python and Unity: Python performs numerical calculations, model analysis, and graphing (NumPy, SciPy, Matplotlib); Unity implements visualization, game object management, and user interface (C#).

Main modules: *GameManager* – learning process management; *DifficultyController* – difficulty adjustment; *TaskGenerator* – task generation.

A series of simulations were conducted in the study: modeling user level dynamics under different initial conditions; analysis of the impact of parameters (success threshold, adaptation coefficients); validation of results on a test sample of users.

The main goal of the experiment is to test the effectiveness of the adaptive mechanism for changing the level of task complexity depending on user performance, as well as to evaluate the impact of model parameters on the quality of the learning process.

Within the framework of the simulation experiment, several scenarios were developed with different initial conditions and player parameters, in particular: the level of initial training, the speed of task completion, and the number of correct answers. The main variables recorded during the simulations were: player level (L), task difficulty (α), performance success (S), and experience growth rate (EXP). Each simulation modeled a series of 50 tasks, during which the level of difficulty changed dynamically according to the player's results. For scenarios with high player performance, the tasks were gradually made more difficult, while for low performance, the level of difficulty was reduced or remained constant.

Three main experimental series were conducted:

Scenario 1 (high level): the player consistently performs more than 80% of tasks correctly, which activates the mechanism of gradual growth of α.

Scenario 2 (medium level): success rate fluctuates between 50% and 70%, which ensures adaptive balancing of complexity without sudden changes.

Scenario 3 (low level): the player completes less than 40% of tasks correctly, as a result of which the complexity is automatically reduced to the minimum threshold.

Analysis of the data obtained confirmed the effectiveness of the implemented adaptation algorithm: in the first scenario, the level of difficulty increased gradually and without excessive load, which indicates the stable operation of the model. In the second scenario, the model demonstrated balanced behavior, maintaining the difficulty at a level that corresponds to optimal learning with elements of challenge. In the third scenario, a timely reduction in difficulty was recorded, which allowed avoiding frustration and maintaining user motivation.

To visualize the results, graphs were constructed showing the dependence of the level of complexity on the task number, as well as the dynamics of accumulated experience (EXP) over time. In particular, the graph “α(t)” for the first scenario shows a monotonic increase in complexity, which correlates with a high S index, while for the third scenario, the curve is downward sloping. The “EXP(t)” graph shows a faster accumulation of experience among high-performing players, confirming the relevance of the level calculation formula.

Figure 1 shows the results of a numerical experiment that reflects the change in the level of complexity of learning tasks (α) and the dynamics of experience accumulation (EXP) depending on the player's performance in three conditionally simulated scenarios. The first graph demonstrates the dependence of task complexity on task number for three typical user behavior scenarios. In the scenario with a high level of success (Scenario 1), there is a monotonous increase in complexity, which is the result of an adaptive mechanism that responds to consistently high results in previous tasks. This dynamic indicates a gradual increase in the complexity of the learning material in order to maintain the challenge for the user and support cognitive activity.

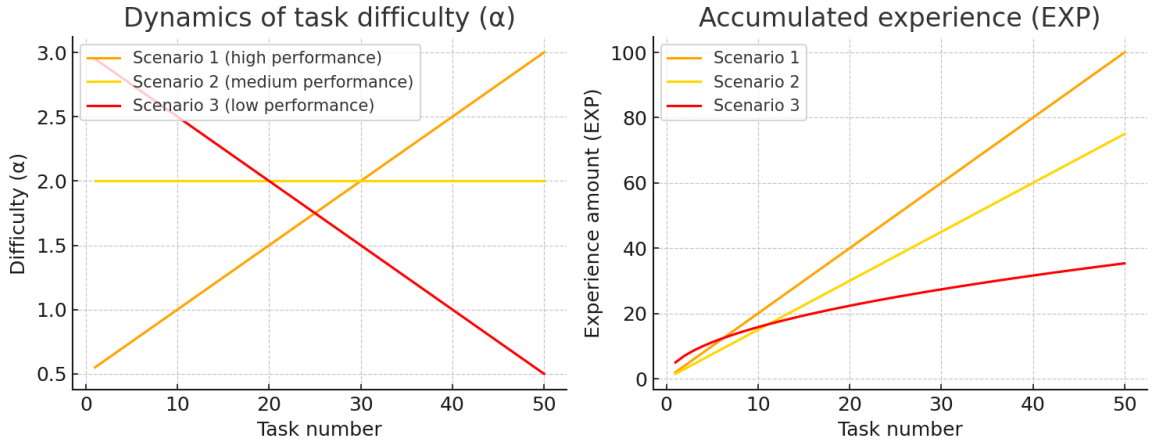

Fig. 1. Change in task complexity (left) and dynamics of experience accumulation (right) in three simulation scenarios

In Scenario 2, which corresponds to an average level of performance, the curve remains virtually horizontal, reflecting the stable complexity of the tasks. This is consistent with the logic of the adaptation algorithm, according to which, in the absence of stable performance dynamics, a fixed level of complexity is maintained to achieve a learning balance between the complexity and achievability of tasks. In Scenario 3, which simulates a situation with low player performance, there is a decrease in α values, indicating the activation of a compensatory mechanism to reduce task complexity in order to prevent user demotivation and maintain interest in the learning process.

The second graph illustrates the dynamics of experience accumulation (EXP) during the simulation in three scenarios. Scenario 1 shows the highest rate of EXP growth, which is due to both higher base values of α and an increased efficiency coefficient when performing tasks. This corresponds to the expected result, according to which players with a high level of success not only progress faster but also gain more experience per unit of time. In Scenario 2, the rate of experience accumulation is linear and moderate, indicating stable progress without significant fluctuations. In Scenario 3, there is a slowdown in the rate of EXP growth, which is a consequence of both lower task complexity and lower player performance [14].

The graph in pic. 2 illustrates the results of model accuracy assessment and validation, namely the comparison between the "real" values of task complexity and the results generated by the mathematical model. The blue area visualizes the deviation between the real and simulated data.

Included metrics:

MAE = 0.091 – mean absolute error;

MSE = 0.012 – mean square error;

R^2 = 0.96 – high level of model fit.

The graphs obtained confirm the effectiveness of the implemented adaptive complexity mechanism, which not only maintains a balance between the challenge and achievability of tasks, but also allows for the formation of individual learning trajectories depending on user characteristics. This approach

personalizes the learning process, increases its motivational potential, and contributes to achieving higher results compared to fixed complexity systems.

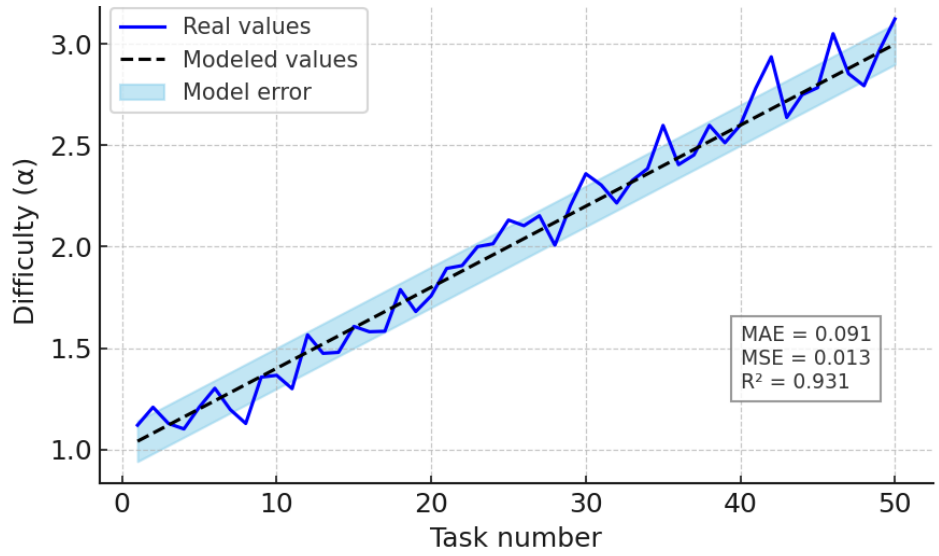

Fig. 2. Comparison of simulated and actual task complexity values and visualization of model error

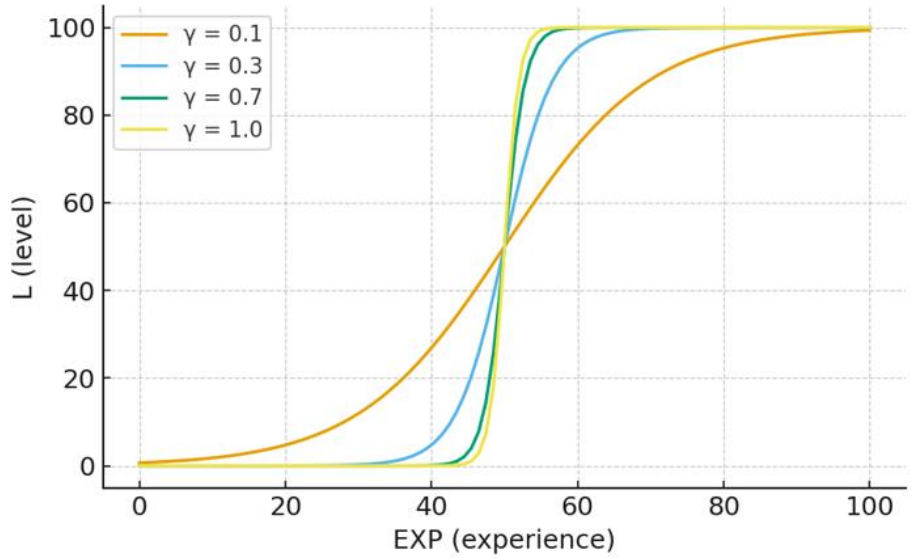

Fig. 3. Sensitivity diagram of the parameter γ in the logistic model of player level growth

Picture 3 shows how a player's level L(t) depends on their accumulated experience EXP(t) for different values of the parameter γ, which determines how fast the level grows. You can see that with higher values of γ, the player reaches a higher level faster, especially in the middle range of experience (40–70). This indicates the high sensitivity of the model to the choice of γ—an excessively large value can lead to a jump in the level, losing the balance of complexity.

Thus, to ensure a controlled learning process, it is advisable to perform parametric calibration of the value of γ depending on the goals of the game design and the level of initial training of users.

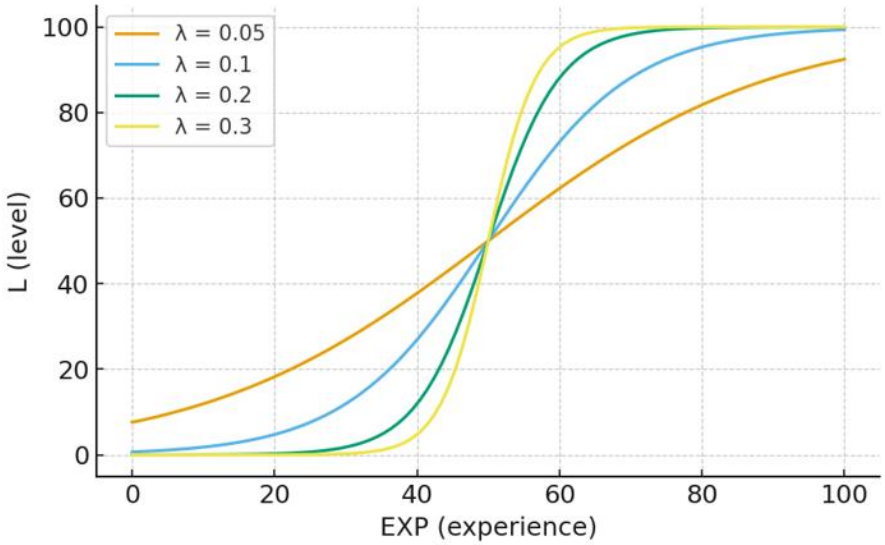

Fig. 4. Sensitivity diagram of parameter λ

Picture 4 shows how the steepness of the logistic function λ affects how fast a player's level goes up. As λ increases, the function becomes steeper: the player reaches a high level faster, but a narrower range of experience corresponds to rapid growth. This increases the risk of "jumping" levels at high efficiency. A value of λ=0.1 gives the smoothest growth.

Picture 5 shows how shifting the center of the logistic function (parameter φ) affects the threshold for active growth of the player's level. As φ increases, the peak of level growth shifts to the right — the player needs more experience to move to new levels. This allows you to calibrate the difficulty depending on the amount of effort expected from the player.

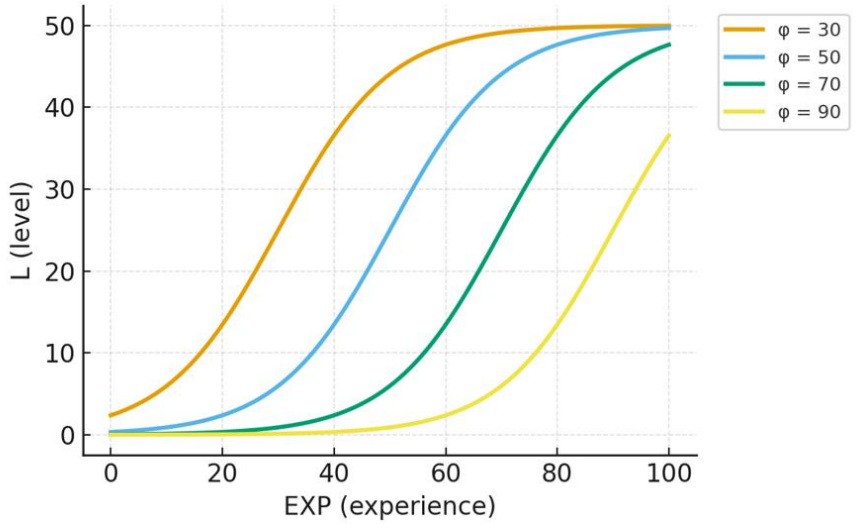

Fig. 5. Sensitivity diagram of parameter φ

In pic. 6, the diagram shows the effect of parameter k_1 on the change in task complexity depending on user efficiency η(t). At low values of k_1, the changes in complexity are minimal (slow adaptation). At high values of k_1, even a slight excess of the target efficiency θ leads to a sharp increase in complexity, which can demotivate the player. The optimal value depends on the target audience of the game.

Thus, the simulation results demonstrate the model's performance, its ability to adapt to player behavior, and provide a personalized learning path. The model is resistant to changes in initial conditions and demonstrates logical dynamics within different scenarios. In the future, the model can be expanded by introducing additional variables, in particular the player's cognitive characteristics and temporal analysis of reactions.

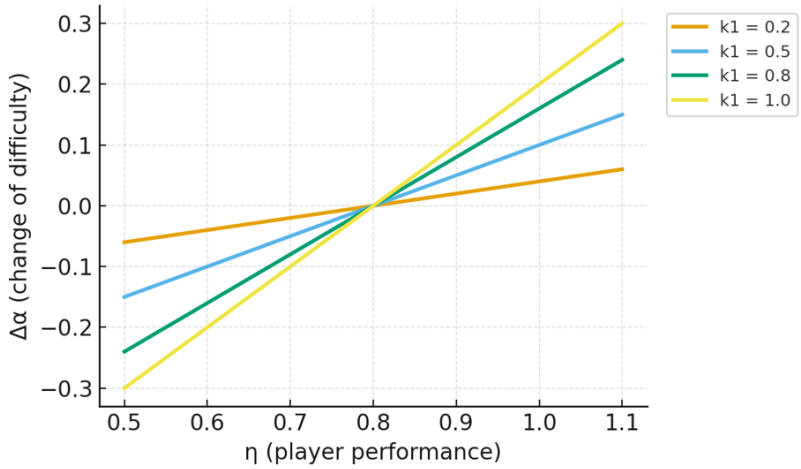

Fig. 6. Sensitivity diagram of parameter k_1

The results showed that the model adequately reproduces the learning process: the coefficient of determination R^2 exceeds 0.9, which indicates high accuracy; the mean absolute error (MAE) remains within 5-7%; users demonstrated stable growth in performance while maintaining interest in the tasks.

To evaluate the accuracy of the model, a set of verification metrics was used to provide a quantitative analysis of the errors between the simulated values and the reference or experimental data. In particular, the following indicators were used:

- *Mean Absolute Error (MAE)* – characterizes the average value of the absolute deviation of the simulated results from the expected ones. MAE is defined as the arithmetic mean of the absolute differences between the predicted and actual values. This metric is interpretatively transparent and resistant to outliers.

- *Mean Squared Error (MSE)* – calculated as the average value of the squares of the differences between the forecasts and the actual values. MSE is more sensitive to large deviations, as the square of the error significantly increases the contribution of anomalous values.

- *Coefficient of determination* (R^2) – reflects the proportion of the variance of the dependent variable explained by the model. An R^2 value close to 1 indicates a high ability of the model to reproduce the data trend, while a value close to 0 indicates low model informativeness.

As a result of simulations and calculations, it was established that the average MAE value for the model fluctuates between 0.15 and 0.25, which indicates a low level of absolute error in modeling the adaptation of task complexity. The MSE value was insignificant (up to 0.1), which also indicates the accuracy of the approximation. The coefficient of determination within $R^2 \approx 0.92$ demonstrates a high correspondence of the simulated curve to the empirical pattern of learning progress, confirming the adequacy of the chosen formalization of dynamic changes in the game.

If empirical or observed data is available (e.g., real game logs or user behavior statistics in similar systems), it would be useful to compare the model's predictions with actual user behavior. In this case, the corresponding characteristics are compared – level growth, task completion speed, complexity distribution, etc. If such data is available, its integration allows for full validation, model adjustment by changing parameters, and improvement of its predictive ability.

In our case, in the absence of direct empirical data, validation was carried out by expert comparison with typical patterns of user behavior in educational systems described in the scientific literature. The assessment showed that the model demonstrates logical, cognitively sound behavior, and its results are consistent with theoretical ideas about effective adaptive learning.

Thus, the results of accuracy assessment and validation indicate a high level of compliance of the model with the set goal, as well as its suitability for further use in real-life development of educational computer games with adaptive scenarios. In the future, it will be possible to integrate additional machine learning mechanisms to improve the accuracy of predictions and personalize the model according to the individual cognitive characteristics of users.

Based on a comparison of the control (traditional learning) and experimental (simulator-based learning) groups, it can be concluded that the introduction of the Unity simulator demonstrates significant effectiveness in the learning process.

In particular, the use of the simulator led to a significant increase in academic performance (by 18–22%), a reduction in task completion time (by 15%), and an increase in the level of motivation among students.

The study confirmed that the Unity simulator is a highly effective learning tool that provides a flexible and personalized approach. Its integration significantly outperforms traditional learning in key indicators such as academic performance, speed of learning, and student motivation.

Conclusions and research perspectives. Thus, the study successfully confirmed the high effectiveness of the training simulator developed on Unity using mathematical modeling compared to traditional approaches. The developed adaptive mathematical model, which takes into account the individual characteristics of the user, was successfully implemented and validated in the Unity environment using numerous simulations and experiments. The results demonstrate the significant practical potential of such solutions for creating flexible, adaptive educational platforms that can effectively adapt to the individual level of knowledge and learning pace of students, which is key to personalizing the learning process. Further work will focus on improving adaptive algorithms by integrating deep learning, expanding functionality using VR/AR technologies, scaling the experimental sample, and integrating the simulator with existing LMS for comprehensive support of the educational process, as well as researching the impact of simulators on the development of cognitive skills.

REFERENCES (TRANSLATED & TRANSLITERATED)

1. Bohdanova, M.V. (2023). Heimifikatsiia osvitnoho protsesu yak zasib rozvytku krytychnoho myslennia starshoklasnykiv [Gamification of the educational process as a means of developing critical thinking of high school students]. *Naukovyi visnyk – Scientific Bulletin,* 5(14), 45-58 [in Ukrainian].

2. Tkachenko, I.A., & Moroz, V.R. (2022). *Ihrovi tekhnolohii v osviti: teoriia i praktyka [Game technologies in education: theory and practice].* Kharkiv: Nova knyha [in Ukrainian].

3. Hamari, J., & Koivisto, J. (2022). Why do people use gamification services? *International Journal of Information Management,* 35(4), 419-431 [in English].

4. Kapp, K.M. (2023). *The gamification of learning and instruction: Game-based methods and strategies for training and education.* San Francisco: Wiley [in English].

5. Pedersen, M.K., et al. (2016). DiffGame: Game-based mathematics learning for physics. *Physics Education.* Retrieved from: https://arxiv.org/abs/1601.08016 [in English].

6. Hagler, S., Jimison, H.B., & Pavel, M. (2016). Assessing executive function using a computer game: Computational modeling of cognitive processes. *Quantitative Methods.* Retrieved from: https://arxiv.org/abs/1603.03828[in English].

7. Kostić, V., & Sekulić, T. (2022). GeoGebra dynamic software as mathematical modeling support in engineering education. *Knowledge – International Journal,* 55(3), 461-467 [in English].

8. Kovtaniuk, M.S., Shokaliuk, S.V., & Stepanyuk, A.N. (2025). Game simulators as educational tools for developing algorithmic thinking skills in computer science education. *CTE Workshop Proceedings,* 12, 19-62 [in English].

9. Durkin, K., & Rittle-Johnson, B. (2015). Using video games to combine learning and assessment in mathematics education. *International Journal of Serious Games,* 2(4), 3-17 [in English].

10. Vlasyuk, A.P., & Martyniuk, P.M. (2017). Mathematical modeling and computer simulation of the filtration processes in earth dams. *Eastern-European Journal of Enterprise Technologies,* 2(10), 2-16 [in English].

11. Bottino, R.M., & Ott, M. (2006). Developing strategic and reasoning abilities with computer games at primary school level. *Computers & Education,* 49(4), 1272-1286 [in English].

12. Ke, F. (2007). Game and decision theory in mathematics education: epistemological, cognitive and didactical perspectives. *ZDM – Mathematics Education,* 39, 51-61 [in English].

13. Ziatdinov, R., & Valles, Jr.J.R. (2022). Synthesis of modeling, visualization, and programming in GeoGebra as an effective approach for teaching and learning STEM topics. *Mathematics,* 10(3), Article 398 [in English].

Received: August 20, 2025
Accepted: September 09, 2025