УДК 004.424:004.272:004.94

V. V. Gavrysh
G. V. Kuharchuk

HOW PROGRAMMING SIMULATES REAL-WORLD PHYSICAL PROCESSES

Vinnytsia National Technical University

Анотація

У роботі розглянуто підходи до симуляції фізичних процесів у програмуванні. Основну увагу приділено тому, як закони фізики реалізуються у вигляді алгоритмів та використовуються для моделювання руху об'єктів у програмному середовищі. Описано принципи роботи фізичних рушіїв та методи обчислення положення, швидкості і взаємодії об'єктів у реальному часі. Показано, що складні математичні моделі можуть бути спрощені та ефективно реалізовані у вигляді дискретних алгоритмів. Результати дослідження демонструють важливість програмних симуляцій у комп'ютерній графіці, відеоіграх та інших сучасних IT-застосуваннях.

Ключові слова: фізичне моделювання, програмування, симуляції, фізичні рушії, комп'ютерна графіка, алгоритми.

Abstract

This article examines approaches to simulating real-world physical processes in programming. The main focus is on how physical laws are implemented as algorithms and used to model the motion and interaction of objects in a software environment. The principles of physics engines and real-time calculations of position, velocity, and object interactions are described. It is shown that complex mathematical models can be simplified and efficiently implemented using discrete computational methods. The results highlight the importance of simulation techniques in modern applications such as computer graphics, video games, and other IT systems.

Keywords: physical simulation, programming, simulations, physics engines, computer graphics, algorithms.

Introduction

Modern software systems actively use simulation of physical processes to create realistic and interactive environments. Such simulations are widely applied in areas such as computer graphics, video games, virtual reality, and engineering applications. By reproducing the behavior of real-world objects, programming allows developers to create dynamic systems where motion, forces, and interactions are calculated automatically.

In practice, real-world physics is translated into algorithms that approximate the behavior of objects over time. Instead of solving complex mathematical equations directly, programmers use simplified numerical approaches that can be efficiently executed in real time. This makes it possible to simulate motion, collisions, and other physical effects within interactive applications.

Physics engines play a key role in this process, providing ready-made tools for handling object dynamics and interactions. These systems manage calculations of position, velocity, and forces, allowing developers to focus on higher-level design and functionality. Widely used platforms such as Unity [1] and Unreal Engine [2] include built-in physics systems that simplify the development of realistic simulations.

The purpose of this work is to explore how programming techniques are used to simulate real-world physical processes and to analyze the basic principles behind their implementation in software systems.

Research results

The study of physical simulation in programming shows that real-world processes can be effectively reproduced using simplified computational models. Instead of relying on complex analytical solutions, modern software systems approximate object behavior step by step, which allows simulations to run in real time.

At the core of most physical simulations lies the fundamental principle that force affects motion. This relationship is commonly expressed through Newton's second law:

$$F = ma, \tag{1}$$

In programming, this relationship is not solved analytically but is implemented through iterative calculations. The state of an object (its position and velocity) is updated at each time step using simple

numerical approximations [3]. A typical implementation can be represented as:

$$velocity = velocity + acceleration * dt,$$
$$position = position + velocity * dt. \quad (2)$$

This approach allows developers to simulate motion in a discrete manner, making it suitable for real-time applications such as video games and interactive simulations.

To demonstrate this, a simple model of an object moving under the influence of a restoring force was analyzed. The simulation shows that the position of the object changes smoothly over time, forming a curve that reflects oscillatory behavior. When damping is added, the motion gradually stabilizes, which corresponds to realistic physical behavior.

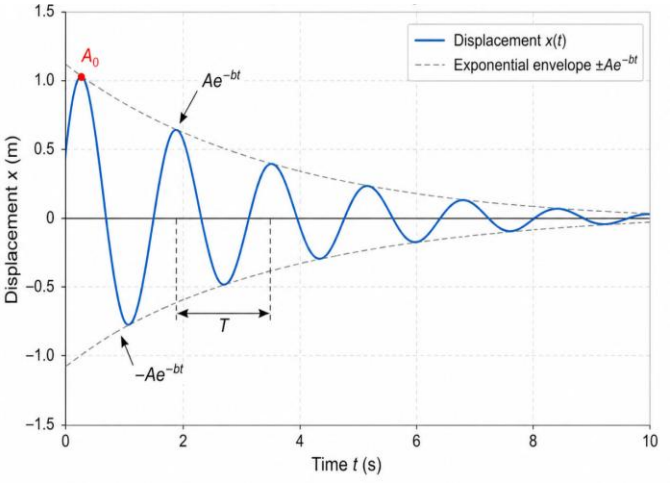

Figure 1. Damped motion

The results of the simulation can be visualized using graphs. A position-time graph shows how the object moves and slows down over time. These visualizations help to better understand the dynamics of the system and confirm the correctness of the implemented algorithm.

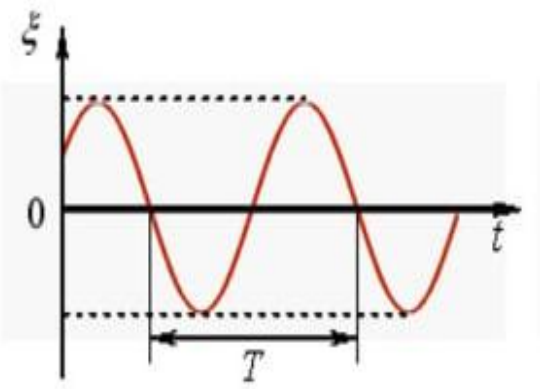

Figure 2. Simple harmonic motion

In addition, it was observed that simple numerical methods provide sufficient accuracy for real-time simulations, although more advanced techniques can improve stability and precision. This makes simplified algorithms widely used in modern physics engines, where performance is often more important than exact precision.

Conclusion

Simulation of physical processes is an essential component of modern software systems. By transforming real-world physics into computational algorithms, programming enables the creation of interactive and realistic environments used in various applications. The study shows that simplified numerical methods allow efficient real-time simulation of motion, forces, and object interactions. Physics engines further enhance this process by providing developers with ready-made tools that simplify implementation and improve productivity.

Overall, the integration of mathematical concepts and programming techniques makes it possible to develop complex dynamic systems that are both practical and widely used in modern technologies.

REFERENCES

1. Unity Technologies. URL: https://docs.unity3d.com/Manual/UnityManual.html (дата звернення: 22.04.2026).
2. Unreal Engine Physics. URL: https://dev.epicgames.com/documentation/unreal-engine/unreal-engine-5-7-documentation (дата звернення: 23.04.2026).
3. Millington I. Integration Basics. URL: https://gafferongames.com/post/integration_basics/ (дата звернення: 23.04.2026).

Гавриш Віталій Вікторович – студент групи 3ПІ-24Б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, м. Вінниця, e-mail: vetalik.gavrysh@gmail.com

Науковий керівник: ***Кухарчук Галина Вікторівна*** – викладач кафедри іноземних мов, Вінницький національний технічний університет, м. Вінниця, e-mail: galinakuh07@gmail.com

Gavrysh Vitaliy Viktorovych – student of group 3PI-24B, Faculty of Information Technologies and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: vetalik.gavrysh@gmail.com

Scientific supervisor: ***Kukharchuk Halyna Viktorivna*** – an Assistant Professor of Foreign Languages Department, Vinnytsia National Technical University, Vinnytsia, e-mail: galinakuh07@gmail.com