International Journal of Advanced Smart Convergence Vol.12 No.2 117-126 (2023)
http://dx.doi.org/10.7236/IJASC.2023.12.2.117

IJASC 23-2-14

Numerical Ballistic Modeling in Game Engines

YoungBo Go[1], YunJeong Kang[2]*

[1] Student, Department of Computer Software Engineering, Wonkwang University, Korea
[2] Professor, Department of Computer Software Engineering, Wonkwang University, Korea

gyb8592@wku.ac.kr, yjkang66@wku.ac.krediti

Abstract

To improve the overall performance and realism of your game, it is important to calculate the trajectory of a projectile accurately and quickly. One way to increase realism is to use a ballistic model that takes into account factors such as air resistance, density, and wind when calculating a projectile's trajectory. However, the more these factors are taken into account, the more computationally time-consuming and expensive it becomes, creating a trade-off between overall performance and efficiency. Therefore, we present an optimal solution to find a balance between ballistic model accuracy and computation time. We perform ballistic calculations using numerical methods such as Euler, Velocity Verlet, RK2, RK4, and Akima interpolation, and measure and compare the computation time, memory usage (RSS, Resident Set Size), and accuracy of each method. We show developers how to implement more accurate and efficient ballistic models and help them choose the right computational method for their numerical applications.

Keywords : *Numerical analysis, Interpolation, Ballistic modeling, Optimization*

1. INTRODUCTION

In the field of virtual reality technology, there has been a lot of recent research into mimicking realistic physical systems to make games more immersive and accurate simulations[1-5]. In particular, in systems that use large numbers of projectiles, such as FPS(First Person Shooter) or military equipment simulators, accurately and quickly calculating projectile trajectories is a key factor in improving performance, realism, and immersion[6-7].

To model the motion of a projectile, external factors such as air density, drag, and wind can be factored into the ballistic model to increase realism by reflecting external factors that affect the motion of the projectile[8-10]. However, these factors increase the complexity and cost of ballistic computations, creating a

Manuscript Received: April. 24, 2023 / Revised: April. 28, 2023 / Accepted: May. 2, 2023
Corresponding Author: yjkang66@wku.ac.kr
Tel: +82-63-850-6290, Fax: +82-63-850-5374
Professor, Department of Computer Software Engineering, Wonkwang University, Korea

trade-off between accuracy and computation time[11-12].

In this study, we use Python to create a ballistic model that accounts for air resistance and describes the projectile's equations of motion. We then apply the Euler, Velocity Verlet, RK2, RK4 methods and Akima interpolation to simulate the projectile and evaluate the computation time, process memory usage, and accuracy. In this way, we hope to contribute to the selection and use of ballistic calculation methods that take into account both realism and accuracy in various fields such as games, military training simulations, and the aerospace industry.

2. RELATED WORKS

Numerical analysis is the art of using computers to perform calculations on real-world systems or phenomena and obtain numerical solutions. It is mainly used in physical systems, engineering problems, and economics, but it is also used in ballistics to calculate the trajectory of a projectile. The trajectory of a projectile can be calculated theoretically using the equations of motion, but when external factors such as air resistance and wind are considered, nonlinearity occurs and the calculation becomes complex, making it difficult to obtain an accurate solution. This is where numerical simulations come in as a powerful tool to calculate projectile trajectories using approximated equations. The following is a description of the four numerical methods used in the study.

The Euler method is a simple and widely used method for solving differential equations numerically, which discretizes the differential equation to estimate the value at each step. Because the gradient of the current step is used to calculate the value of the next step, errors can be introduced and accumulated in the process of approximating the continuous function of the differential equation as a discrete function. This can lead to low accuracy for high-dimensional or nonlinear equations. Therefore, to get a more accurate approximation, you need to break up the calculation into smaller time intervals. Reducing the time interval can lead to more accurate results, but at the cost of increased computational time and cost.

$$\mathrm{y_{n+1} = y_n + v_n \times \Delta t} \qquad (1)$$

To overcome the limitations of the Euler method, we used the Velocity Verlet method, which is widely used in physical simulations. The Euler method has the problem that the error accumulates even when the time interval is set to small. To overcome this problem, the Velocity Verlet method takes into account velocity and acceleration and can simulate physical systems with smaller errors than the Euler method.

$$\mathrm{y_{n+1} = y_n + v_n \times \Delta t + 0.5 * a_n \times \Delta t^2}$$
$$\mathrm{v_{n+1} = v_n + 0.5 \times (a_n + a_{n+1}) * \Delta t} \qquad (2)$$

RK2 uses two steps to approximate the differential equation. It computes an intermediate position from the current position and the next position, and computes the gradient from the intermediate position to calculate the final next position.

$$k_1 = \mathrm{f(t_n, y_n)}$$
$$k_2 = \mathcal{F}(\mathrm{t_n + \Delta t, y_n} + k_1 \times \Delta \mathrm{t})$$
$$\mathrm{y_{n+1} = y_n + \Delta t} * (k_1 + k_2)/2 \qquad (3)$$

The RK4 method performs additional calculations to get a more accurate position estimate than the RK2 method. The RK4 method calculates intermediate positions twice and utilizes the gradient at each intermediate position to calculate the next position. Among the numerical methods, the RK4 method is the most commonly used because of its high accuracy, which is usually close to the solution, but it is characterized by a relatively long computation time and considerable computational cost. Therefore, the results of Euler, Verlet, and RK2 calculations are compared with the results of the RK4 method that minimizes the time interval to evaluate the accuracy and computational cost.

$$\begin{gathered} k_1 = \mathrm{f}(\mathrm{t_n}, \mathrm{y_n}), \\ k_2 = \mathrm{f}(\mathrm{t_n} + \Delta \mathrm{t}/2, \mathrm{y_n} + k_1 * \Delta \mathrm{t}/2), \\ k_3 = \mathrm{f}(\mathrm{t_n} + \Delta \mathrm{t}/2, \mathrm{y_n} + k_2 \times \Delta \mathrm{t}/2), \\ k_4 = \mathrm{f}(\mathrm{t_n} + \Delta \mathrm{t}, \mathrm{y_n} + \mathrm{k3} \times \Delta \mathrm{t}), \\ \mathrm{y_{n+1}} = \mathrm{y_n} + \Delta \mathrm{t} \times (k_1 + 2 * (k_2 + k_3) + k_4)/6 \end{gathered} \tag{4}$$

Interpolation is a method of comparing neighboring data to fill in missing values or to smoothly interpolate data to estimate new values. While there are many different types of interpolation, Akima interpolation uses a local approach to approximate the value of a function by only considering points that lie close to the curve between data points. Akima interpolation is very fast to compute, can accurately estimate the value of the approximating function if the data points are evenly spaced, and can minimize overshoot. Due to the nature of ballistic calculations, parabolic points are almost evenly spaced, so Akima interpolation was used, and the time interval $\Delta \mathrm{t}$ was increased to generate discrete parabolic points to reduce the time and cost of ballistic calculations. However, the interpolation method itself has some computational cost, and the error of the calculated estimate is different for each type of interpolation method, so it is necessary to select the appropriate interpolation method for the situation to improve efficiency.

3. PROPOSAL MODEL

This study deals with out-of-river ballistics, and the model is simplified for ease of implementation in games. For this purpose, complex factors such as projectile rotation, flow phenomena in the surrounding air, changes in air density and gravitational acceleration with altitude, and the Coriolis Effect were excluded. In addition, to simplify the modeling of the interaction with the air, the projectile is assumed to be in the form of a bullet, and the area of contact with the air is determined by the diameter and mass of the projectile. To realize the projectile's motion, data such as gravitational acceleration, projectile area, mass, air resistance coefficient, initial position, and velocity are essential.

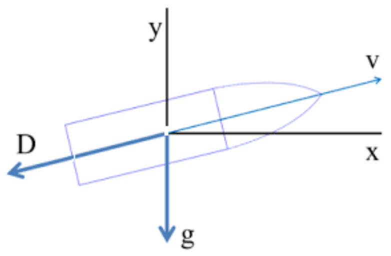

Figure 1. Drag and gravity acting on the bullet

In this study, we conducted experiments with the ACF-3 ballistic model that implements the movement of shells in Garry's Mod, one of the games based on the Source engine. The Source engine is a widely used engine in game development, making it a good candidate for applying the basic principles and characteristics of ballistics. The Source engine uses two scales to accurately handle information such as the size of maps, buildings, and models in the game, the speed of projectiles, and collision handling. Architectural Scale represents the scale of elements such as terrain, buildings, and specific models, and is set to 0.01905 meters for 1 Unit. Entity Scale is used to scale most characters and certain models, and uses inches at 0.0254 meters per unit. This unit system allows the game engine to effectively handle different elements, improving the quality of game development and providing a more immersive experience for players.

Table 1. Ballistic model parameters

Parameter	Meaning
$g = 600$	gravity (g)
$r = 60$	diameter (r)
$m = 10$	projectile mass (m)
$d = 40$	drag divisor (d)
$v_x = 50$	horizontal velocity (vx)
$v_y = 50$	vertical velocity (vy)
gv = (0, -g)	vector of gravity (gv)
k = (π *r^2)/(4000000*m*d)	drag coefficient (k)
p = (0, 0)	initial location (p)
v = (vx, vy)*39.3701	initial velocity (v)

The ballistic model is shown in Eq. 5.

$$\begin{gathered} \mathrm{drag} = \mathrm{k} * \mathrm{v}^2 \\ \mathrm{accel} = (\mathrm{g} - \mathrm{drag}) * \Delta\mathrm{t} \\ \mathrm{p} = \mathrm{p}^0 + (\mathrm{v} + 0.5 * \mathrm{accel}) * \Delta\mathrm{t} \\ \mathrm{v} = \mathrm{v}^0 + \mathrm{accel} \end{gathered} \tag{5}$$

k is the drag coefficient, g is the gravitational acceleration, p and v are the following positions and velocities, respectively. Air resistance and acceleration are calculated according to Δt, reflected in p and v, and updated to the next step. Air resistance is proportional to the square of the velocity and acts in the opposite direction of the projectile's motion. The ballistic model thus created determines the motion state of the projectile according to initial conditions. Through this calculation formula, the trajectory and motion of the projectile can be implemented.

4. EXPERIMENT

We used Python's Numpy library to generate a ballistic model that takes into account the initial conditions of the projectile in two-dimensional space and secondary air resistance. For the ballistic model, we performed 100 ballistic calculations using the Euler, Berrett, RK2, and RK4 methods and the Akima interpolation. For each method, 100 ballistic calculations are performed, and the calculation time and memory usage are averaged using the time and psutil libraries. The accuracy of the calculation results and the calculation time depend on the characteristics of each method, and the optimal method is selected by comparing with the results calculated by RK4. The experiments were conducted on an Intel Core i7-9700KF processor and an NVIDIA GeForce

RTX 2070 graphics card.The following are the results of the computation time and memory usage experiments.

Table 2. Process Memory Resident Set Size (RSS, MB)

	0.1	0.01	0.001	0.0001
RK2	50.456252	50.59414	51.09414	52.551173
RK4	50.439062	50.56250	51.16171	52.361722
Euler	50.388280	50.62578	51.11328	52.420314
Verlet	50.433594	50.60390	51.08359	52.515624

Table 3. Calculation time (Sec) table

	0.1	0.01	0.001	0.0001
RK2	0.00100769996	0.00817556379	0.088949608797	1.41097750665
RK4	0.00139641762	0.01266536712	0.12964167594	1.84817433358
Euler	0.00100207328	0.00559225082	0.05983428956	1.13937988284
Verlet	0.00101008416	0.00649733544	0.0694647789	1.21266651154

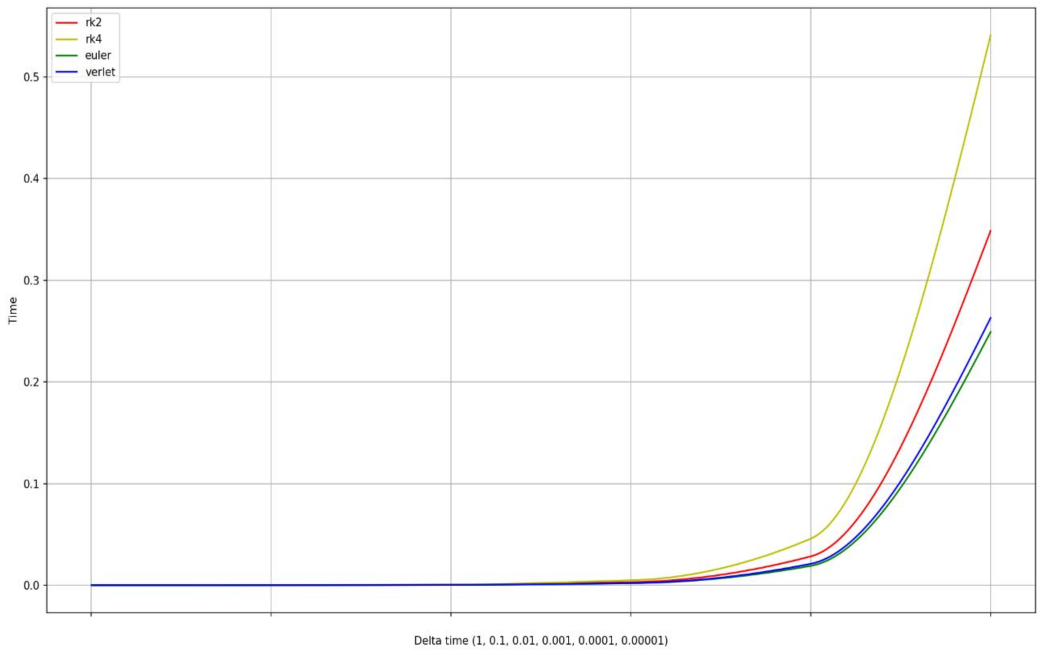

Figure 2. Increase in Delta time vs speed

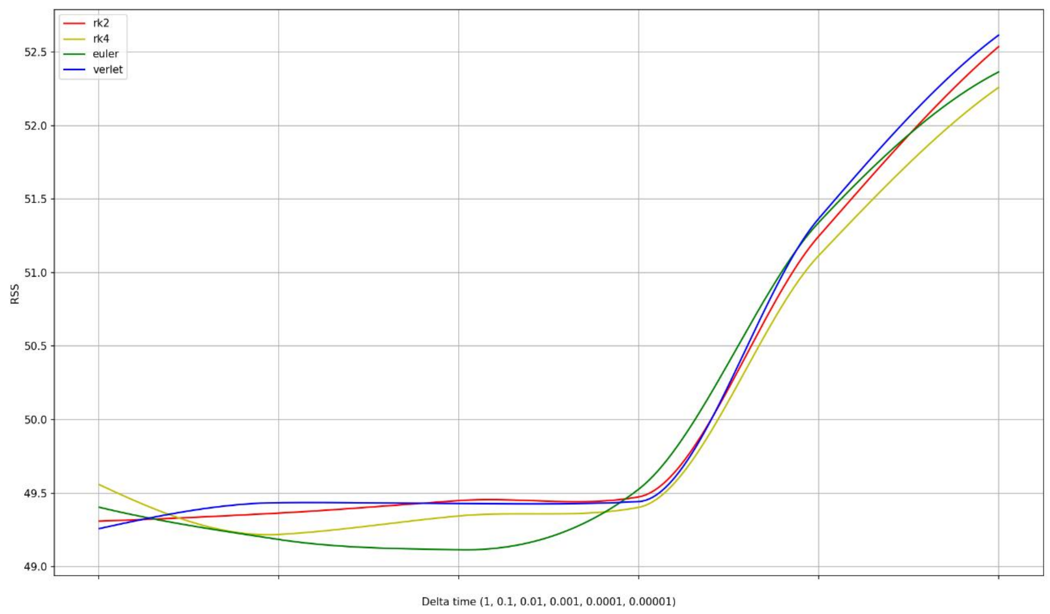

Figure 3. Increase in Delta time vs memory usage

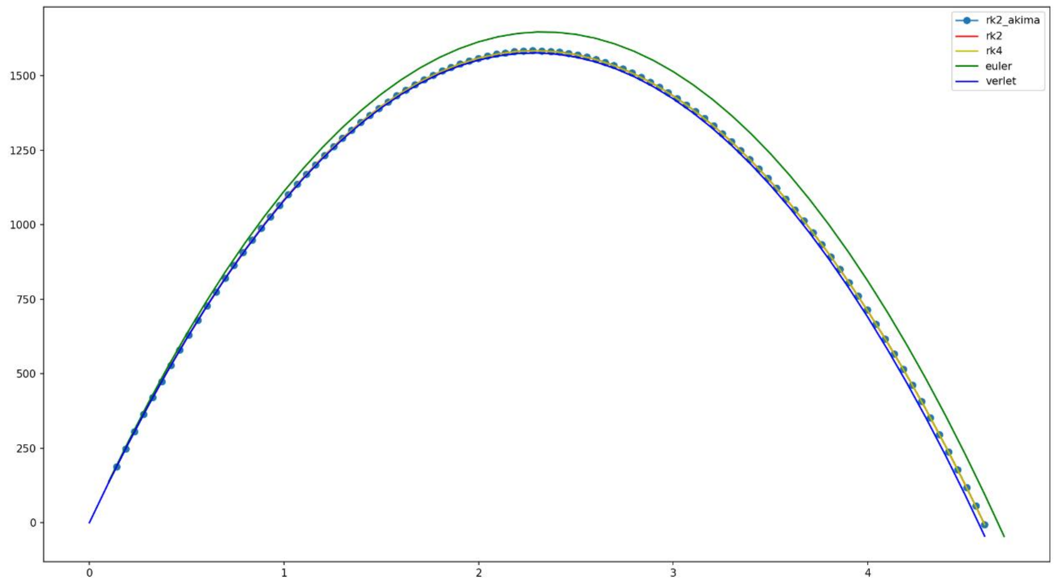

Figure 4. Parabola graph calculated by each method

The RK4 method required more computation time than the other methods. While this can vary depending on how the algorithm is implemented, the general rule of thumb is that more computation tends to lead to higher accuracy. Since the RK4 method performs more sophisticated calculations, it has the highest accuracy, which means it is more computationally expensive. When comparing the RK2 method to the Velocity Verlet method, the RK2 method takes longer in terms of computation time, but produces results that are closer to the RK4 method. This is because the RK2 method has a similar computational approach to the RK4 method. Also, when t is set to 1 in the RK2 method, the error is similar to that of Verlet and Euler's results when t is set to 0.0001.

Figure 4 compares how similar the parabola computed with each method is to the parabola computed with RK4, and the results are shown in Figure 5, showing the accuracy of the parabola computed with each method.

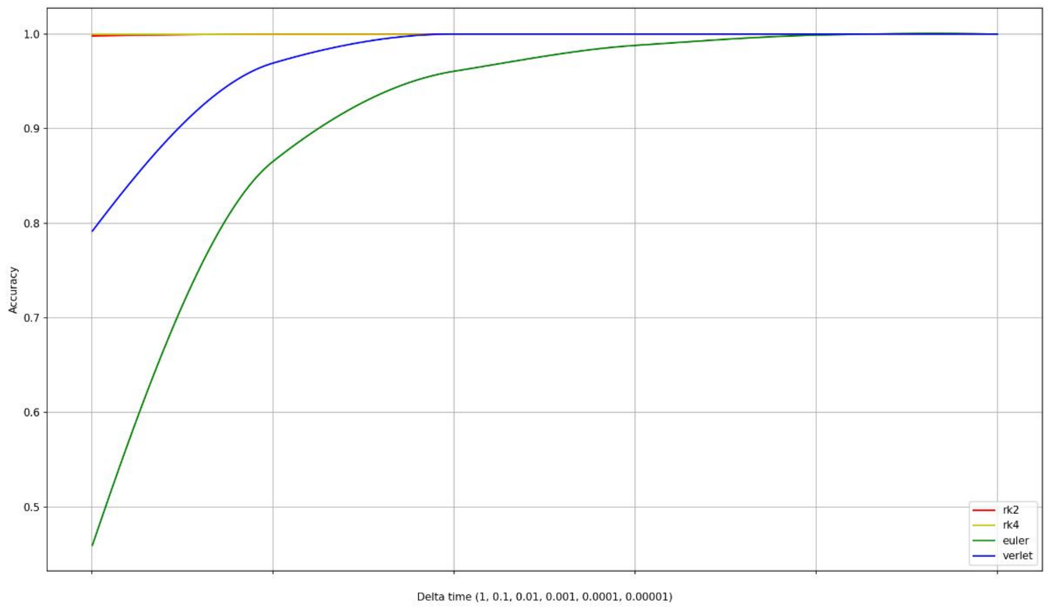

Figure 5. Parabolic accuracy compared to RK4

We also show graphs in Figures 6, 7, and 8 to evaluate the overall performance, considering memory usage, computation time, and parabolic accuracy compared to RK4.

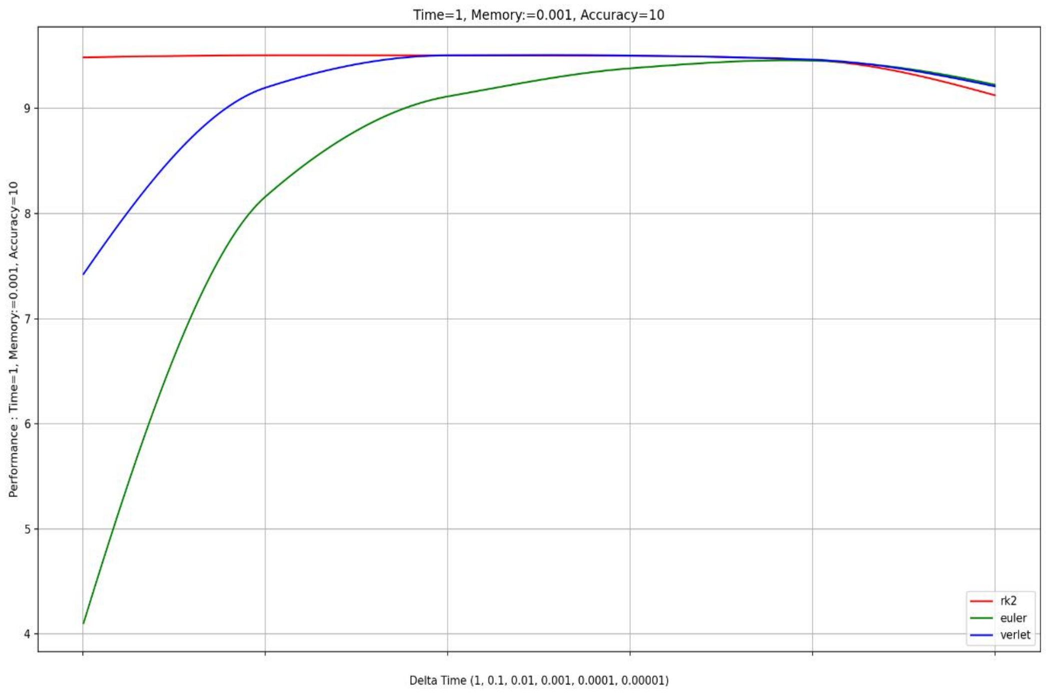

Figure 6. Performance Graph (Time=1, Memory=0.001, Accuracy=10)

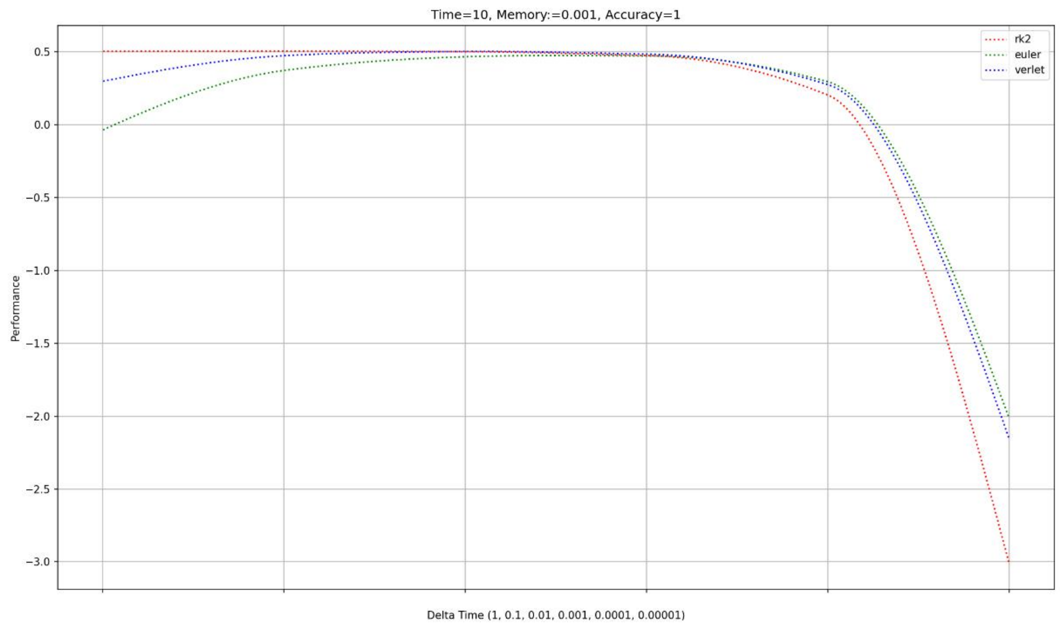

Figure 7. Performance Graph (Time=10, Memory=0.001, Accuracy=10)

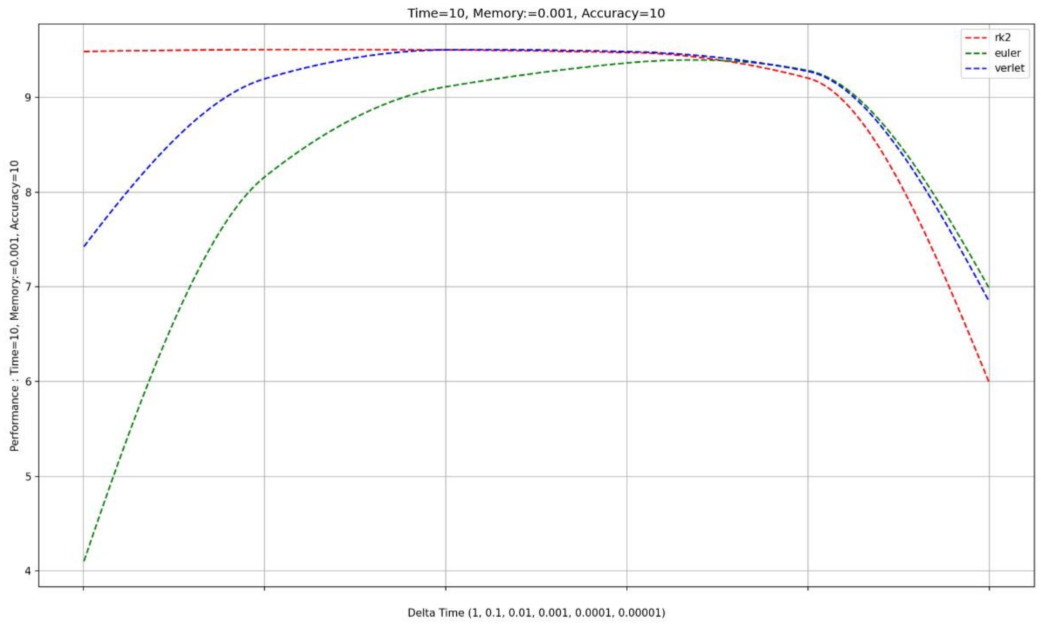

Figure 8. Performance Graph (Time=10, Memory=0.001, Accuracy=1)

The performance calculation is shown in Eq. 6.

$$\text{Perfomance} = -(\text{CalculationTime} \times \text{w1} + \text{MemoryUsage} \times \text{w2}) + \text{Accuracy} \times \text{w3} \quad (6)$$

w1, w2, and w3 are weighted values for computation time, memory usage, and accuracy, respectively, and each weight can be adjusted to set the importance of the ballistic calculation factor to the game. Since the memory usage of each numerical solution is roughly the same, we used a weight of 0.001 for memory usage and weights of 1 and 10 for computation time and accuracy. The RK2 method outperformed in all respects, and the performance score was calculated in the order of Verlet followed by Euler. To verify that the performance of the ballistic calculation in Python is similar to the performance of the ballistic calculation in the actual game, we implemented the ballistic calculation in the game.

Below is the ballistic calculation implemented in Python and applied to a three-dimensional game based on the Source engine.

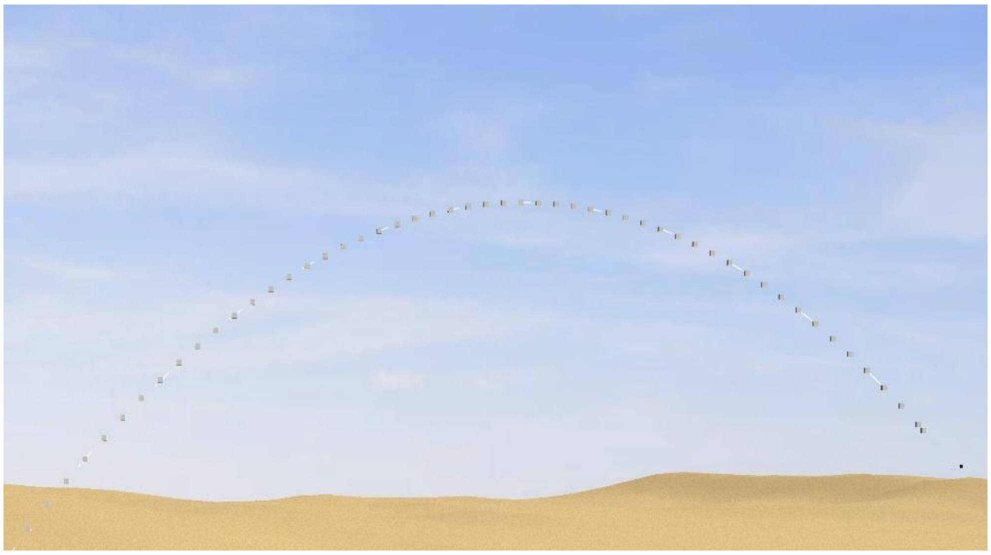

Figure 9. Simulation implemented in the game

The white cubes in the figure are parabolic coordinates computed with RK4, and the thin object between the cubes is a projectile in flight computed with Eq. 5. The projectile calculated by Eq. 5 is accurately penetrating the cube calculated by RK4. The game has a tick rate of 66/s, which is the reason for the difference in calculation time between the ballistic calculation implemented in the Python environment and the game implemented in Lua. The ballistic calculation implemented in the game took longer, and the computational cost was similar to the results of the paper.

5. CONCLUSION

The RK4 method has the highest accuracy but is computationally time-consuming, while the RK2 method has higher accuracy than the Euler and Velocity Verlet methods and is very computationally efficient. This suggests that the RK2 method can increase the efficiency of ballistic calculations. However, the limitations of this study are that we did not perform ballistic calculations in various environments, and there are many numerical analysis methods other than Euler, Velocity Verlet, RK2, and RK4. The computational time and memory usage may vary significantly due to different hardware performance and resource management methods in different computer environments, but the computational efficiency of each method was similar to the results of this paper. Further experiments and verification should be conducted to further clarify the applicability of this research to other environments.

ACKNOWLEDGEMENT

This paper was supported by Wonkwang University in 2022.

REFERENCES

[1] H.S. Kim, T.J. Park, "Technology Trends in Virtual Reality (VR) and Augmented Reality (AR) and Implementation Cases in Game Engines".Journal of the Korean Communications Society(Information and Communication), Vol.33, No.12, pp.56-62, Dec 2016.

[2] K. Hullett, The science of level design: Design patterns and analysis of player behavior in first-person shooter levels. University of California, Santa Cruz, 2012.

[3] Karavolos, Konstantinos Daniel. "Orchestrating the generation of game facets via a model of gameplay," 2020. https://www.um.edu.mt/library/oar/handle/123456789/70321

[4] schertler. ruben. kriglstein simone. wallner. Günter, "User guided movement analysis in games using semantic trajectories," In: Proceedings of the Annual Symposium on Computer-Human Interaction in Play, pp. 613-623, Oct 2019. https://doi.org/10.1145/3311350.3347156

[5] K. Jędrasiak, "Advanced Ballistic Model and Its Experimental Evaluation for Professional Simulation Systems," Advanced Technologies in Practical Applications for National Security, pp. 195-228, Sep 2017. https://doi.org/10.1007/978-3-319-64674-9_12

[6] K. I. Kim, "Experimental Implementation of Digital Twin Simulation for Physical System Optimization," Journal of Convergence for Information Technology, Vol.11, No.4, pp.19-25, Nov 2021.
DOI : 10.22156/CS4SMB.2021.11.04.019

[7] S. H. Chung, J. S. Kim, D.H. Song, "A Comparative Analysis of First Person Shooter Games on Battle Style and Equipment/Skill Patterns-Overwatch vs AVA Online vs Battleground," In Proceedings of the Korean Institute of Information and Communication Sciences Conference, The Korea Institute of Information and Communication Engineering, pp. 443-446, Oct 2017.
http://koreascience.or.kr/article/CFKO201714956117213.page?&lang=ko

[8] C. S. Oh, Y. K. Park, S. J. Jo, B. C. Sun. "Lift-off Dynamic Modeling and Clearance Analysis of Test Launch Vehicle ," in Proc. KSAS 2018 Fall Conference, pp. 964-965, Nov 2018.
https://www.dbpia.co.kr/pdf/pdfView.do?nodeId=NODE07620020&googleIPSandBox=false&mark=0&ipRange=false&accessgl=Y&language=ko_KR&hasTopBanner=true

[9] J. B, Song, P. Kang, "Dynamic Analysis of the Turret for Analyzing the Accuracy Impact Factor of the Ground Combat Vehicle." Transactions of the Society of CAD/CAM Engineers, Society for Computational Design and Engineering, Vol. 19, No. 4, pp. 340–346, Dec 2014.
DOI : 10.7315/cadcam.2014.340

[10] S. I. Lee, S. B. Cho, B. C. Sun, "Flight Test Trajectory Data of Test Launch Vehicle," in Proc. KSAS 2019 Spring Conference, pp. 319-320, Apr 2019.

[11] L.S. Kim, S.J. Kim, D. W. Lee, H.H. Bang, "Modeling of Landing Gears in 6 Degrees of Freedom Environment for Simulation of Fixed Wing Aircraft Landing Performance," in Proc. KSAS 2021 Spring Conference, pp. 780-781, Jul 2021.

[12] J.H. Jin, D. H. Han, J. H. Jin. "Conceptual Configuration Design of Short Range Ballistic Missiles by Using Multidisciplinary Design Optimization Approach," Journal of the Korean Society for Aeronautical & Space Sciences, Vol. 47, No. 3, pp. 228-239, Mar 2019.
DOI : 10.5139/JKSAS.2019.47.3.228

[13] H.Y. Jung, N. H. Kim, K.J. Park, "Pressure Analysis and Conceptual Design for Indoor Ballistic Test Range by Numerical Methods,". Journal of the KIMST, Vol. 20, No.1, pp. 55-62, Feb 2017.
DOI : 10.9766/KIMST.2017.20.1.055