Fast simulations in augmented reality

Mateusz Ksyta[1], Wojciech Kordylewski[1], Marcin Łoś[1], Piotr Gurgul[2], Maciej Sikora[1], and Maciej Paszyński[1]

[1] AGH University of Krakow, Poland
[2] Snap Switzerland GmbH
paszynsk@agh.edu.pl

Abstract. Augmented reality may soon revolutionize the world we live in. The incorporation of computer simulations into augmented reality glasses opens new perspectives for the perception of reality. In this paper, we investigate the possibility of performing numerical simulations in real time within augmented reality glasses. We present the technology that can be successfully employed in the real-life simulations of the Partial Differential Equations (PDE) based phenomena. We designed and implemented a two- and three-dimensional explicit dynamics solver in Lens Studio using Finite Difference Method (FDM) on the augmented reality glasses. We performed tests on the computational cost, memory usage, and the capability of performing real-life simulations of advection-diffusion and wave propagation problems.

Keywords: augmented reality · finite difference method · real time simulations · three-dimensional advection-diffusion equations · two-dimensional wave equations · lens studio

1 Introduction

We consider a fast solver allowing for real-life simulations of physical phenomena described by Partial Differential Equations (PDEs) in augmented reality of Lens studio [1]. The solver employs the explicit dynamics and Finite Difference Method (FDM) algorithm [6–8]. The figures 1-2 present screenshots from simulations that have been rendered as Lenses in the Snapchat App. However, these lenses can also run on Snap Spectacles, which are Snap's augmented reality glasses. Our goal is to develop algorithms that can perform simulations in real-time using the SnapChat rendering engine of augmented reality glasses. The proposed algorithm, a finite difference method with an Euler time integration step scheme, has a linear computational complexity of $\mathcal{O}(N)$ where N is the number of spatial points in which we recalculate the state of the modeled phenomenon. Similarly, the memory complexity of the proposed algorithm is linear $\mathcal{O}(N)$ with respect to the number of spatial points processed, assuming that we store the state of spatial points from the current (N points) and previous (N points) time instants. Specifically, the proposed algorithms have been tested on two- and three-dimensional computational problems, such as

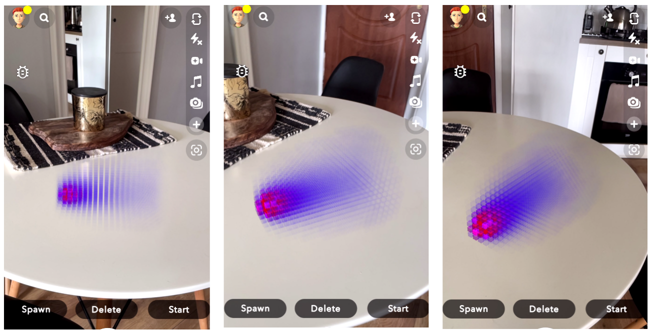

Fig. 1: Three-dimensional simulations of the advection-diffusion equations in augmented reality.

- the phenomenon of advection-diffusion in three dimensions (spreading of substances, e.g., smoke in the air by means of the phenomenon of diffusion and advection modeling air movements in the room)
- the phenomenon of wave propagation in two dimensions (to illustrate the possibility of visualizing the simulation on a virtual plane and the possibility of observing the course of the simulation by researchers using augmented reality glasses).

The work investigated the maximum size of the computational domain in two- and three-dimensions in which real-time simulations could be carried out using the fastest possible algorithm with linear complexity. The simulations performed by ANSYS with the use of the finite element method has been projected into the augmented reality [2]. The finite element method loading is based on the data provided by the sensors detecting the surrounding structures [2], and the computed results are displayed in the augmented reality. Another paper [3] projects the previously computed finite element method results onto real structures in augmented reality using the Microsoft Holo Lens. In our paper, for the first time, we present fast real-life computations with the finite difference method using a much lighter Snapchat Lens, allowing for integrating the lens with normal life activities without using any external sensors or servers. There are also some preliminary attempts to visualize the computational results of the finite element method computations performed offline on the real models using augmented reality [4, 5].

2 Advection-diffusion problem

We start from the formulation of the advection-diffusion solver

$$\begin{aligned}\frac{\partial c(x,y,z,t)}{\partial t} - \varepsilon_x \frac{\partial^2 c(x,y,z,t)}{\partial x^2} - \varepsilon_y \frac{\partial^2 c(x,y,z,t)}{\partial y^2} - \varepsilon_z \frac{\partial^2 c(x,y,z,t)}{\partial z^2} \\ + b_x \frac{\partial c(x,y,z,t)}{\partial x} + b_y \frac{\partial c(x,y,z,t)}{\partial y} + b_z \frac{\partial c(x,y,z,t)}{\partial z} = f(x,y,z,t)\end{aligned} \quad (1)$$

where we seek the scalar concentration field $c(x, y, z, t)$; here, the vector (b_x, b_y, b_z) is the advection vector, denoting the wind blowing in the domain, are the diffusion coefficients along the x, y, z-axis of the coordinate system. We introduce the explicit dynamics time integration scheme, where we compute the values of $c(x, y, z, t + \Delta t)$ based on the previous time step configuration $c(x, y, z, t)$:

$$\begin{gathered}\frac{c(x,y,z,t+\Delta t)-c(x,y,z,t)}{\Delta t}-\varepsilon_x\frac{\partial^2 c(x,y,z,t)}{\partial x^2}-\varepsilon_y\frac{\partial^2 c(x,y,z,t)}{\partial y^2}\\ -\varepsilon_z\frac{\partial^2 c(x,y,z,t)}{\partial z^2}+b_x\frac{\partial c(x,y,z,t)}{\partial x}+b_y\frac{\partial c(x,y,z,t)}{\partial y}+b_z\frac{\partial c(x,y,z,t)}{\partial z}=\\ f(x,y,z,t)\end{gathered}\tag{2}$$

$$\begin{gathered}c(x,y,z,t+\Delta t)=c(x,y,z,t)\\ +\Delta t\varepsilon_x\frac{\partial^2 c(x,y,z,t)}{\partial x^2}+\Delta t\varepsilon_y\frac{\partial^2 c(x,y,z,t)}{\partial y^2}+\Delta t\varepsilon_z\frac{\partial^2 c(x,y,z,t)}{\partial z^2}\\ -\Delta t b_x\frac{\partial c(x,y,z,t)}{\partial x}-\Delta t b_y\frac{\partial c(x,y,z,t)}{\partial y}-\Delta t b_z\frac{\partial c(x,y,z,t)}{\partial z}\\ +\Delta t f(x,y,z,t)\end{gathered}\tag{3}$$

We introduce the three-dimensional mesh with points (x_i, y_j, z_k), $i = 1, \ldots, N_x$, $j = 1, \ldots, N_y$, $k = 1, \ldots, N_z$, and we write down the equations in the nodes of the mesh

$$\begin{gathered}c(x_i,y_j,z_k,t+\Delta t)=c(x_i,y_j,z_k,t)\\ +\Delta t\varepsilon_x\frac{\partial^2 c(x_i,y_j,z_k,t)}{\partial x^2}+\Delta t\varepsilon_y\frac{\partial^2 c(x_i,y_j,z_k,t)}{\partial y^2}+\Delta t\varepsilon_z\frac{\partial^2 c(x_i,y_j,z_k,t)}{\partial z^2}\\ -\Delta t b_x\frac{\partial c(x_i,y_j,z_k,t)}{\partial x}-\Delta t b_y\frac{\partial c(x_i,y_j,z_k,t)}{\partial y}-\Delta t b_z\frac{\partial c(x_i,y_j,z_k,t)}{\partial z}\\ +\Delta t f(x_i,y_j,z_k,t)\end{gathered}\tag{4}$$

We approximate the first and the second derivatives using the central finite differences

$$\begin{gathered}c(x_i,y_j,z_k,t+\Delta t)=c(x_i,y_j,z_k,t)\\ +\Delta t\varepsilon_x\frac{c(x_{i-1},y_j,z_k,t)-2c(x_i,y_j,z_k,t)+c(x_{i+1},y_j,z_k,t)}{\Delta x^2}\\ +\Delta t\varepsilon_y\frac{c(x_i,y_{j-1},z_k,t)-2c(x_i,y_j,z_k,t)+c(x_i,y_{j+1},z_k,t)}{\Delta y^2}\\ +\Delta t\varepsilon_z\frac{c(x_i,y_j,z_{k-1},t)-2c(x_i,y_j,z_k,t)+c(x_i,y_j,z_{k+1},t)}{\Delta z^2}\\ -\Delta t b_x\frac{c(x_{i+1},y_j,z_k,t)-c(x_{i-1},y_j,z_k,t)}{2\Delta x}-\Delta t b_y\frac{c(x_i,y_{j+1},z_k,t)-c(x_i,y_{j-1},z_k,t)}{2\Delta y}\\ -\Delta t b_z\frac{c(x_i,y_j,z_{k+1},t)-c(x_i,y_j,z_{k-1},t)}{2\Delta z}+\Delta t f(x_i,y_j,z_k,t)\end{gathered}\tag{5}$$

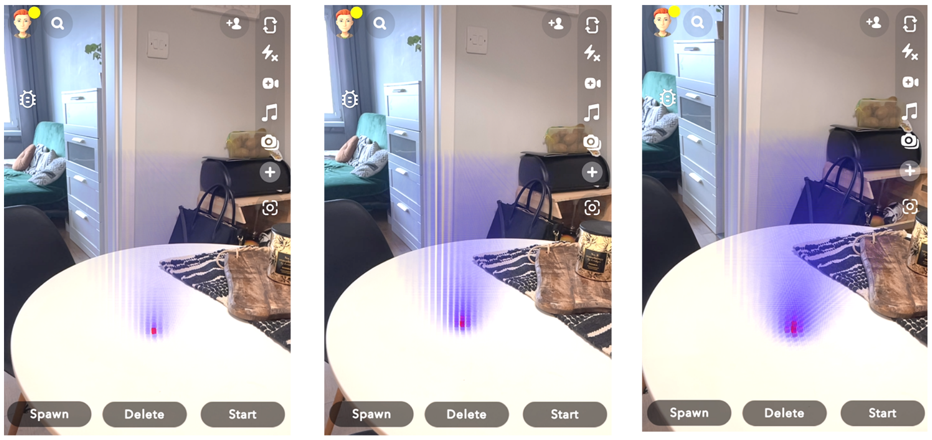

Fig. 2: Another simulation of three-dimensional advection-diffusion in augmented reality

The initial state is the zero concentration, the problem is driven by non-zero force component at a given point of the mesh, and the problem is modeled with free open boundary.

3 Wave equations

We start from the formulation of the wave equation solver

$$\begin{aligned}\frac{\partial^2 u(x,y,t)}{\partial t^2} - \frac{\partial}{\partial x}\left(g(u(x,y,t)-b(x,y))\frac{\partial u(x,y,t)}{\partial x}\right) \\ -\frac{\partial}{\partial y}\left(g(u(x,y,t)-b(x,y))\frac{\partial u(x,y,t)}{\partial y}\right) = f(x,y,t)\end{aligned} \tag{6}$$

where we seek the water level $u(x,y,t)$. Here, g denotes the acceleration due to gravity $g = 9.81$, $b(x,y)$ is given water bed. The initial condition is the shape of the initial wave, and the problem is modeled with free open boundary. We introduce the explicit dynamics time integration scheme, where we compute the values of $u(x,y,t+2\Delta t)$ based on the two previous time step configurations $u(x,y,t+\Delta t)$, $u(x,y,t)$

$$\begin{aligned}\frac{u(x,y,t+2\Delta t)-2u(x,y,t+\Delta t)+u(x,y,t)}{\Delta t^2} \\ -\frac{\partial}{\partial x}\left(g(u(x,y,t)-b(x,y))\frac{\partial u(x,y,t)}{\partial x}\right) \\ -\frac{\partial}{\partial y}\left(g(u(x,y,t)-b(x,y))\frac{\partial u(x,y,t)}{\partial y}\right) = f(x,y,t)\end{aligned} \tag{7}$$

We rewrite the equation to emphasize that the new state $u(x, y, t+2\Delta t)$ is given by the update of the previous state $u(x, y, t+\Delta t)$, the estimation of the wave velocity computed based on the last two time steps $[u(x, y, t+\Delta t) - u(x, y, t)]$ and the physics of the wave propagation phenomena (all the remaining terms)

$$\begin{aligned} u(x,y,t+2\Delta t) = u(x,y,t+\Delta t) + [u(x,y,t+\Delta t) - u(x,y,t)] \\ +\Delta t^2 \frac{\partial}{\partial x}\left(g(u(x,y,t) - b(x,y)) \frac{\partial u(x,y,t)}{\partial x} \right) \\ +\Delta t^2 \frac{\partial}{\partial y}\left(g(u(x,y,t) - b(x,y)) \frac{\partial u(x,y,t)}{\partial y} \right) + \Delta t^2 f(x,y,t) \end{aligned} \tag{8}$$

We introduce the dumping constant in front of the difference $C(u(x, y, t+\Delta t) - u(x, y, t))$, to emphasize that the wave velocity is dumped due to internal forces

$$\begin{aligned} u(x,y,t+2\Delta t) = u(x,y,t+\Delta t) + C(u(x,y,t+\Delta t) - u(x,y,t)) \\ +\Delta t^2 \frac{\partial}{\partial x}\left(g(u(x,y,t) - b(x,y)) \frac{\partial u(x,y,t)}{\partial x} \right) \\ +\Delta t^2 \frac{\partial}{\partial y}\left(g(u(x,y,t) - b(x,y)) \frac{\partial u(x,y,t)}{\partial y} \right) + \Delta t^2 f(x,y,t) \end{aligned} \tag{9}$$

We introduce the two-dimensional mesh with points (x_i, y_j), $i = 1, \ldots, N_x$, $j = 1, \ldots, N_y$ and we write down the equations in the nodes of the mesh. We also assume a flat water bed $b(x, y, t) = 0$

$$\begin{aligned} u(x_i,y_j,t+2\Delta t) = u(x_i,y_j,t+\Delta t) + C(u(x_i,y_j,t+\Delta t) - u(x_i,y_j,t)) \\ +\Delta t^2 \frac{\partial}{\partial x}\left(g u(x_i,y_j,t) \frac{\partial u(x_i,y_j,t)}{\partial x} \right) + \Delta t^2 \frac{\partial}{\partial y}\left(g u(x_i,y_j,t) \frac{\partial u(x_i,y_j,t)}{\partial y} \right) \\ +\Delta t^2 f(x,y,t) \end{aligned} \tag{10}$$

We expand the derivatives

$$\begin{aligned} u(x_i,y_j,t+2\Delta t) = u(x_i,y_j,t+\Delta t) + C(u(x_i,y_j,t+\Delta t) - u(x_i,y_j,t)) \\ +\Delta t^2 \left[\left(g \frac{\partial u(x_i,y_j,t)}{\partial x} \frac{\partial u(x_i,y_j,t)}{\partial x} \right) + \left(g u(x_i,y_j,t) \frac{\partial^2 u(x_i,y_j,t)}{\partial x^2} \right) \right] \\ +\Delta t^2 \left[\left(g \frac{\partial u(x_i,y_j,t)}{\partial y} \frac{\partial u(x_i,y_j,t)}{\partial y} \right) + \left(g u(x_i,y_j,t) \frac{\partial^2 u(x_i,y_j,t)}{\partial y^2} \right) \right] \\ +\Delta t^2 f(x,y,t) \end{aligned} \tag{11}$$

and we approximate the first and the second derivatives using the finite differences,

$$\begin{aligned} u(x_i, y_j, t+2\Delta t) &= u(x_i, y_j, t+\Delta t) + C(u(x_i, y_j, t+\Delta t) - u(x_i, y_j, t)) \\ &+\Delta t^2 \left[\begin{array}{c} \left(g \left(\frac{u(x_{i+1}, y_j, t+\Delta) - u(x_{i-1}, y_j, t+\Delta)}{2\Delta x} \right)^2 \right) + \\ \left(g u(x_i, y_j, t+\Delta t) \frac{u(x_{i+1}, y_j, t+\Delta) - 2u(x_i, y_j, t+\Delta t) + u(x_{i-1}, y_j, t+\Delta)}{\Delta x^2} \right) \end{array} \right] \\ &+\Delta t^2 \left[\begin{array}{c} \left(g \left(\frac{u(x_i, y_{j+1}, t+\Delta) - u(x_i, y_{j-1}, t+\Delta)}{2\Delta y} \right)^2 \right) + \\ \left(g u(x_i, y_j, t+\Delta t) \frac{u(x_i, y_{j+1}, t+\Delta) - 2u(x_i, y_j, t+\Delta t) + u(x_i, y_{j-1}, t+\Delta)}{\Delta y^2} \right) \end{array} \right] \\ &+\Delta t^2 f(x, y, t) \end{aligned} \tag{12}$$

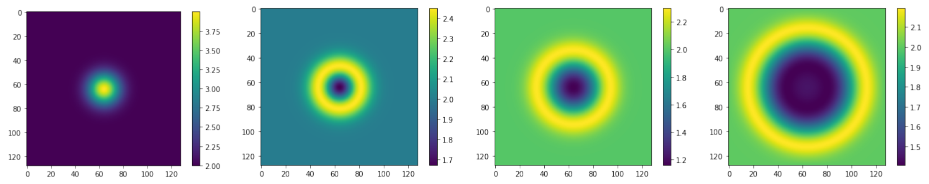

Fig. 3: Verification of the two-dimensional wave propagation simulations by Python code

4 Implementation

The whole project for implementing 2D or 3D simulation and visualization contains three files:

- WorldMeshController.js - the script which is responsible for controlling the simulation. It allows the definition of the dimensions of the computational mesh, the number of time steps, sources, the diffusion coefficients, and the advection vector.
- TweenColorChange_3D.js + Tween.js - these scripts perform the animation of the numerical results of the advection-diffusion model solved with explicit dynamics and finite difference method. They display the color values based on the concentration parameter in the following way:

```
StartColorValue = {      EndColorValue  = {
    r = 0,                   r = 255 * cellValue
    g = 0,                   g = 0,
    b = 255,                 b = 255 * (1 - cellValue)
    a = 0                    a = cellValue
}                        }
```

– Spawn3D.js - scripts that generate the matrix of the concentration values and the matrix of objects representing cells in the computational mesh. This script re-computes the values of the scalar concentration field via WorldMesh-Controller.js.

5 Numerical results

We present two illustrative examples. The first concerns the advection-diffusion simulation of a point-shape concentration scalar field source, with the assumed advection and the constant diffusion coefficients. The snapshots from the simulations are presented in Figures 1 and 2. The second one concerns the simulation and visualization of the two-dimensional wave equation, executed over a flat surface, with the initial states, the time step, the dumping constant, and the mesh size of 100x100 elements. We first run the Python code on a laptop to verify our simulation. The exemplary numerical results are presented in Figure 3. They can be compared to the simulations in Lens to be visualized on a floor in Figure 4.

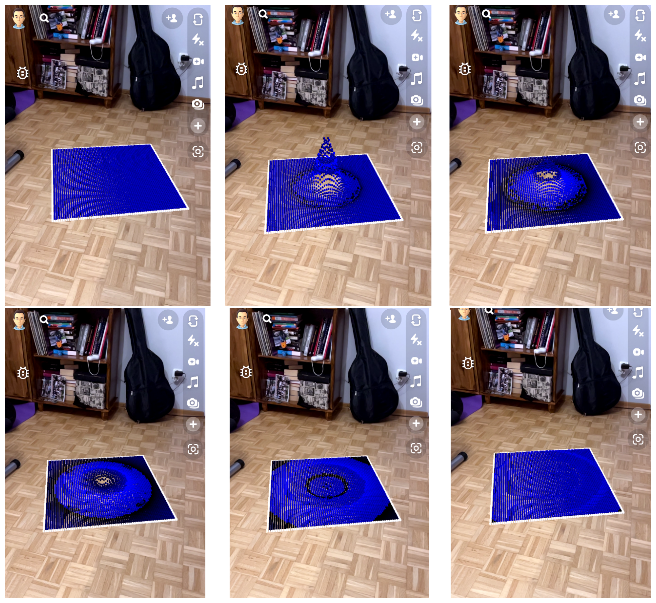

Fig. 4: Another two-dimensional simulations of the wave equation on a floor in augmented reality.

6 Conclusions

In this paper, we proposed a fast solver for performing finite difference explicit dynamics simulations in augmented reality. For two-dimensional simulations, the maximum possible size of the computational grid was 100×100 spatial points. For three-dimensional simulations, the maximum possible size of the computational grid was $30\times30\times30$ spatial points. Visualization of the results for the $30\times30\times30$ grid was characterized by a low frame rate (about five frames per second 5 FPS). Increasing the grid size resulted in an even more significant reduction in FPS. For a better numerical simulations we will need a computationally stronger hardware, or employ precomputed numerical results. Future work may involve experimenting with ARKit [9], and application of explicit dynamics solvers based on higher-order finite element method [10–12].

7 Acknowledgements

The funds the Polish Ministry of Science and Higher Education assigned to AGH University of Krakow. Research project partially supported by the program "Excellence initiative - research university" for the AGH University of Krakow.

References

1. https://docs.snap.com/lens-studio/home
2. J.M. Huang, S.K. Ong, A.Y.C. Nee, Real-time finite element structural analysis in augmented reality, Advances in Engineering Software, 87 (2015) 43-56
3. A. Logg, C. Lundholm, M. Nordaas, Finite element simulation of physical systems in augmented reality, Advances in Engineering Software, 149 (2020) 102902
4. Y. Erkek, S. Erkek, E. Jamei, M. Seyedmahmoudian, A. Stojcevski, B. Horan, Augmented Reality Visualization of Modal Analysis Using the Finite Element Method, Applied Sciences, 11 (2021) 1310
5. J. Lin, J. Cao, J. Zhang, C. Treeck, J. Frisch, Visualization of indoor thermal environment on mobile devices based on augmented reality and computational fluid dynamics. Automation in Construction, 103 (2019) 26–40
6. J. Strikwerda, Finite Difference Schemes and Partial Differential Equations (2nd ed.). Society of Industrial and Applied Mathematics (2004)
7. G. D. Smith, Numerical Solution of Partial Differential Equations: Finite Difference Methods, 3rd ed., Oxford University Press (1985)
8. R. J. LeVeque, Finite Difference Methods for Ordinary and Partial Differential Equations. Society of Industrial and Applied Mathematics (2007)
9. ARKit, Apple Developer Documentation https://developer.apple.com/documentation/arkit (accessed April 17, 2024)
10. M. Woźniak, M. Łoś, M. Paszyński, L. Dalcin, V. M. Calo, Computing and Informatics, 36 (2) (2017) 423-448
11. M. Łoś, J. Munoz-Matute, I. Muga, M. Paszyński, Isogeometric Residual Minimization Method (iGRM) with direction splitting for non-stationary advection–diffusion problems, Computers & Mathematics with Applications, 79 (2) (2020) 213-229
12. M. Łoś, A. Kłusek, M. A. Hassaan, K. Pingali, W. Dzwinel, M. Paszyński, Parallel fast isogeometric L2 projection solver with GALOIS system for 3D tumor growth simulations, Computer Methods in Applied Mechanics and Engineering, 343 (2019) 1-22