h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

Analyzing In-Engine Performance of Realtime Crowd Simulation Approaches

Yassine Boutaouas | Semester 8 | Summer Term 2023

animation +game
B.A. ANIMATION AND GAME
AG.MEDIENCAMPUS.H-DA.DE
M.A. ANIMATION AND GAME DIRECTION
AGD.MEDIENCAMPUS.H-DA.DE

Abstract

An integral part of creating lifelike cities in games is the real-time simulation of NPC crowds [1]. Crowd behavior can play a critical role in enhancing the player's immersion and enjoyment in certain types of games. However, simulating crowds in real-time can be computationally very expensive and is therefore a subject that many game developers deal with, especially when considering that crowds are not simulated in isolation. The project will conduct a series of experiments to examine and evaluate different crowd simulation approaches. It mainly focus on comparing the performance of object-oriented programming (OOP) and data-oriented technology stack approaches to crowd simulation. The performance of each approach will be profiled and compared to one another. The findings of this research project will contribute to the development of more efficient and effective methods for simulating crowds in video games, with potential applications in other fields such as urban planning and crowd management.

State of The Art

Common crowd simulation approaches get grouped into fluid simulations, cellular automata and particle simulations [2]. According to the publication fluid models approach crowds on a larger macroscopic level, suggesting that crowds move similarly to the flow of fluids. The second approach is described as cellular automata, which uses spatial partitioning techniques to create a grid of cells, which can then be used to direct pedestrians in each cell according to the state of neighboring cells. Thirdly, particle-based approaches consider each pedestrian as an individual particle that interacts with its surrounding particles based on physical or social laws [2].

Behavioral Model

Many available crowd simulations implement the flocking behaviors described by Craig W. Reynolds. Even though the publication is based on the behavior that is perceived from herd animals such as fishes, birds or sheep, the rules are applicable to crowds of pedestrians as well. At its core the paper describes three distinct rules that flocks inherit. These are:

1. Collision avoidance
2. Velocity matching
3. Flock centering

Each of these rules can be used to create a behavioral model
for the crowd simulation at hand. Generally, the behavioral model can be extended with custom behavior. One of these extensions is proposed in the article "On the Use of Virtual Animals with Artificial Fear in Virtual Environments" [3]. This article introduces the concept of pheromones that further influence the movement of boids. Each boid is assumed to perceive pheromones of different kinds. In the case of fear a repulsion behavior similar to the rule of separation mentioned above can be added.

$$V_A = \underbrace{(Cf \cdot Cef \cdot Cv)}_{Cohesion} + \underbrace{(Af \cdot Aef \cdot Av)}_{Alignment} + \underbrace{(Sf \cdot Sef \cdot Sv)}_{Separation} + \underbrace{(Ef \cdot Eef \cdot Ev)}_{Escape} \dots$$

$$Velocity_A = limit(V_A, (MVef \cdot MaxVelocity))$$

Scientific Question

The research project centers around the question: What methods can be explored and implemented with regards to object-oriented and data-oriented programming approaches to maintain and improve the overall performance of real-time crowd simulations?

Areas of Improvement

While the research project primarily focuses on comparing OOP and DOTS approaches to crowd simulation, it is worth examining each performance intensive process within the simulation. Overall there are multiple areas of improvement with regards to the design of the base simulation:

1. Programming paradigm: Changing the programming paradigm from object-oriented programming to DOTS programming and using the Burst compiler to improve the overall performance of the simulation [4, 5].

2. Neighbor search: Employing a spatial partitioning method using octrees to improve the process of finding the nearest neighbors of each entity [6].

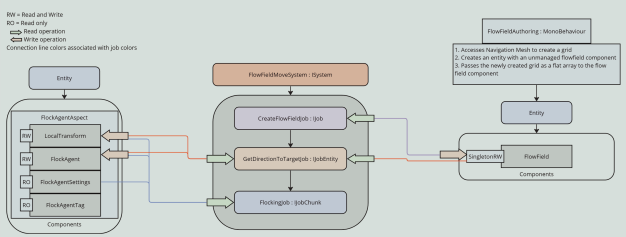
3. Pathfinding: Using a flow field implementation over Unity's shortest-path solution to improve the pathfinding process [7].

4. Rendering: Enabling GPU instancing and using the DOTS hybrid renderer to improve the rendering task [8].

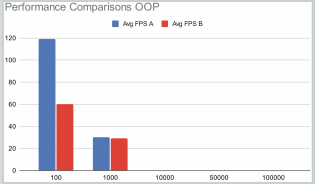
Program Instructions

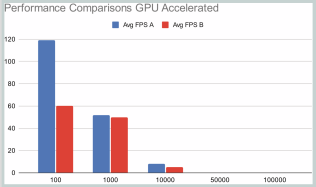
```
1. Input: Array FlockAgent Agents, float PerceptionRadius, float AvoidanceRadius
2. for i = 0 to Agents.Length
    1. for j = 0 to Agents.Length
        1. if i == j then skip iteration
        2. float3 offset ← Agents.Position – Agents[i].Position
        3. float sqrDistance ← offset.x * offset.x + offset.y * offset.y + offset.z * offset.z
        4. if sqrDistance < PerceptionRadius * PerceptionRadius then
            1. Agents.NumPerceivedFlockMates++
            2. Agents.AvgFlockHeading += Agents[i].Forward
            3. Agents.CenterOfFlockMates += Agents[i].Position
            4. if sqrDistance < AvoidanceRadius * AvoidanceRadius then
                1. Agents.AvgAvoidanceHeading -= offset /sqrDistance
                2. end if
        5. end if
    2. end loop
    3. Agents.UpdateVelocity()
3. end loop
```

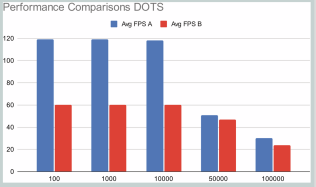
DOTS Program Architecture



Performance Comparisons







Conclusion

Each testing device (reffered to as A and B on the graphs) records 24 to 30 FPS even with 100000 entities on screen, without using a spatial partitioning system. This proves that the Data-Oriented Technology Stack provides a very viable framework, which can be used to create efficient massive crowd simulations.

Regarding the overall animation, the inclusion of a proper collision avoidance algorithm such as the RVO algorithm has to be considered. As stated before, the RVO algorithm can be used in conjunction with the flow field pathfinding system, which suggests its viability [9]. This may be the most important addition to the proposed simulation. In conclusion the simulation at hand can be improved regarding its believability. With regards to the research question, the DOTS simulation approach proves to be a viable solution to greatly improve the overall performance of real-time crowd simulations.

References: [1] Charles Tremblay. 2023. "Building Night City: The Technology of Cyberpunk 2077" Game Developers Conference 2023. || [2] Richard Legget. December 2004. "Real-Time Crowd Simulation: A Review" || [3] Carlos Delgado-Mata, Jesús Ibáñez-Martínez, Simon Bee, Rocio Ruiz-Rodarte and Ruth Aylett. 2007. "On the Use of Virtual Animals with Artificial Fear in Virtual Environments" New Generation Computing, vol.25, no.2, pp. 145-169. || [4] Mutel, Alexandre. September 2019. "Behind the Burst Compiler, Converting .NET IL to Highly Optimized Native Code." || [5] Unity Technologies. n.d. "Mono overview." Unity Documentation. || [6] Bertram H. Drost, Slobodan Ilic. 2018. "Almost constant-time 3D nearest-neighbor lookup using implicit octrees" Machine Vision and Applications, vol.29, pp. 299-311. || [7] Yassine Tazouti, Yasser Lamalem and Khalid Housni. 2022. "Efficient Autonomous Pathfinding Method for Virtual Environments" Proceedings of the 6th International Conference on Big Data and Internet of Things, vol.489, pp. 464-473. || [8] Unity Technologies. n.d. "DOTS Hybrid Renderer" Unity Documentation. || [28] Jur van den Berg, Ming Lin and Dinesh Manocha. 2008. "Reciprocal Velocity Obstacles for Real-Time Multi-Agent Navigation" 2008 IEEE International Conference on Robotics and Automation, pp. 1928-1935.