ECS Spatial Partitioning - Move Millions of Entities

About:

This asset provides a simple base for Unity ECS games with a top-down view which need to move a very large number of individual entities with their own targets, speeds and destination wait time. Use cases range from zombie apocalypse shooters to city builders. If you need a million different moving entities, this asset can be your starting point.

The asset includes placeholder spawner and random movement systems that you should replace with your own logic.

The asset utilizes ECS 1.0 features, Burst and the Job System (DOTS) as well as spatial partitioning to update entities off screen in a more performant, but consistent, way than entities currently within the screen.

Bonus: The asset also comes with an RTS camera script.

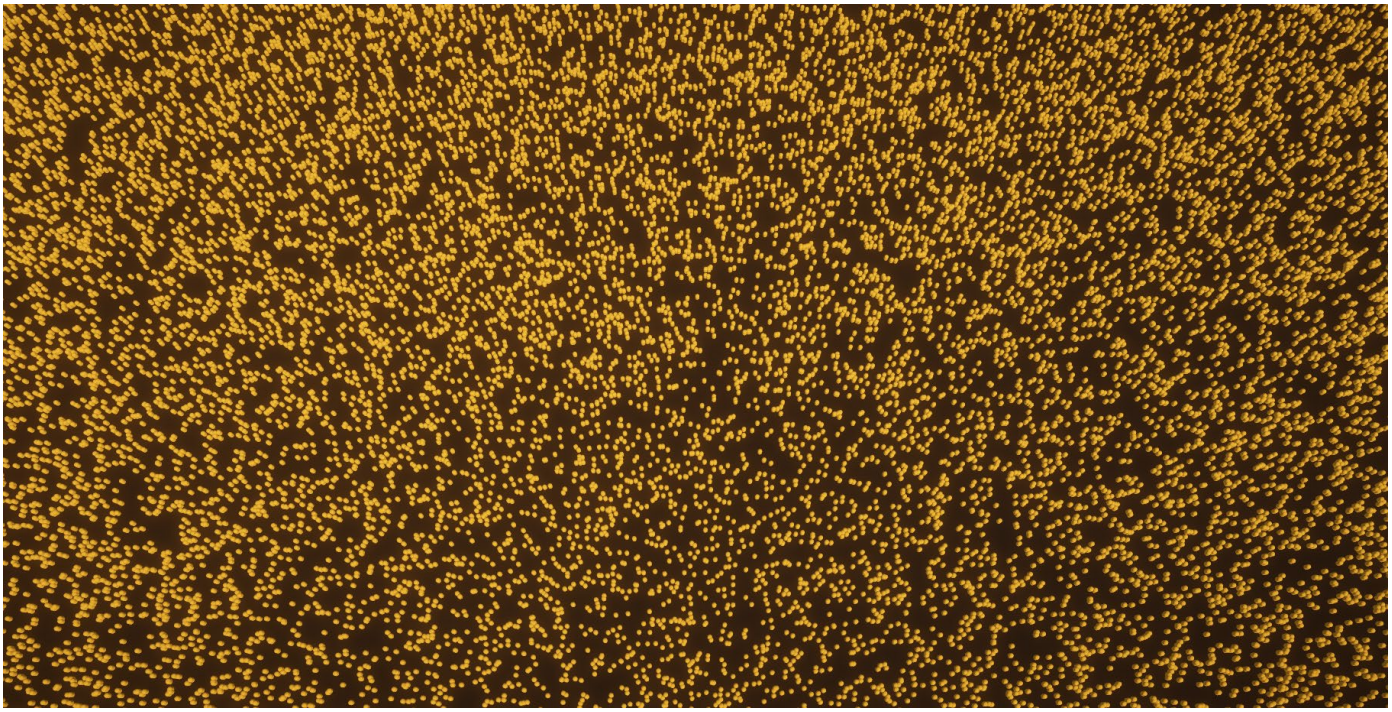

Image: This scene contains one million moving entities and runs at 70-90 FPS on only a i5-9600K CPU.

Project Requirements:

- URP Project in Unity 2022 LTS
- Packages: (1) Entities and (2) Entities Graphics, version 1.0
- Rendering Path: Forward+

Setup:

- Add the scripts to your project and **wait for the Burst compiler to finish**
- Add a new entity and add the SpawnerAuthoring script
- Add a new entity and add the GridAuthoring script
- Create a Prefab and add the GridEntityAuthoring script
- If you want to Prefab to move and/or be disabled when off screen then also add the MovingAuthoring and RendererCanBeDisabledAuthoring scripts
- Reference the prefab in the spawner and configure the amount of agents you want to spawn
- Hit Play (or compile a standalone). You can zoom with the mouse wheel and move the camera around with the middle mouse button, WASD, or arrow keys.
- Ensure that you do not select anything in the Hierarchy window. With that many objects in the scene, it will worsen the performance significantly.

All the above is already set up in the example scene.

Additional thoughts:

- Performance in a standalone build is better than the stats in the editor suggest
- Don't forget to enable GPU instancing on your materials
- The SpawnerSystem and RandomMovementSystem are placeholders that you should replace with your own systems. Please note that they do contain a hardcoded value for the size of the playable area, to keep the code simpler.
- GridEntity, Moving and RendererCanBeDisabled are separated because the asset is intended to also support entities that do not move (e.g., map tiles, buildings, static scene objects).
- The best settings for the cell size (defined in GridAuthoring) depends on the number of entities and their movement speed (how often they change cells). It is a good idea to test and profile this for your individual use case.
- Do not create more cells than you need for your playable area (GridWith and GridHeight).
- When you create new components for your game, think about how often they really need to update when off screen. Many components don't need to be updated every frame. Refer to the MovingSystem for an example on how to distribute updates for off screen entities over multiple frames in a consistent but framerate independend way. Some components don't need to update at all and can be disabled when off screen. Refer to SwitchRendererWhenOffScreenSystem for an example on how to do this.
- Have fun!