WHAT IS THIS?
Delta building visualization (DBV) is a 3D simulation of the delta building with the goal of being a showcase at Delta and Future University of Tartu expos. The simulation is made to work with real-time data that is gathered from sensors spread around the delta building.

WHAT'S NEW?
New additions since last student project competition:
- **Big visual overhaul.** - Added realistic grass. - two new contributors. - Custom rendering is now instance for better usage of computer resources. - Agent system now multi threaded. - New weather system with snow trails. - New and upgraded rigid agent model. - Created 128 handcrafted furniture models. - New node graph navigation network system. - Matched project to real Delta furniture layout. - New unity editor for the node graph nav network • Gigantic performance boost from ~30fps to ~140 fps. - Massively reduced custom draw calls from 2500+ to circa 3. - Recived feedback from University of Tartu students • Created one new thesis.

MULTITHREADED AGENTS

Thanks to the navigation network it was now possible to split the agent system agent calculations onto multiple threads. To do this a thread pooling system was created that takes in work and once the work is done synchronizes back with the main thread. Multi threading allowed us to not worry so much about the performance of the code and keep it simple rather than cooking performant spaghetti.

NEW AGENTS

To render the agents the previous iteration used Unity’s GameObjects and Unity’s Animation component to make them appear on the scene. The simulation is run at Delta with a lot of agents in a single draw call and animation baking to have individually animated agents. The agent model was also upgraded to have a more modern look and it was rigged to make creating animations easier. Instanced rendering is a technique where instead of creating a draw call for each object a single draw call is used for objects that use the same mesh, this is vastly more performant. Animation baking is a technique where the animation data over a period of time and writing that data to a texture which can then be used during rendering to read the vertex data back into the mesh. This allows us to effectively replace costly calculations with a simple texture read.

GRASS

The current iteration has a cool new grass system which renders individual grass tufts onto existing grass patches. The grass tufts locations are prebaked. For better performance the grass tufts are from a .ogg and 3D dither, culled to the view. The reason to render what is necessary. 3D dither culling is a technique where instead of rendering the entire image we render a pattern and is surprisingly effective while not creating a pattern that the user sees, at least not when viewed from eye level. The grass tufts are also moved by simulating wind on them.

BUBBLES

Occasionally the agents emit Sims like speech bubbles with the aim of making the agents more relatable. In the previous iterations bubbles were just Unity GameObjects which had a big negative impact on performance. In the current version the bubbles have been converted to use custom instance rendering. Rendering 10k bubbles in the way of the old iteration resulted in 4.5 fps, rendering them the new way gives us 130 fps. Batching bubbles with different textures in a single draw call is possible thanks to using prebaked texture arrays for the diffuse map. Bubbles also had their visuals upgraded to a more modern style but some polish is required.

FURNITURE

In the old project precise plans of Delta were not available so the furniture was wrong and too big. In this iteration we have completely furnished the first and second floor of the building according to reality and the Delta building plans. To do this we had to model 128 new furniture objects. When creating the models we went for a low poly modern style. Decorations were added outside as well. The building mesh itself was further optimized by removing unnecessary parts like the collar.

NAVIGATION NETWORK

The previous iteration (see last student project competition) used Unity’s pathfinding to move the agents but this is a very general solution and therefore slow and difficult to extend and modify. A new node graph network-based solution was created for this iteration along with a custom Unity editor to make use of it. In this solution the new network is split into nodes and connections, there are three types of nodes to cover our use cases: a walk node to define the path the agents move across, an animation node to define a spot where an agent will do a custom animation and a spawn node to define where the agents come and go from. The editor was made by creating a custom Unity editor window and using mainly Unity’s handles system to draw the gizmos but handles are only drawn near the mouse because using too many handles at once gets quite laggy.

WEATHER SYSTEM

The weather system was rebased from the last project to upgrade it’s visuals. The new weather system supports the following situations: cloudy, snowfall, rainfall, normal. To simulate the falling precipitation particles a custom compute shader particle simulation was created. It works by simulating the particles using gravity and wind and then GPU frustum culls them to renders as few as possible. To make the particles not appear inside the building a height mask texture is used to hide them when inside the building. When it snows a distance tessellated ground plane is used to simulate snow path trails for the agents. To make this possible each agents position is passed to a compute shader that draws them to a snow trail texture. To simulate the snow trails fading the draw texture is regerated to simulate snow accumulating in the trails and filling them.

FEEDBACK

The feedback for new version was highly positive. Most notable improvement were in general appearance of the rooms (p=0.005) and the furniture (p=0.014). Besides that new rooms are remarkably more real-life-like than the old ones (p=0.004). In conclusion new design of the furniture improved the project as a whole (p=0.003) and makes it more visually pleasant. Some of the key takeaways suggest that the new version of furniture could have imperfect placement and more textures. Also it was noted, that for some people the furniture is too colorful.

CONCLUSION

All in all this project iteration has taken over 700 manhours to complete. Compared to the previous version the project gained a hefty boost in both visuals and performance.