Beehive Sim

Joe Bartusek and Skyler Liu
Professor: Felix Heide

Abstract

Beehive Sim visualizes the social construction of honeycomb structure by dozens of simulated bees. This involves two processes, which the bees engage in simultaneously: determining cell locations, and building cell walls around those locations. Each of these are implemented in code with simple constraints, to which bees adhere as they wander around the hive. We allow the user to interact with the simulation by setting parameters such as the number of bees and the variance in bee size, which have implications for structural uniformity and construction speed.

1. Introduction

1.1. Goal

The goal of this project is to graphically simulate the process of beehive construction. We wanted to do so in a way that closely mimics the behaviors of bees in nature. In addition, we wanted to create a visually appealing scene that users could not only interact with but also become enchanted and mesmerized by.

1.2. Previous Work

We were initially interested in building a project that simulated natural and simplistic processes that when scaled, become increasingly complex. Inspired by Sebastian Lague's video "Coding Adventure: Ant and Slime Simulations," where he simulates ant colony behavior and slime mold movement, we decided to look into another beautiful natural process — beehive simulation. A link to the Sebastian Lague video can be found in the References at the end of the paper [3].

While the entire process of beehive construction is not fully understood, specifically whether bees build in a hexagonal pattern to begin with or if hexagonal lattices arise as regularly-spaced cylindrical cells squeezed together, there has been research into the mathematical modelling of honeycomb construction. We took inspiration from Francesco Nazzi's research into the hexagonal shape of bee cells, documented in his paper "The Hexagonal Shape of Honeycomb Cells Depends on the Construction Behavior of Bees" [2]. In addition, we built off of Scott Camazine's work into looking at "self-organization," patterns of cell types that emerge from dynamic bee interactions, which is documented in his paper "Self-organizing Pattern Formation on the Combs of Honey Bee Colonies" [1].

1.3. Approach

We utilize Three.JS Node.JS to visualize the three dimensional honeycomb structure.

The high-level structure of our project is illustrated in Figure 1. The application centers around app.js, which controls which scene is displayed (StartScene or SeedScene) and sets up the renderer, canvas, and camera accordingly. StartScene.js is the index page where users can select inputs for audio, size variance, and number of bees. Once the user starts the simulation, SeedScene.js is called. This scene displays the actual beehive simulation, customized accordingly with the users' inputs and relies on Bee.js, CellLocation.js, CellWalls.js, and Frame.js. In addition, SeedScene.js has lighting and audio characteristics.

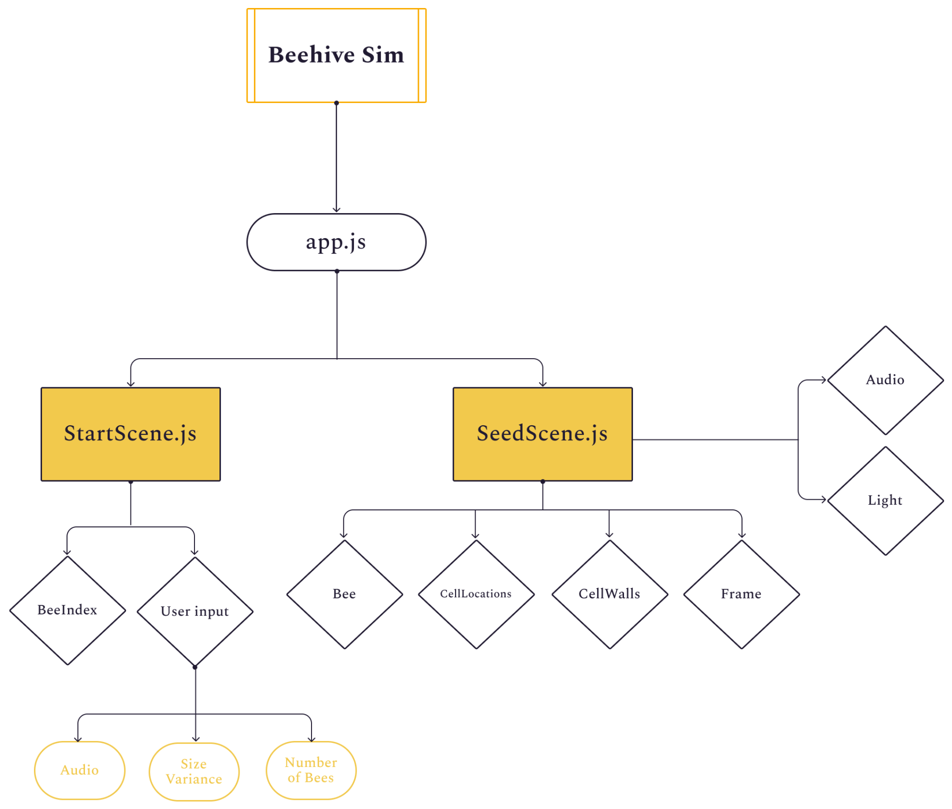

Figure 1: High level project architecture.

2. Methodology

The construction process is simulated over time and incorporates randomness from initial wax depositing to continuous bee movement. We will break the overall process down into two parts: determining cell locations, and building cell walls.

2.1. Cell Locations

Simulation begins with one predetermined cell location, from which all future cell locations are extrapolated. New locations, which are proposed by all bees on every update, are accepted provided they satisfy two constraints. The proposed location must be:

- At least the appropriate distance from all existing locations, and
- The appropriate distance from two existing locations which are also the appropriate distance from each other.

The “appropriate distance” here is bee-specific, and determined randomly for each bee at the beginning of simulation. As discovered by Nazzi, the coefficient of variation of cell size in real-world honeycombs approximates the coefficients of variation of various body parts of bees, implying that bees consult their own body measurements during construction [2]. We define a bee’s personal “measure” as proportional to their size, which is perturbed by a small, random amount. When a bee proposes a new cell location, the constraints are specified using that bee’s personal measure. We also apply a small amount of tolerance for accepting new locations, given that the bees are moving randomly and the chance is tiny that they would arrive at such a precise location. The pseudo-code for cell location proposals is presented in Algorithm 1.

Our implementation of this behavior is based on Nazzi’s work [2], in which he observes that construction of a new cell begins from the groove between two existing cells. This is shown to be sufficient for the alternating pattern of similarly-sized cells to emerge, which we empirically observe in our own simulation (see Figure 3).

As mentioned earlier, the variance in bee size is one of the input parameters which we leave up to the user. The user can expect that higher variance will cause a less regular cell pattern to emerge. Figures 2 and 3 illustrate this phenomenon.

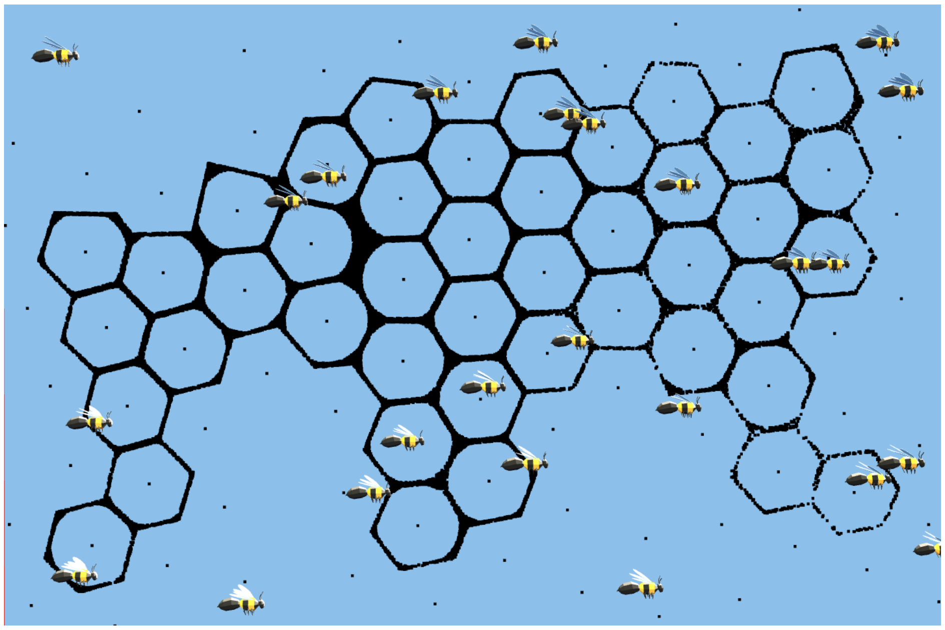

Figure 2: Honeycomb under construction by bees with large size variance. Note the inconsistent cell pattern and susceptibility to unfillable gaps.

2.2. Cell Walls

Cell walls are visualized as a Threejs mesh composed of many point meshes, simulating the gradual process of expanding cell walls via many wax deposits. When determining whether a bee at a given position can add a point, we first require the bee to be close enough to existing cell walls, as real-world honeycomb construction builds directly on existing wax. For the remaining criteria, we first determine the closest planned cell location, which invokes a Voronoi division—this is essentially the process by which alternating, regularly-spaced cell locations give rise to hexagonal

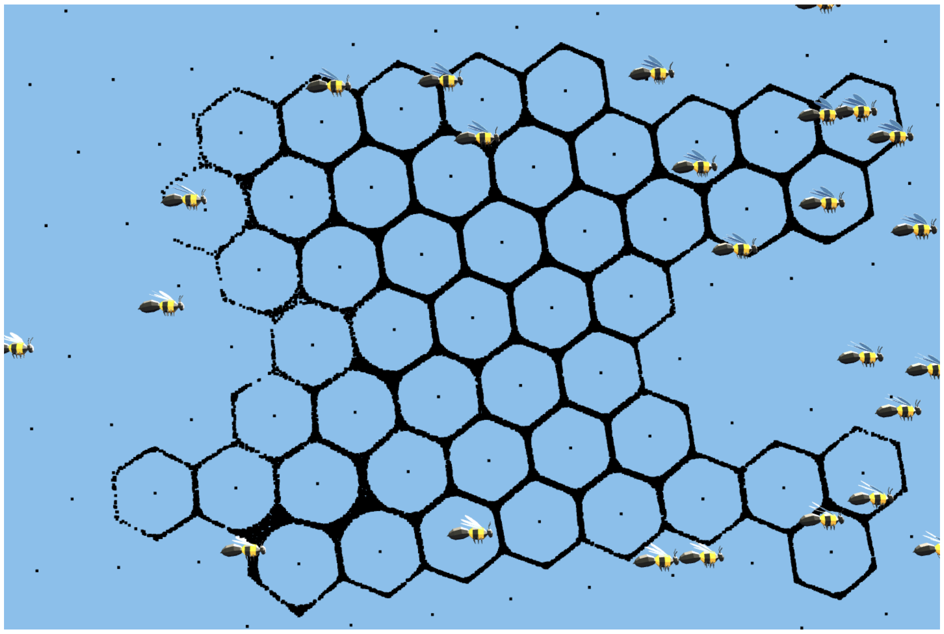

Figure 3: Honeycomb under construction by bees with small size variance. Note the consistent hexagonal cell pattern and resistance to gaps.

cells. We also determine the second-closest cell location, allowing us to control the width of the cell wall. We furthermore determine if the closest cell location is adequately surrounded by other locations (or else the cell wall could bulge out in one direction), and, lastly, constrain the cell wall to beyond a certain radius from the cell location. This last constraint produces the rounded inside corners of the cell walls.

3. Results

As a caveat, we are still working on refining our beehive simulation model so the results included below will be updated before final submission.

The images included below are what our beehive simulation looks like as of now. So far, feedback has been positive. People who have watched the simulation run have reported feeling calmed by the collaborative construction process. In addition, people have been eager to interact with the scene. User interaction will be included in the next phase of development.

4. Discussion and Conclusion

Overall, our project was buzzing. Our application simulated beehive construction and did so while maintaining a high level of integrity to the actual process of honeycomb construction by bees in nature. By working on this project, we were able to expand on coursework covering basic computer graphics concepts, such as meshes, Threejs geometry, and materials, to create a novel simulation.

Since this paper is being submitted as an intermediary step, we are still currently working on additional features. The most important addition that we are working on is implementing self-organization of cell storage roles, as discussed by Scott Camazine, in an attempt to mimic the geometrical representation and deposition mechanisms outlined in his paper [1]. In addition, we are working on incorporating user input by adding in front-end elements that will allow the user to

```
$position \leftarrow$ current position of bee;
$locations \leftarrow$ current array of locations;
$numLocations \leftarrow locations.length$;
$measure \leftarrow$ bee's measure;
$tolerance \leftarrow$ 0.0065 (optimized by hand);
if $numLocations = 1$ then
    $distance \leftarrow$ EuclideanDistance($position$, $locations[0]$);
    if $|distance - measure| < tolerance$ then
        add new location at $position$;
    end
end
if $numLocations > 1$ then
    $apprDistFromTwo \leftarrow false$;
    for $i$ in $numLocations$ do
        $distFirst \leftarrow$ EuclideanDistance($position$, $locations[i]$);
        if $measure - distFirst \geq tolerance$ then
            return;
        end
        $locFirst \leftarrow locations[i]$;
        for $j$ in $numLocations$ do
            if $j = i$ then
                continue;
            end
            $distSecond \leftarrow$ EuclideanDistance($position$, $locations[j]$);
            if $measure - distSecond \geq tolerance$ then
                return;
            end
            $locSecond \leftarrow locations[j]$;
            $distBetween \leftarrow$ EuclideanDistance($locFirst$, $locSecond$);
            if $|distBetween - measure| < tolerance$ then
                add new location at $position$;
            end
        end
    end
end
```

Algorithm 1: Criteria for placing new cell location

customize the scene. At the start of the simulation, the user will be able to interact with the beehive simulation by choosing the audio, the size variance of bees, and the number of bees in the scene.

While we are happy with how our project turned out, there are aspects of the simulation that vary from the actual construction process in nature. For example, our representation does not perform a fluid simulation of the semi-liquid wax with which honeycombs are constructed. In addition, our simulation does not take into account several structural characteristics of honeycombs, specifically that normal honeycombs are two-sided with foundations in between, multiple combs can be constructed in parallel, and that these structures are more three dimensional than in our

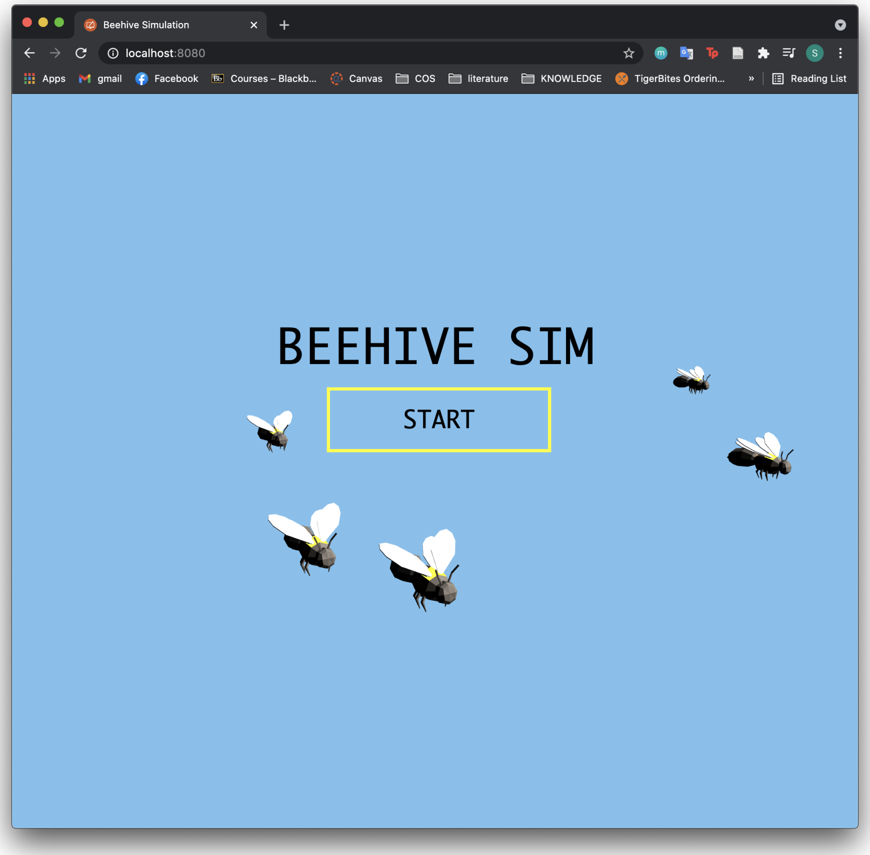

Figure 4: Starting page. Bees move around the scene and user presses button to start simulation.

simulation. Our project also does not take into account gravity and external forces that can influence the construction of beehives. Given these deviations from the actual real-world process of honeycomb construction, future work could be done to make our simulation more closely resemble the natural formation.

5. Contributions

While working on this project we were as busy as bees. We divided the project tasks between the two of us. Joe developed implementations of relevant bee behaviors. Skyler organized the project from a high level, coded the start page, set up components of the scene, and conceptualized user interactions.

References

[1] S. Camazine, "Self-organizing pattern formation on the combs of honey bee colonies," vol. 28, no. 1. [Online]. Available: http://link.springer.com/10.1007/BF00172140

[2] F. Nazzi, "The hexagonal shape of the honeycomb cells depends on the construction behavior of bees," vol. 6, no. 1, p. 28341. [Online]. Available: https://doi.org/10.1038/srep28341

[3] Sebastian Lague, "Coding adventure: Ant and slime simulations." [Online]. Available: https://www.youtube.com/watch?v=X-iSQQgOd1A