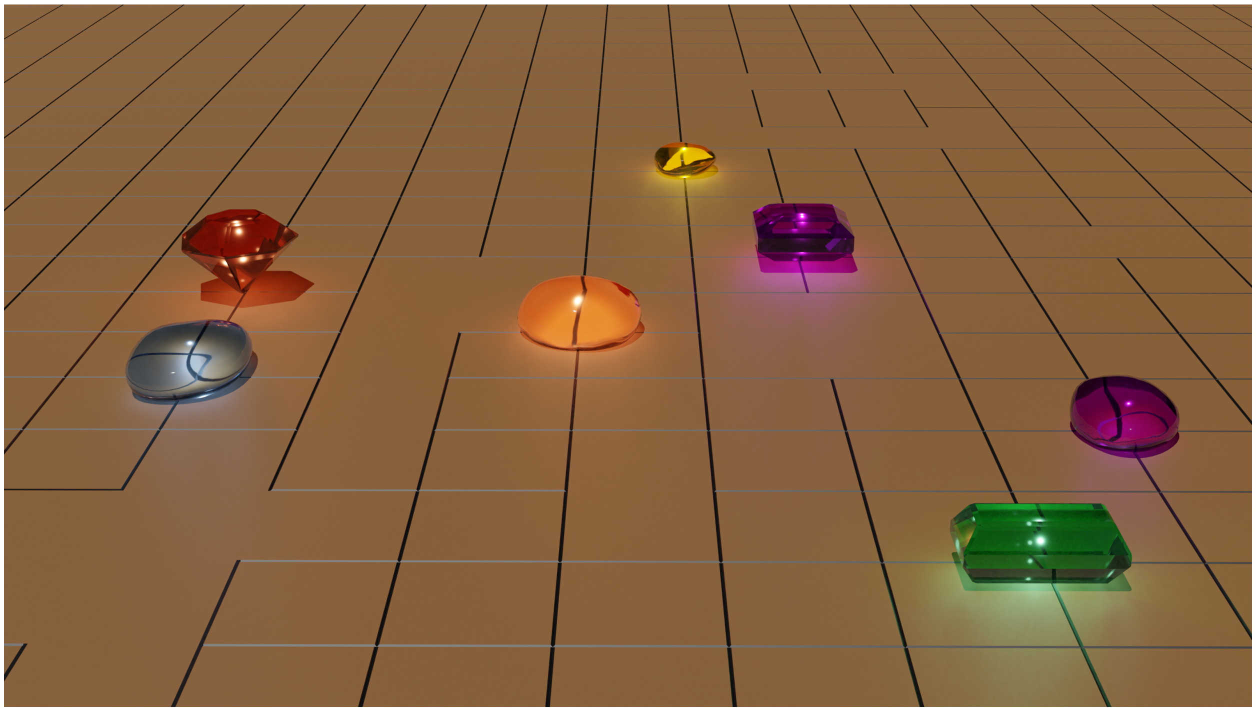
Final Release: *Gemji*

Team *DreiKopf*:

Felix Brendel
Jonas Helms
Van Minh Pham

July 2021



Contents

1 Final Version

Since the alpha release of *Gemji* several additions have been made. Starting with processing feedback from the playtests, we further added features we felt would enhance the experience of the game. In the following we present these features.

1.1 Movement

Previously the player would have to click on a gem, then drag it to the new position. A considerable amount of playtesters found this input method cumbersome. Sometimes the gem would not move to the desired spot because the mouse position was slightly outside of the boundaries of the spot. So we made the following change: When holding a gem now, the game calculates and shows the nearest viable position the selected gem could go to (including the original position) in form of a ghost gem. Should the player now release the mouse key, the gem will snap to that shown location. We believe this input method is an upgrade to the old one as players do not have to drag the gem from spot to spot completely accurately anymore. The exact mouse position on release does not matter as much anymore, the right direction suffices now. This should help players concentrate more on the puzzles without the controls getting in the way anymore.

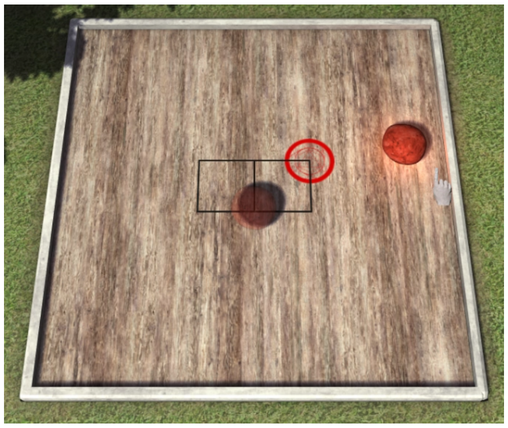


Figure 1: Here a gem is being held here. Should the player now release the mouse button, the gem will snap to the location shown by the ghost gem

1.2 Improved Visuals

The most critized aspect from our playtesters as well as from ourselves after the alpha stage were the visuals of the game. This is why we heavily focused on improving the visual fidelity of our game for the final release in several different ways.

1.2.1 Particle Effects

One of the first things that we wanted to implement for improved visuals were particle effects. Most of the gems in our game have magical effects that interact with each other but the way they interact was not automatically clear just from their animations. For this reason all gem effects also trigger a type dependent particle effect when they trigger their effect. We think that the implemented particle effects are a great way to visualize the effects of the gems and will help our players to understand the game more easily. The particles additionally add more flavor to our world and make the game more fun to play in general. We also added a visual effect when the player is clicking the cursor similar to effects in games like Hearthstone. We think that these kind of effects add a feedback to the player that triggers the same sensory areas in the brain as haptic feedback does and are a great addition to turn-based games like board, card and puzzle games. The particles in our game are implemented as game objects with textures that are rendered on a CPU side quad mesh (just like all other game objects). A wrapper function fills a particle info struct which is then used as the argument to a spawner function which creates the particle game object, animates it and frees said particle from the information stored within the info. The VFX that are used in our game right now are hand crafted and animated but we also implemented some functions for general VFX.

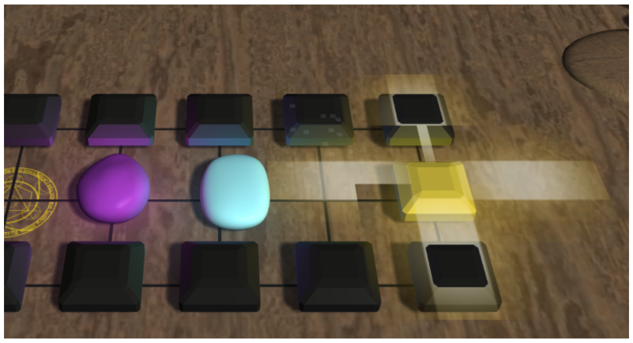


Figure 2: Yellow particle effect

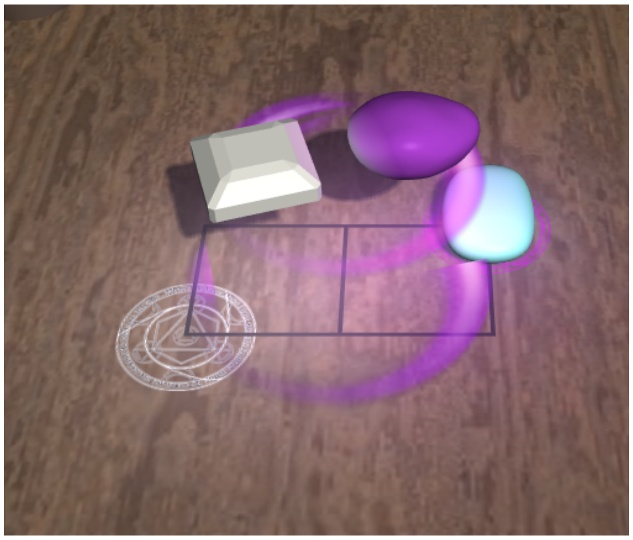


Figure 3: Purple particle effect

1.2.2 Point Lights

To add more atmosphere to the game we implemented point lights and put a colored point light in every colored gem so that it would shine in its own color. While the gems are animated, the lights will follow the gems' position. Point lights consist of a position, color, radius and intensity where the light falls of non-linearly. We experimented with point lights that "emit" negative light, so basically steal light around them for an effect for the black gems, but ultimately decided against it because we wanted to keep the graphics style bright and happy. You can see the effect of negative lighting in Figure 5. Additionally we also animated the light intensity depending on if a gem is triggered or not. However we also noticed that this would make the game harder to understand for beginners and also decided against using it in the final game.

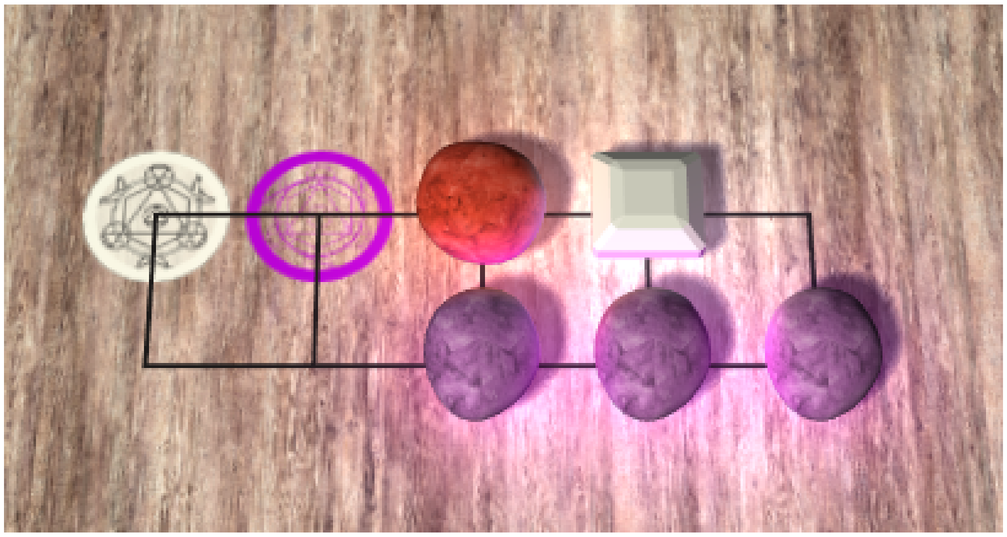


Figure 4: Every gem emits light in its own color

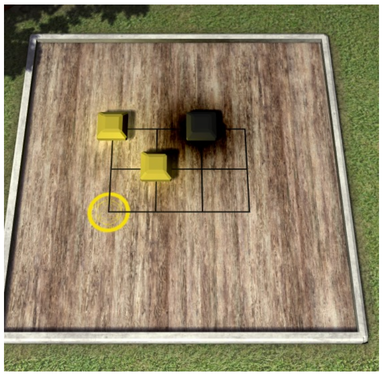


Figure 5: An experimental effect of black gems that did not make it into the final game

1.2.3 Animated cursors & new textures

As we were looking for a general game setting for our game world and scenery, we also decided to upgrade the cursors to better reflect that setting. The new cursors are in form of a marble hand and an hourglass. The new cursors also change depending on the object they are hovering over. There are six different cursors that indicate the following states:

- Normal: The default state
- Can Grab: Additional green ring around the cursor when the hovered gem can be grabbed
- Grabbing: A cursor showing that the player is holding a gem

- Cannot Grab: The normal cursor wih a red cross-out showing a non-movable gem
- Placement Forbidden: The same cross-out without the hand when a placement of a gem would be forbidden
- Hourglass: A hourglass showing during loading and when effect animations are playing

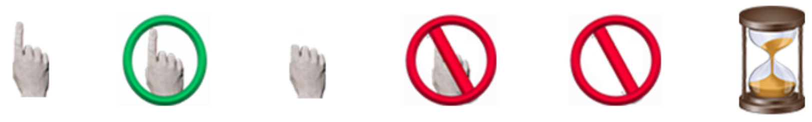
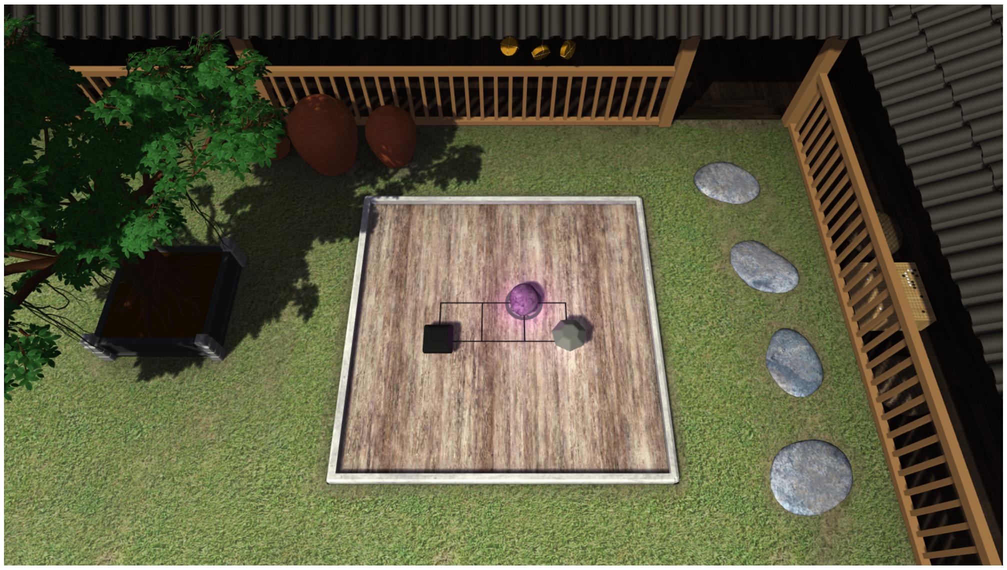


Figure 6: Gemji cursors

1.2.4 Scenery

Since the game did not have a setting yet, we decided to put the playingfield in a 3d environment. We went for the asthetics of a man-sized chess board in the couryard of a building. And since our game drew some inspiration from the ancient chinese board game of Go, we decided on a east asian mood for the surroundings. The models for the scenery was done by us ourselves, except the tree and the textures, which we obtained from the internet. We used Blender for the 3d modelling and texture painting to draw some of the textures on some meshes.



1.2.5 Transparency

To be able to render the ghost gems as an indicator where a gem will be moved to we also had to implement transparency. If a fully opaque gem would be rendered there it would be confusing for the player to see which gem is the preview gem and which gem is the "real" gem. The transparency was implemented using standard alpha blending. We thus also made sure to render the gems in the correct order so that the trancparency was rendered correctly.

1.3 Menus

The bring more structure into the game we added menus. When starting up the game, the player is greeted by a main menu (after the initial loading screen). From there they can either go to the campaign menu to play a level we designed, play a randomly generated level in the endless mode or quit the game.

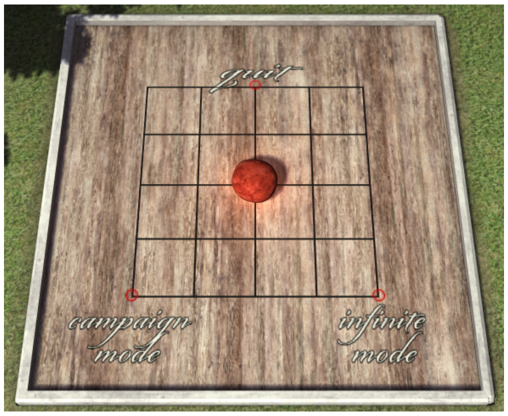


Figure 7: This menu opens at the start of the game and after the player presses escape. The campaign, the endless mode and the quit option are accessible from here.

In the campaign menu players choose between the levels they wish to play. It is important to note that players have to unlock later levels by beating previous ones. From the beginning the very first level is available. To achieve this availability feature we implemented a simple save system. When finishing a level, the game now saves the latest beaten level index in a .txt file. This way we can always convey the players' progress to them. Furthermore when pressing escape the main menu opens up offering flexible transitions between the options. The interesting aspects of our menus is that they themselves are levels. To navigate through them players have to move a gem across a board with the options being finish tiles. In the campaign menu the finish tiles leading to locked levels are sealed off with a black gem on them. This makes locked levels unreachable until they are unlocked.

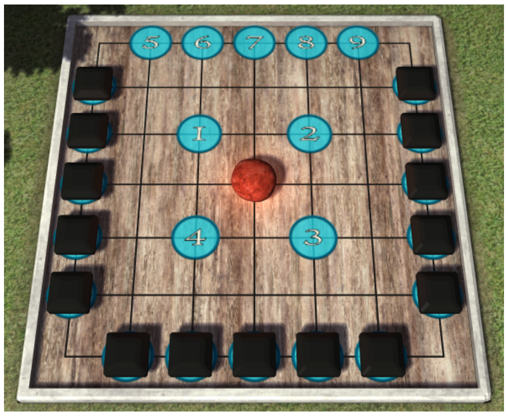


Figure 8: Levels are accessed via the campaign overview. Here the first 9 levels are unlocked. The rest are closed off with black gems.

1.4 Endless Mode

Apart from the main campaign with its 24 levels we implemented an endless mode. This mode procedurally generates new levels for the player to play for as long as the player wants to. To achieve this we integrated our generator from the Interim demo that we had lovingly called Bruce. The details of the implementation of Bruce are located in the report of the Interim demo. As the player solves the levels, the difficulty should gradually increase. So we had parameters for level generation change depending on the difficulty level ranging from 1 to 10. With each beaten level the grid size, number of gems and gem density of the level increases. Since the generator also returns a suggested solution for a generated level, we can also increase the number of steps that this solution needs to raise the difficulty.
Even though we have implemented a clear difficulty curve in the endless mode, we still encourage players to play through the campaign levels first and then move on the endless mode.

2 Experiences

Looking back on this term's project we have certainly gained a lot of experience in game development. These experiences including difficulties, successes and our experience with the theme are illustrated in the following.

2.1 Difficulties

Projects almost always come with a heap of difficulties, usually attributed to synergies within the group, technical nature or from setting the expectations too high from the beginning. This semesters projects difficulties were almost exclusively from a technical nature and something that comes from trying to write a game from the ground up without using a ready to use engine. These are factors the we had already anticipated and were used to due to last semester's project. This means that there is barely anything to mention in terms of difficulties except for the technical challenges we brought upon ourselves. One thing can be said for certain: Even though we took the challenge to not use an engine and managed to power through the issues: Developing everything on your own is very difficult and should not be taken lightly.

2.2 Working with the theme

When we first heard about this semester's theme we were really glad as we thought that this is something we could easily work with. After starting to iterate through ideas that could fit the theme we quickly realized that it might not be as easy as we initially thought it would be. We therefore took an extended period thinking about our initial idea that fits within our perception of Order and Chaos and were quite happy with what we came up with. During the other milestones our confidence in the idea started slowly to fade as the game is essentially not chaotic at all but after the playtest we were again reassured when we realized that for new players it does quite feel like a chaotic game.

2.3 Greatest success

The greatest success for this project was the implementation of the endless mode which was met with a lot of doubt initially. One aspect that we are especially proud of is that we are able to generate these levels in a reasonable amount of time so that there is no real interruption in the flow of the endless mode. This quick calculation of new and solveable levels is realized using a handful of mathematical/optimization tricks implemented by Felix Brendel who deserves recognition for this accomplishment. If you want more information on how he did it you can read it again in the Interim project report. Additionally we think that we were able to create a very fun and chill but at the same time deep puzzle game that relies on very basic mechanics. Considering that there are only 4 active gem effects for now we can see that the possibilities for this game are nowhere near exhausted. We are further very proud that we were able to add several new features to the already existing skeleton engine that we used in last semester's project. These features include:

- Scheduler instantiation
- Raycast mouse interaction
- Better tooltips and interface fading
- Particle System
- Point Lights
- Transparency

2.4 Overall sentiment

We are very happy with the final version of our game and are proud that we were able to implement everything we set out to do. One factor that was definitely influential for this aspect was that we had already taken part in last semester's GamesLab and were used to the report and development schedule. We additionally noticed that the workflow within our group has improved tremendously and that we were much better equipped to split up and manage the workload.