



This project has received funding from the European Union's
EU framework Programme for Research and Innovation
Horizon 2022 under Grant Agreement No 812716

Graduation Project

Playing with density

Dubois Anke

Haelterman Kevin en Adriaensen Kasper (LuGus Studios)

Truijen Theo (Hogeschool PXL)

Preface

Starting at a very young age, I've always had an interest in computers, art, science and videogames, yet sometimes life runs its course and we end up on different paths. Before the worldwide pandemic, I worked in logistics, yet never felt truly satisfied with the direction my life was headed in. During 2020, I made the choice to finally pursue my passions, and start my coding journey, with the hopes to eventually end up in game development.

Besides a love for games, I also look very fondly at games that have an added value to society. Gaming can be used as a tool to create fun, excitement and teach the player new skills while helping them attain knowledge. The project developed in this dissertation, combines exactly that and will take the reader through a literary study regarding density and an explanation of the technical approach and how this concept was turned into a 'Learning Game'.

I'd like to thank both the team at LuGus Studios and Sanne van Loenen for their guidance through this process; they've provided me with necessary insights, help and moral support when most needed.

Summary

LuGus Studios werd opgericht in 2011 met als motto 'Serious about games'. Hierbij gaan ze steeds op zoek naar games met een meerwaarde. Samen met ETN-charming en Early Stage Researcher Sanne van Loenen zijn ze op zoek naar een manier om chemie op een fijne, leerrijke en virtuele manier te verwerken binnen een schoolse context.

Het doel van deze graduaatsproef is dan ook om een werkend experiment in Virtual Reality af te leveren, met als overkoepelend thema 'chemie' en dit voor een leeftijdsgroep van 10 tot 14 jaar oud. Hierbij wordt de focus gelegd op het concept van dichtheid. Dichtheid kan op een aantal manieren aangetoond worden, zonder daarbij te moeten terugvallen op ingewikkelde formules en saaie lessen. Daarnaast moeten de interacties verwerkt in het experiment ook nut hebben en bijdragen aan het uiteindelijke resultaat van het experiment. Met het experiment zelf proberen we kinderen en jonge tieners dan ook een idee te geven van wat dichtheid nu eigenlijk is.

Om dit doel te bereiken worden een aantal stappen ondernomen. Binnen een literatuurstudie wordt bekeken wat dichtheid nu eigenlijk is, de voorkennis van de gebruikers en welke interacties het beoogde resultaat van het experiment zullen bekomen. De interacties binnen het experiment duiden op verschillende handelingen die de gebruiker kan ondernemen, zoals het scannen van een element met een tablet, enzoverder.

Het experiment wordt opgebouwd in Unity, met code geschreven in C#. Hiervoor worden modellen gemaakt in Blender en shaders voegen een extra laag aan animaties en texturen toe. De interacties zullen gebruik maken van de hand tracking, mogelijk gemaakt door het gebruik van de Oculus Quest als VR-bril.

Op het einde van het proces worden de resultaten van het literatuuronderzoek gepresenteerd, samen met het afgewerkte experiment. Hierbij overlopen we de interacties, animaties, andere aspecten en de theoretische basis die vorm hebben gegeven hieraan. Tijdens de presentatie wordt er tevens aandacht geschonken aan de architectuur en uitdagingen van game development in Virtual Reality.

Inhoudsopgave

Introduction	1
1.1 LuGus Studios	1
1.2 ETN - Charming	1
Research question, subquestion and results	2
Motivation	2
Research question and subquestions	2
2.2.1 Research question	3
2.2.2 Sub questions	3
Research and results	3
2.3.1 The concept 'density'	3
2.3.2 Previous knowledge of the end user.	4
2.3.3 The different interactions in our experiment.	5
2.3.4 Timespan	9
2.4 Stakeholders	10
Technical implementation	11
Environment	11
3.1.1 Unity	11
3.1.2 Blender	11
3.1.3 Oculus Quest	12
Technical approach	12
3.2.1 Oculus Integration SDK	17
Result	21
[Table 1: Example of all useable objects]	24
3.3.1 Gamemanager	24
3.3.2 Scriptable objects	25
3.3.3 Interfaces and the selection manager	25
3.3.4 Scripts attached to objects	26
3.3.5 Shaders	28
Conclusion and recommendations	30
Conclusion	30
Recommendation	31
Personal analysis	32
4.3.1 Designer	32
4.3.2 Developer	32
4.3.3 Tester	33
4.3.4 Communicator	34

1 Introduction

1.1 LuGus Studios

With LuGus Studios' lifetime slogan being 'Serious about games', their focus has been on making games that, apart from their obvious entertainment value, usually have 'something more'. This vague description of 'something more' can, in many cases, be defined as an added value. This can be an added societal value or in many cases even educational value. A couple examples of projects like these are 'Mendel', a prototype of a game where a small robot helps the player solve puzzles related to chemistry. Another example is 'De STEM', a VR-project commissioned by Syntra Limburg, in which the player is guided through their first day at a new job, experiencing gender and minority effects in a virtual environment.

LuGus Studios is based in Hasselt, the bustling capital of Limburg, Belgium and has continued to make games ever since they were founded in 2011. At the time of writing there are 8 employees. Albeit a small company, their longest running franchise is currently 'Liftoff', with the first game in the franchise 'Liftoff: FPV Drone Racing' being released in Early Access in 2015. After release it quickly became one of the most popular drone racing simulators currently on the market.

1.2 ETN - Charming

ETN - Charming is short for European Training Network for Chemical Engineering Immersive Learning. Charming tries to focus scientific efforts on making learning about chemistry more engaging and more fun through the use of virtual/augmented reality and other techniques. Charming uses many different people from many different walks of life and disciplines to achieve this and mainly tries to focus their attention on innovative chemical engineering, instructional psychology and pedagogy [1].

2 Research question, subquestion and results

This chapter will describe the research question and go into depth about the sub questions that need an answer. It will also shine a light on the answers to several concerns and will function as a base to build the technical implementation on.

2.1 Motivation

The choice to make an experiment regarding density is both based on personal preference and interest as the willingness and drive of LuGus Studios to look for educational and societal value in the projects created. Furthermore, the experience gained from making a VR-application is substantial and comes with unique challenges that need to be overcome in order to successfully bring this project to an end.

Overall the choice for this experiment is also based on a willingness to add to that educational and societal value, and give back in the form of knowledge and a prototype that also can be used by Sanne van Loenen to further her own research.

LuGus Studios is always looking for new, innovative ways to approach projects that have an added value. In terms of VR they're constantly searching to bring relevance and entertainment to abstract themes such as chemistry. Having done multiple VR-projects like this, having a fresh set of eyes can result in new and unexpected solutions.

2.2 Research question and subquestions

The research question for this dissertation is mainly centered around a single situation. Considering the earlier project proposal, the target age group is children between the ages of 10 and 14, and the goal of our experiment would be for the end users to gain an abstract understanding of the concept of density, without strictly adhering to known formulas, there are a couple questions that in the end will form one big research question.

2.2.1 Research question

“Within the timespan of 10-15 minutes, what steps can we let children between the ages of 10-14 take within a virtual experiment to give them an abstract understanding of the concept of density.”

2.2.2 Sub questions

To formulate a solid, well thought out answer for this larger question, a couple smaller questions can be asked. These questions largely center around the concepts of density and the hypothesis of steps within the experiment to let children ‘feel’ that concept. Feel is a term loosely used within this dissertation, as VR is still a non-tactile medium. Instead of focusing heavily on mathematical formulas, etc., children should be able to intuitively experience and feel what density means.

After defining what density is exactly, it’s important to look at the previous knowledge of our end users, what do they know already and how are they taught? Afterwards, we can mold this knowledge into interactions for the technical implementation of the experiment while also taking a look at the time frame this would fit in.

2.3 Research and results

2.3.1 The concept ‘density’

Often the concept density is described as follows. The formula ($d=m/V$) simply states that density is an object's mass over volume, where d indicates the density, m indicates mass and V indicates the volume of the object.[1] However, for many people this formula teaches nothing about the essence of density. Students are often overly occupied with the math of this concept, without first grasping the effects of density in their own world. [2]

In real-life examples, there are a couple of statements that teach us more about 'density' as a real-life thing, something we can hardly achieve with a mathematical formula.

- Density is a measure of the denseness with which mass is packed.
- The more closely atoms are packed, the higher the density will be.
- Other things being equal, compounds of atoms with greater atomic mass are denser.
- Compression increases density.
- Density of an object does not depend on its size or shape
- "Heavier" as applied to a substance is synonymous with "greater density" [2]

Although these statements already give a better illustration for the concept they also provide some 'rules' that can be transformed into game mechanics or interactions within our experiment to illustrate density. However, depending on the previous knowledge of the end users, it's possible these statements might need to be adjusted.

For this experiment specifically, it is useful to move away from terms such as 'atoms', 'atomic mass', etc. These terms ask for certain previous knowledge or interest in the subject. An interesting takeaway, however, is the fact that the density of an object is not related to its size or shape. In short, what these statements are trying to relay is that density is read from how 'packed' the particles within an object are. [3, Appendix B]

For the sake of simplifying the concept enough to fit within a VR-experiment in line with the one Sanne van Loenen previously designed, the focus will mostly be on density as related to the amount of 'particles' within an object.

2.3.2 Previous knowledge of the end user.

As previously stated, the end users that will be carrying out the experiment are children in the age group 10-14, located in Belgium.

Looking at keywords within Belgian school curricula, "material properties" comes out on top, characterized as "Phenomenological and diverse qualitative characteristics of various materials and their use" [4]

This means that most schools will focus on the properties of objects, such as size and weight. Phenomenology also means that the way these subjects are taught is closely related to how

they would *experience* these properties. Weight, for instance, is easy to distinguish by having a student hold two different materials. This means that concepts that have no initial relation to what students can experience are less prevalent in teaching. [5]

Because of this, the concept of density poses a unique challenge when teaching older kids and pre-teens, as they will often have heard of the terms density, mass and volume, but will conflate mass and volume with weight and capacity. This is likely due to them trying to relate these properties to something they've already experienced. A study in 6 American middle schools also used the concept of floating in fluids and see-through jars that simulated an object. Through experimenting with these, students were able to figure out the previously mentioned rules. Bigger does not always mean more dense and weight does not equal mass.[6].

Due to the challenge of teaching this concept in middle schools, a more practical approach is often needed to ensure teachers overcome their own initial opposition surrounding the teaching of these, considered more advanced, concepts to the age group of 10-14 years old. [6] This is also reflected in the curricula for Belgian schools [4] and thus an assumption can be made that the end users have either no previous knowledge about density or they confuse mass and volume with weight and capacity. In line with how science is taught to the end users, a phenomenological approach to the experiment would be a good approach, which means the focus should be on how our end users experience density, but in a virtual environment without the use of tactile means.

2.3.3 The different interactions in our experiment.

So how would this need for a simplified and abstracted experience translate to a virtual environment? In the previously described examples, students had the ability to displace objects, providing them with tactile feedback. In VR, that tactile element is non-existing. Therefore, it is best to play to the strengths of VR.

A study done in 2017 shows a couple of clear advantages for using VR-technology in an educational context [7]. There's an increased sense of immersion when using VR, pupils explore more and they have access to otherwise limited resources. However, some pupils still showed a lack of interaction that could be attributed to boredom. When designing the

experiment, enough attention should be given to making the interactions fun enough to capture students' attention.

Combining this new knowledge with the information looked at earlier, there are a couple of things that need to be visualized in the experiment to reach an understanding of density.

1. Denser does not mean 'heavier'. Density and weight, although related, are not the same.
2. Denser does not mean 'bigger'. An object might be of a different size, but have an equal density.
3. Density is more related to what is IN the object and this should be shown in a structured universal way.

Knowing this, and knowing that in earlier experiments, putting elements in liquid gave children a better understanding of density when combined with other steps, we can assume that this method also can be used in VR. More so, the virtual objects have no weight, so the student cannot be confused or use this as a parameter to base their understanding off. However, because the objects have no weight, a clear visual system is needed to differentiate the different classes of density. It is still the case that when an object is higher in density, it will sink deeper in liquid.

One of the pitfalls of this experiment could possibly be that the end user does not have the motivation or will to explore the experiment. However one of the advantages of using Virtual Reality is an increased enjoyment [8]. This increased enjoyment comes from the fact that students can be given the feeling they are playing a game, and the concept 'fun' is quite important here. Although our end users need a sense of realism to be able to orient themselves in this 'new' world, this does not exclude using non-realistic elements in our setup. This might divert their attention away from what they know, and force them to reconsider earlier notions.

Visualizing this can be done by combining elements the end users already know but adding an element of fun to it as a motivator and to increase enjoyment and immersiveness. Assuming that the end user has some previous experience with placing an object in liquid to discern certain material properties [4] through the school curricula, the user should be able to place the element in a liquid. To add an element of playfulness, we can use a 'mode of transportation' for the element to get to the desired space. This mode of transportation can

be something that they know but also something that adds that element of fun. Furthermore, the experiment should be designed to encourage exploration without a large amount of verbal guidance from a teacher or adult. This is where techniques often applied by video games come in. There are certain visual cues that can be used to give users hints as to what they need to do, related to the size, texture and color of on screen elements [9]. For this experiment a tube will be used that vacuums up the element. A tube like this is something both related to reality, a technical challenge and an element we can add visual cues, sounds and animations to that add an element of fun and clarity[Figure 1].



[Figure 1: Concept art]

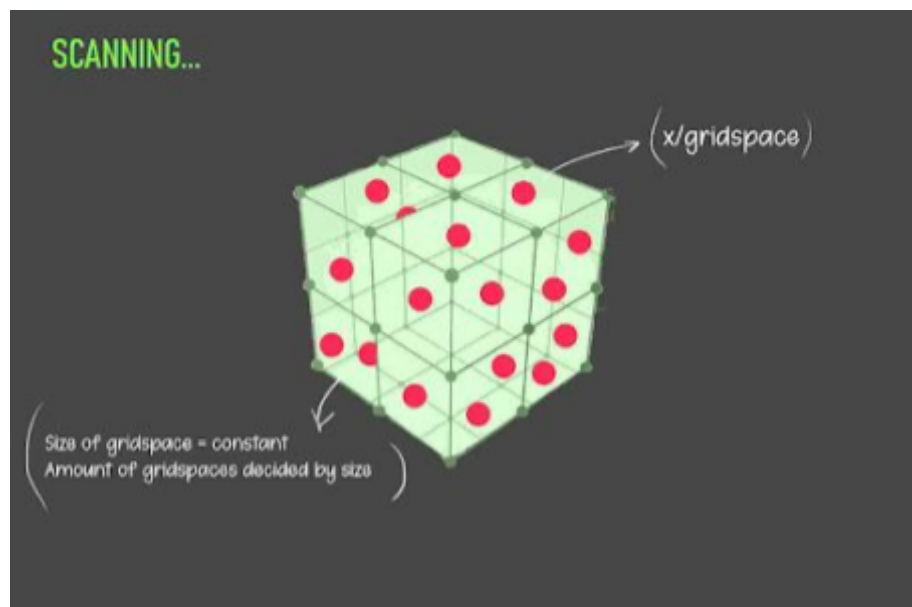
Once the element is in the liquid, it will sink to a certain color, indicating their density. The elements that are spawned however, may have a different size, but the same density, and will therefore sink to the same color in our glass jar[Figure 2]. This hopefully should already challenge the idea of ‘bigger being denser’. However, this interaction alone will not be enough, as the user still gets no visual image of the density of a material. Furthermore, we need an interaction that will return the element back to its original position. The latter can simply be done by a button, as mentioned earlier, visual cues, such as flashing or the use of color can guide the user to interact with a button[Figure 2].



[Figure 2: Concept art]

Now that the user is forced to consider that volume does not dictate where the object lands in our 'glass jar', they should also be able to find out why that is. To do that, we should again pair something fun with something they already know. Children are slowly turning into 'digital natives' and according to a 2020 study in Europe, most children have used a smartphone or a tablet before the age of 12 [10], so providing the users with something similar might entice them to pick up the object and explore. Again, visual cues can provide more guidance.

When picking up the tablet, it's possible to show something new on the screen. To add an element of sci-fi and fun, the tablet can function as a scanning mechanism[Figure 3], effectively showing what would be 'inside' of the element. However, a structured, uniform way of displaying the essence of density, which we've deduced earlier, is needed. As density is closely related to the amount of 'particles' inside an element, we can simplify this into an element having an amount of particle per square in a grid. Because we are playing around with different volumes, the grid would stay the same size. This would translate to an element being the size of four squares in the grid, each square in said grid containing x particles, another element could be only the size of one square in the grid, yet containing the same amount of particles, thus resulting in being the same density[Figure 3].



[Figure 3: Concept art]

Besides these interactions the user should also have the possibility to spawn a new element. A button can be used for this interaction as well.

Combining these interactions would hopefully result in our user reaching a couple conclusions, these conclusions, in no particular order, are:

- An object can be of different size and still have the same density
- Density can be abstracted to $x/1$, where 1 is one square in our made-up grid.
- Denser objects will sink deeper, but this isn't because an object is heavier.

2.3.4 Timespan

Now that the interactions used to reach an understanding of the concept have been looked at, it's imperative to look at the experiment within the physical context it will be used in. In this case, it can be used in classrooms for a workshop or at conventions. All these options have limited use-time in common. If being used in a classroom, there could be up to 20 or even 30 students. In a convention setting, this experiment would also need to work for large quantities of people.

Sanne van Loenen has already considered a couple of disadvantages of using VR in an educational context and built a setup that eliminates some concerns, like dealing with controllers and having to place the VR-glasses on the user. Having a setup that tells users where to put their hands and has the glasses as a static part of the setup severely reduces the time needed to switch between users.

In Flanders, an average school week consists of 28 to 29 blocks of 50 minutes [11]. Considering 20 to 25 students per classroom, if the experiment takes up around 10-15 minutes, this would already take up quite a timeslot. However, the real question here is if 10-15 minutes of experimenting would be enough for our user to understand density.

A fully correct answer to this question would require an analysis of data collected by testing. However, with the time available to finish the product, this is not the main focus and thus, a hypothesis will be made regarding this.

Comparing the method used for our experiment with the methods currently used in Belgian curricula, both of these rely heavily on learning through experience. However, it is not clear how much time is spent on each subject, but we can presume it is somewhere around 50 minutes or more.

Furthermore, children at age 10 learn very differently compared to young teens at age 14, and how fast they learn is highly dependent on their interest and motivation for the subject. In conclusion, a statement can be made that for some users, 10-15 minutes would be a sufficient amount of time to understand the concept, but for some users, it might not be. A conclusive answer would require testing in a real life environment, with children fitting the age group, from various backgrounds, schools and ages. Another option to test this would be to look at the ranking system implemented in Sanne van Loenen's earlier experiment, but due to time constraints, this will not be implemented in this version. Instead a comparison will be implemented between two objects with the same density.

2.4 Stakeholders

Finishing the theoretical part of this dissertation, it's important to take a look at the different stakeholders. The first stakeholder is Sanne van Loenen, as the experiment will be used for her research as part of the ETN-Charming project. Another stakeholder is as such, albeit not directly involved with the development of this project, ETN-Charming and the European Union.

Another stakeholder is of course LuGus Studios, as they are involved in the ETN-Charming project as well as supervising Sanne van Loenen during her PhD project.

3 Technical implementation

3.1 Environment

3.1.1 Unity

One of the technologies used to bring this project to life is Unity. Unity is a game engine used to create 2D or 3D games for several platforms [12].

Game engines function as frameworks for easier, faster and streamlined game creation. Unity can be used to quickly set up environments and scenes, and provides developers with a set of built-in tools that smooth out the process of having to create a game.

In the case of this experiment, Unity will be used to set up the VR-environment and to link the Oculus Quest to the experiment. It will function as a frontend for code written in C# and as the base of the experiment, using built-in functions to make several scripts work together in unison.

3.1.2 Blender

Blender is an open source modeling software, used to make 3D assets. For this experiment, it'll be used to make the elements in the scene and will be used to rig the elements needed. Rigging in game development is a process that adds bones and structures to assets in preparation for movement[13].

Blender is capable of procedurally generating assets or parts of assets and will be used for the tubing system, buttons, etc. [13]

3.1.3 Oculus Quest

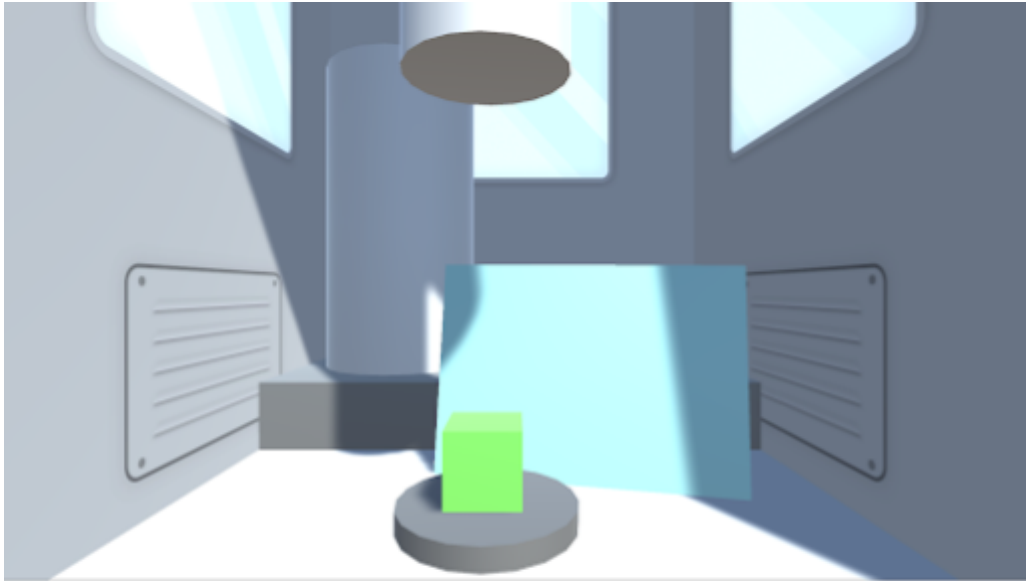
The Oculus Quest is a VR-headset developed by Meta, previously known as Facebook. It's widely known for being one of the best headsets in the world, and is easy to use for both new and experienced users.

In this experiment the Oculus Quest is mounted on the construction, and the experiment can be run from a mobile device [14]. The experiment also uses hand tracking to avoid having to use controllers. The Oculus Quest uses a technique commonly known as inside-out tracking to achieve this. Four cameras mounted on the outside of the headset track the world surrounding you, and detect your hands and fingers. Through machine learning, the system learns what is a hand and what isn't [15].

3.2 Technical approach

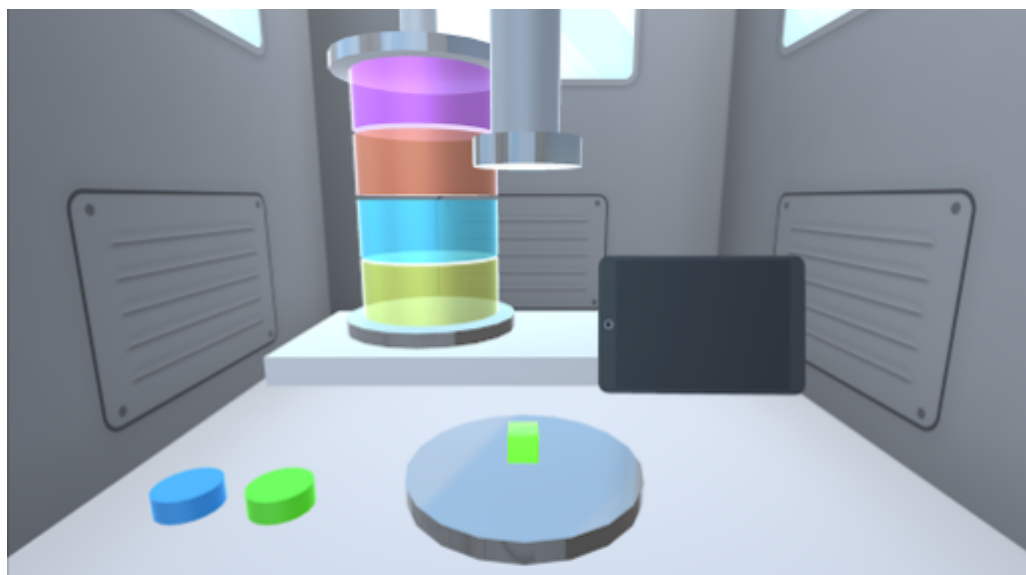
LuGus Studios believes firmly in iterative design. Every iteration, interactions are added and changed, assets are made and replaced and ideas and thoughts are discussed with the team. After brainstorming, changes can be made and some testing can be done by the developer in order to see what works best.

The first step in creating this experiment was setting up a test scene to start programming. The test scene is often a very minimal representation of the objects that will be interactable in a scene. When talking about a *scene*, we're referring to a data container within the Unity Editor that contains all objects and scripts needed to make that experience function[17].

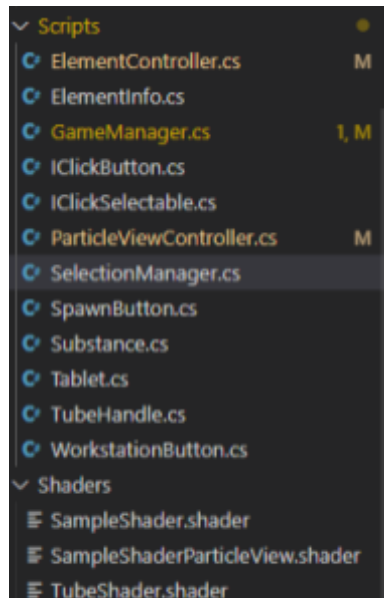


[Figure 4: Initial setup]

After this initial setup[Figure 4], a decision was made to not initially write every interaction immediately for VR. The interactions were first implemented without the use of VR and focused on interaction using the mouse[Figure 5]. During this time, assets were also expanded on, to give greater visual fidelity making sure to keep asset dimensions as consistent as possible.



[Figure 5: Setup to be used with mouse and keyboard]



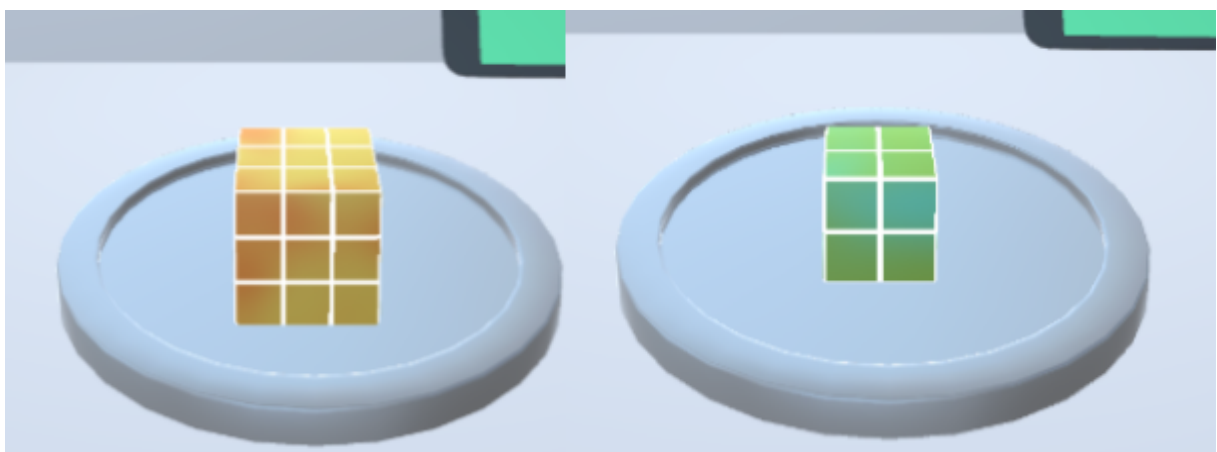
[Figure 6: Collection of scripts attached to objects]

Important when working in Unity is considering architecture and separation of concerns. In this case the architecture uses a GameManager, a SelectionManager and several interfaces and other scripts. In most cases a 'game' will always have a Game Manager, which can be seen as a script that manages everything about the game, such as loading in the initial scene and the objects needed. A Game Manager will also look at if a game is started or ended. The SelectionManager will tie everything together and manage selected objects.

When looking at the rest of the scripts, every object in our scene has a script that is attached and contains the implementation for the fitting interface. Buttons implement

IClickButton, while objects that can be held and dragged around implement IClickSelectable[Figure 6]. By doing this, the application becomes scalable and uses an uniform system to select, move and go through interactions. Examples of the code in these scripts can be found in the next chapter.

There are two scripts in this tree that function as a 'class' that adds properties to objects. ElementInfo and Substance both contain info that can be made into a ScriptableObject. ScriptableObjects in Unity can be seen as custom data containers and can be instantiated as an object in the scene. An example of these ScriptableObjects can be found in the next chapter. These scriptable objects will make sure we can (later on) load in objects dynamically, based on data contained in said objects.

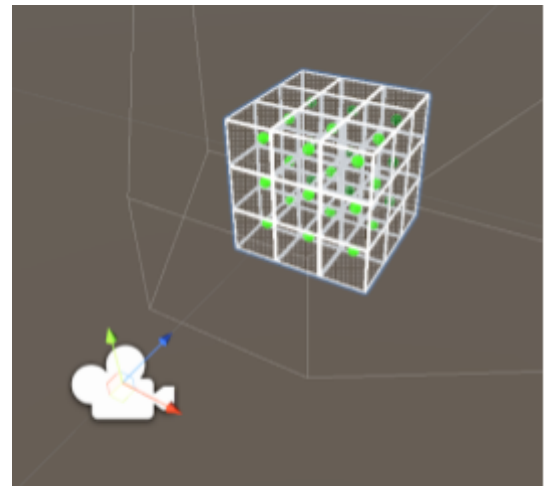


[Figure 7: Examples of rendered elements with grid spaces]

When considering how to render the image on the tablet, in Unity, shaders can be used to create several visual effects. Custom shaders are often scripts that contain small mathematical equations to create these effects. . Input values for the shader can be manipulated through code. Both of these objects have the same shader attached, but the amount of divisions was dynamically adjusted through code. In this case the shader is used to display the amount of grid spaces on the object[Figure 7].

After configuring the shader inputs, the screen of the tablet needs to display the rendered image that shows the internal structure of the element.

A second camera was placed in the scene, and for each gridspace, a cube was spawned with a white outline containing the amount of particles equal to its density[Figure 8]. Looping through all three axes lets us place these new cubes in the shape of a cube itself[Figure 10] with the grid spaces matching the objects pictured[Figure 7]. To make sure the objects, no matter their size, are portrayed with the same size fitting the tablet screen, an offset is added to the camera and the object[Figure 9]. The offset of the cube is added to the position of the game object that needs to be instantiated and the offset to the camera is added to the camera after this process.



[Figure 8: Second camera rendering the image for the tablet screen]

```
Dictionary<int, Vector> cubeOffsets = new Dictionary<int, Vector>
{
    { 1, Vector3.zero },
    { 2, Vector3.one * 0.5f },
    { 3, Vector3.one }
};

//reference
Dictionary<int, float> cameraZAxisOffsets = new Dictionary<int, float>
{
    { 1, 0.15f },
    { 2, 0.25f },
    { 3, 0.3f }
};
```

[Figure 9: Offsets]

```

private void RenderParticleView(ElementController element, int subdiv)
{
    spawnedParticleElementParent.transform.rotation = Quaternion.Identity;

    cubeParent = new GameObject("cubeParent");
    cubeParent.transform.parent = spawnedParticleElementParent.transform;
    cubeParent.transform.localPosition = Vector3.zero;
    cubeParent.transform.localRotation = Quaternion.Identity;

    for (int x = 0; x < subdiv; x++)
    {
        for (int y = 0; y < subdiv; y++)
        {
            for (int z = 0; z < subdiv; z++)
            {
                Vector3 pos = (new Vector3(x, y, z) - cubeOffsets[subdiv]) * ElementInfo.baseSize;
                spawnedParticleElement = Instantiate(cubeParticleViews[elementOnWorkstation.density - 1], pos, Quaternion.Identity);

                spawnedParticleElement.gameObject.name = x + "/" + y + "/" + z;
                //Debug.LogError(x + " " + y);

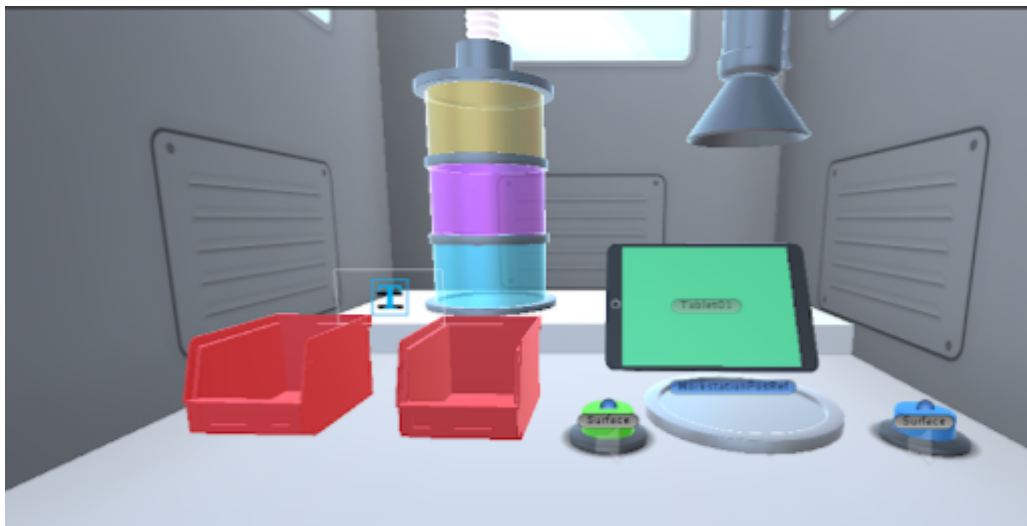
                spawnedParticleElement.transform.parent = cubeParent.transform;
                spawnedParticleElement.transform.localPosition = pos;
            }
        }
    }

    GameObject particleCase = GameObject.FindGameObjectWithTag("ParticleCase");
    particleCase.transform.localPosition = new Vector3(0, 0, -cameraAxisOffsets[subdiv]);
}

```

[Figure 10: For-loops handling the cube spawns]

By testing the experiment in this form, it became clear that there was no clear goal to work towards and no incentive for users to play around. Adding two boxes[Figure 11] that will function as holders to compare two elements made sure the user would be able to receive feedback, whether or not they are going through the experiment as intended.

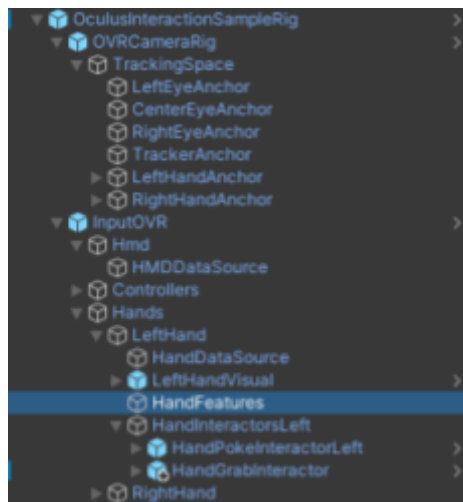


[Figure 11: Setup with two boxes added]

During this stage of the development process the virtual reality and hand tracking was implemented. For the Oculus Quest and Unity, a package is available that provides easier integration into existing systems. After the implementation of the VR, explained in the next

subchapter, the final stages of production were entered, testing it in the physical setup, making changes accordingly, and adding a final layer of polish in the form of post-processing, a process that can add color grading and an overall smoother look to the game[17], animations and shaders.

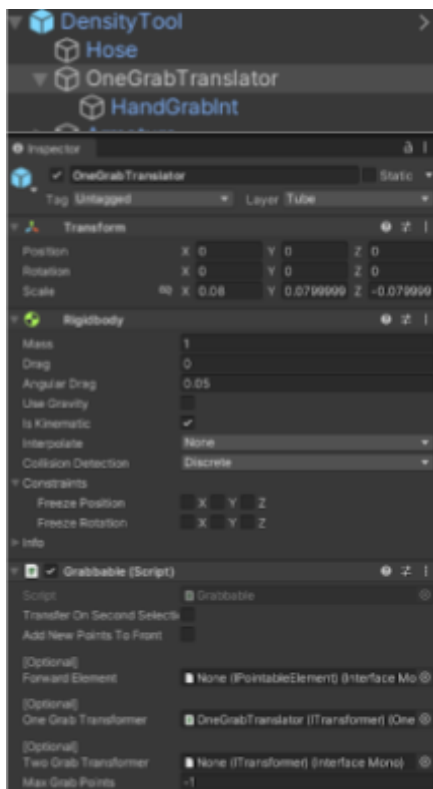
3.2.1 Oculus Integration SDK



[Figure 12: Oculus Integration Sample Rig]

The Oculus Integration Software Development Kit is a package developed by Meta, previously known as Facebook, for easier VR Unity development. It functions as a framework that can be used for easier and faster interactions [16]

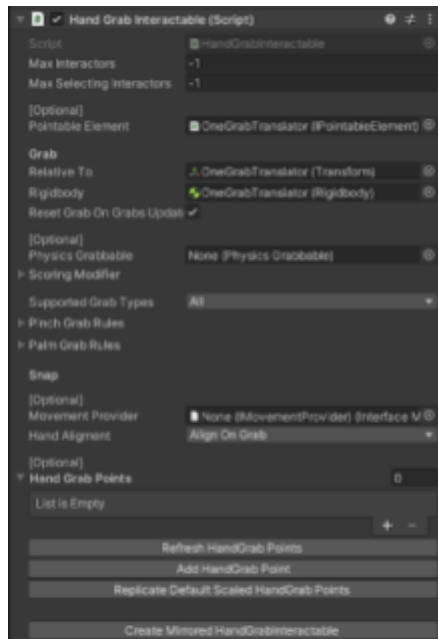
The kit provides developers with premade 'objects' that can be used in a scene, such as rigs that contain the camera, controller and hands needed to implement virtual reality[Figure 12]. These objects have scripts attached that handle eye-tracking and more



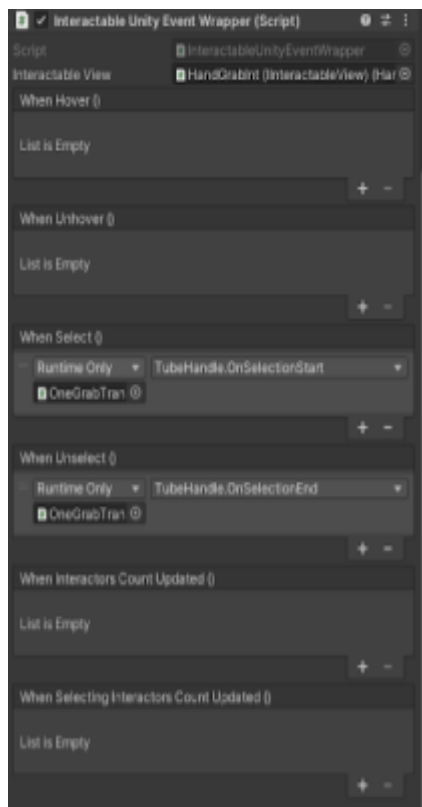
[Figure 13: Grabbable object]

For this project, the integration of controllers was not used and the focus was solely on hand tracking. The SDK provides a couple of interactors, the most important being the 'Grab' and 'Poke' interactors. The interactors in this case are the objects that will interact with the objects in the scene and can be viewed as the instigators of the contact.

The objects in the scene need to be made interactable in return. The way this is done consists of two layers. First, a parent object needs to be available in the scene. This object needs a Rigidbody, which is a component that makes the object subject to Unity's physics engine, providing the object with gravity and a solid body[17]. After this the 'Grabbable' script needs to be attached[Figure 13].



[Figure 14: Hand Grab Interactable]

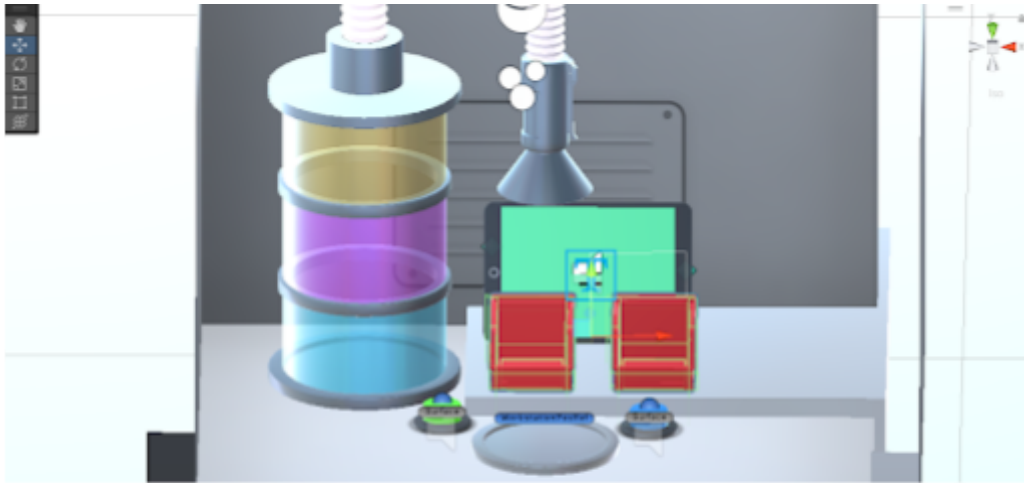


[Figure 15: Unity Event Wrapper]

The second layer is adding the interactable as the child of the grabbable parent. This object does not need a solid body and can be considered 'empty'. The interactable contains a script called 'Hand Grab Interactable'. The earlier parent is used as a reference point for the variables needed in this script. In Unity, the editor provides a drag and drop function for these public variables[Figure 14].

After adding these, an object becomes interactable, meaning it can be picked up, pointed at, etc. depending on the interactors added earlier. However, the interactions provided by the SDK are limited and additional methods are needed to code the interactions. The interfaces we used and the methods we implemented can now be used to add the virtual reality implementation to an event wrapper. This event wrapper takes the place of the Selection Manager. The interactable event wrapper gives options such as `OnSelect()`, `OnHover()` and more. The interfaces we wrote earlier contained functions and methods that were built on the same idea as these event wrappers. For instance, the `SelectionStarted()` methods can be added to the `OnSelect()`, with minimal adjustments to the previously written code[Figure 15].

Building the experiment this way makes the implementation of Virtual Reality much smoother, as architecture can be recycled, without having to add completely separate methods for both ways of controlling the objects in the scene. Examples of the code written within these methods can be found in the next chapter.



[Figure 16: Final setup]

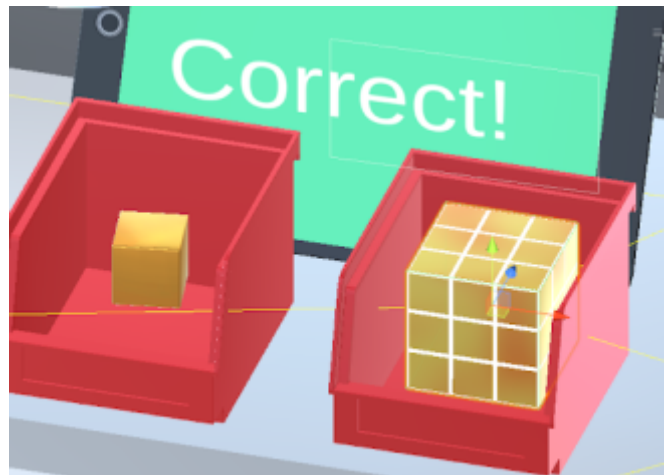
After the implementation of the VR and code cleanup needed for movement and general game mechanics, in the final step the virtual environment was tested in the real life environment that would be used within the context of classrooms, workshops, etc. The placement of game objects was finalized. Most interactable objects were too far to the side and needed to be placed closer towards the center and each other. Eventually, a final setup was reached, while considering the average arm length of a child being far shorter than the arm length of an adult.

Besides the final setup, in this phase, some visuals were also added, such as particle systems, which can emulate certain effects, like explosions, smoke, and other visual indicators and effects. Some of these systems received their own managers, thus providing us with a centralized place to activate these visual systems from. These visual systems also help the end user understand what they can grab, use, and what happens to objects in the virtual time and space. During this part of the process, bugs are usually encountered and fixed as well and restrictions are often added to what the user can and cannot do. These restrictions are important to lead the user through the loop without the application stuttering or crashing. Runtime exceptions in Unity are not necessarily halting and the engine can run through the rest of the code, often breaking the rest of the application[17].



[Figure 17: Reference game object and secondary game object]

A further implementation of the game loop involved adding a reference or 'primary' element that does not have a grid displayed[Figure 17]. Users need to compare this element with a second element by placing them in the boxes[Figure 18]. The boxes are set up with trigger areas. Entering or exiting these trigger areas can function as an event[Figure 19], in this case used to cache the object the user dropped into it.[17]. This system is reused to check what object is in the canister or on the workstation.



[Figure 18: UI displaying a message to the user]

```

public ElementController elementInPrimaryBox;
GameManager manager;
SecondaryBoxTrigger secondaryBoxTrigger;

private void Start()
{
    manager = GameObject.Find("GameManager").GetComponent<GameManager>();
    secondaryBoxTrigger = GameObject.Find("RightBoxTrigger").GetComponent<SecondaryBoxTrigger>();
}

private void OnTriggerEnter(Collider col)
{
    if (col.gameObject.tag != "Element")
    {
        return;
    }

    elementInPrimaryBox = col.gameObject.GetComponent<ElementController>();

    if (secondaryBoxTrigger.elementInSecondaryBox != null)
    {
        if (elementInPrimaryBox.density == secondaryBoxTrigger.elementInSecondaryBox.density)
        {
            StartCoroutine(WinAnimation(2f));
        }
        else
        {
            StartCoroutine(LoseAnimation(2f));
        }
    }
    else
    {
        return;
    }
}
}

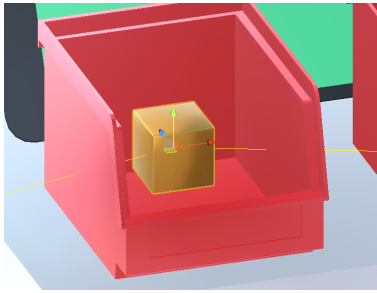
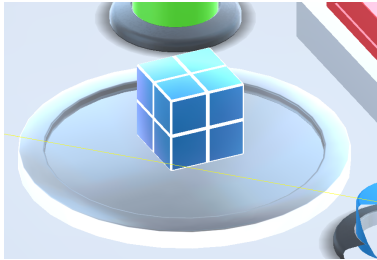
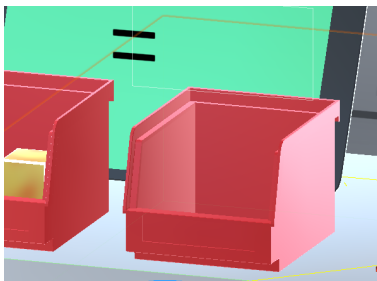
```

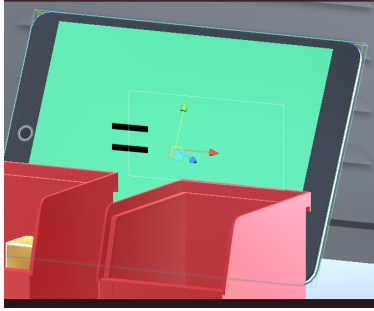
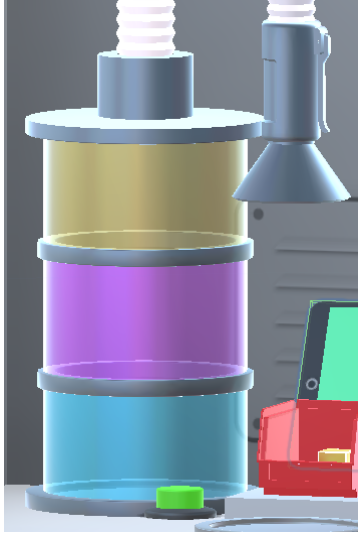

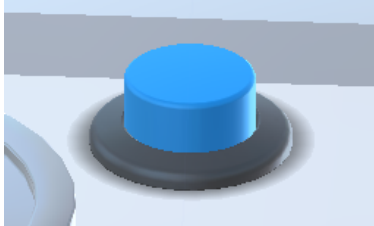
[Figure 19: Trigger event]

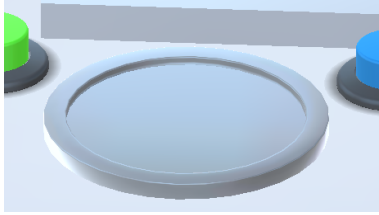
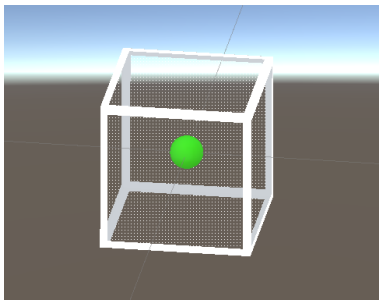
3.3 Result

The technical implementation resulted into a virtual experience that can function in the intended classroom setup. In the following sub-chapters, the different systems discussed earlier in the technical implementation will be clarified through finalized code examples and snippets.

The final product handles several interactions and visual elements, in Table 1.

OBJECT	INTERACTION	VISUAL
	<ul style="list-style-type: none"> • Can be picked up by the user's hands. • Can be sucked up by the element vacuum. • Can be placed on the workstation. • Can be placed in either of two boxes. 	<ul style="list-style-type: none"> • A shader handles the grid divisions based on size of the object. • A particle effect 'teleports' the object from one location to the other. • When the user hovers over the element, it lights up
	<ul style="list-style-type: none"> • Can be picked up by the user's hands. • Can be sucked up by the element vacuum. • Can be placed on the workstation. • Can be placed in one of two boxes. 	<ul style="list-style-type: none"> • A shader makes swirls inside the texture. • A shader handles the amount of grid spaces based on size of the object. • A particle effect 'teleports' the object from one location to the other. • When the user hovers over the element, it lights up
	<ul style="list-style-type: none"> • The end user can place an object within. • The user can put 1 element in each box, and check their density. 	<ul style="list-style-type: none"> • UI-elements float above the boxes, giving the users visual cues whether or not they are correct, or the two objects are not in the boxes. • When a user places an element in the box, it lights up briefly

	<ul style="list-style-type: none"> • The user can pick up the tablet, with the left hand or the right hand • The user can use the tablet to scan the element on the workstation and show the 'internal' makeup of the element. • The user can let the tablet go and it will snap back to its original position. 	<ul style="list-style-type: none"> • The tablet screen renders the view of a second camera • When the tablet returns to its original position, it visually fades out • When the user hovers over the tablet, it lights up briefly.
	<ul style="list-style-type: none"> • The user can grab the handle and pull it down, if it touches an element, said element goes through the vacuum • The user can let go of the handle, and the tubing will move back to its original space. 	<ul style="list-style-type: none"> • When an element gets sucked in, a shader-based animation is played that simulates the tube expanding when the element travels through it. • When the element lands in the liquid, it sinks with a slight animation. • When the user hovers over the handle, it lights up briefly.
	<ul style="list-style-type: none"> • The user can press this button to spawn a new element. 	<ul style="list-style-type: none"> • When the user hovers over this button, it lights up briefly.
	<ul style="list-style-type: none"> • The user can press this button to return the object to the workstation. 	<ul style="list-style-type: none"> • When the user hovers over this button, it lights up briefly.

	<ul style="list-style-type: none"> • The user can place an object onto the workstation, opening it up for multiple interactions, such as scanning. 	<ul style="list-style-type: none"> • When an object is placed on the workstation, it briefly lights up, to indicate to the user that they performed the right action.
	<ul style="list-style-type: none"> • The user cannot directly interact with this object, but gets to see it through interacting with the tablet. 	<ul style="list-style-type: none"> • A shader handles the rendering of this object, based on the element. • A second camera renders this element to the tablet screen.

[Table 1: Example of all useable objects]

3.3.1 Gamemanager

In Unity development a game manager is often utilized to handle the state of the game. It handles basic lifetime management such as when the game starts, when it ends, when it pauses, and more. In our case, when the game starts, an element should be initialized [Figure 19]. The following code example shows the initializing of an element within the game manager.

```
void Start()
{
    // Initialising an element from list of scriptableObjects
    referenceElement = InitReferenceElement();
    correctUI.SetActive(false);
    incorrectUI.SetActive(false);
    equalsign.SetActive(true);

    primaryBoxTrigger = GameObject.Find("LeftBoxTrigger").GetComponent<PrimaryBoxTrigger>();
    secondaryBoxTrigger = GameObject.Find("RightBoxTrigger").GetComponent<SecondaryBoxTrigger>();
    canister = GameObject.Find("TestTubeTrigger").GetComponent<Canister>();
}

public ElementController InitReferenceElement()
{
    // Initialisation of the the reference element with density, scale and a separate material
    // gets a random element out of our list of elements, this simply contains the info we need from the scriptable objects
    chosenElementInfo = elementList[Random.Range(0, elementList.Count)];

    // we have to instantiate the element (the cube is added in the editor, so we know what 'object' to spawn)
    spawnedElement = Instantiate(this.cube, leftBoxPosRef.transform.position, Quaternion.identity);
    spawnedElement.name = "referenceElement";

    // next we need to find the elementcontroller attached to this object and get the density
    elementController = spawnedElement.GetComponent<ElementController>();
    elementController.density = chosenElementInfo.substance.density;

    // we need to set the local size of the object to the size the element should have
    spawnedElement.transform.localScale = Vector3.one * chosenElementInfo.size;

    // In order to set the material, we need to get the Renderer
    Renderer renderer = spawnedElement.GetComponent<Renderer>();
    // set the material
    renderer.material = referenceCubeMaterial;

    return elementController;
}
```

[Figure 19: Example of a method in the Game Manager]

3.3.2 Scriptable objects

Data of the various substances that users can compare is contained in the Substance class, which exposes several configurable properties in the Unity editor. The underlined class attribute allows Unity to serialize this class as a game asset in the project.

```
[CreateAssetMenu(fileName = "Substance", menuName = "density charming/Substance", order = 0)]  
  
public class Substance : ScriptableObject  
{  
    public int density = 1;  
    public Material material = null;  
}
```

[Figure 20: Example of a scriptable object]

3.3.3 Interfaces and the selection manager

In the first phases of the project, before the implementation of the VR, interfaces and the selection manager were used to handle the interactions with mouse and keyboard for the different objects[Figure 21, 22]. As mentioned earlier, in the case of VR, the Unity Event Wrapper takes over the job of the selection manager. The selection manager can be easily repurposed by renaming the methods needed for mouse and keyboard.

```
5 references  
public interface IClickSelectable  
{  
    1 reference  
    void OnSelectionStart();  
    3 references  
    void UpdateSelection();  
    1 reference  
    void OnSelectionEnd();  
}
```

[Figure 21: Example of the methods in an interface]

```

if (Input.GetMouseButtonDown(0))
{
    Ray ray = new Ray();

    ray = Camera.main.ScreenPointToRay(Input.mousePosition);

    if (Physics.Raycast(ray, out hit, (1 << LayerMask.NameToLayer("Element"))
        | (1 << LayerMask.NameToLayer("Tube"))
        | (1 << LayerMask.NameToLayer("Tablet"))))
    {
        Transform selection = hit.transform;
        if (hit.transform != null)
        {
            currentSelectable = hit.transform.GetComponent<IClickSelectable>();
            if (currentSelectable != null)
            {
                currentSelectable.OnSelectionStart();
            }
        }
    }

    if (Physics.Raycast(ray, out hit, (1 << LayerMask.NameToLayer("Button"))))
    {
        Transform selection = hit.transform;
        if (hit.transform != null)
        {
            currentClickable = hit.transform.GetComponent<IClickButton>();
            if (currentClickable != null)
            {
                currentClickable.OnClick();
            }
        }
    }
}

```

[Figure 22: Example of the implementation of an interface]

3.3.4 Scripts attached to objects

A “MonoBehaviour” script can be attached to a game object to implement actions on that specific game object. For the following examples, the script attached to the different elements will be used, as it contains both a Trigger and a Coroutine, two common techniques when developing for Unity.

OnTriggerEnter(Collider col) is a method provided by Unity[Figure 23]. It can be used to check if an object with a collider enters another[17].


```
0 references
void OnTriggerEnter(Collider col)
{
    if (col.gameObject.tag == "Tube" && gameManager.playerIsHoldingTube)
    {
        if (density == 1f)
        {
            StartCoroutine(AnimateTube(0f));
        }
        else if (density == 2f)
        {
            StartCoroutine(AnimateTube(0.1f));
        }
        else
        {
            StartCoroutine(AnimateTube(0.2f));
        }
    }
}
```

[Figure 23: Example of a trigger-event]

This is particularly useful when you need to check if objects hit other specific objects. For instance in this example, only if the trigger collides with the tubing should the rest of the code be executed.

Unity's Coroutines can be used to delay or spread code execution over several frames without using Unity's typical Update method.[Figure 24]. In this example the Coroutine handles the animation of the tubing expanding and the visual trajectory of the element traveling through the tube.

```

IEnumerator AnimateTube(float height)
{
    Explosion();
    this.GetComponent<Renderer>().enabled = false;
    TeleportCube();

    float lumpValue = 0;
    float lumpSpeed = 1f;

    while (lumpValue < 1)
    {
        lumpValue += Time.deltaTime * lumpSpeed;
        tubeRenderer.material.SetFloat("_LumpMiddle", lumpValue);
        yield return null;
    }

    lumpValue = 1;
    tubeRenderer.material.SetFloat("_LumpMiddle", lumpValue);

    yield return new WaitForSeconds(0.5f);

    float cannisterLumpValue = 1;
    float cannisterLumpSpeed = 1f;

    while (cannisterLumpValue > 0)
    {
        cannisterLumpValue -= Time.deltaTime * cannisterLumpSpeed;
        cannisterTubeRenderer.material.SetFloat("_LumpMiddle", cannisterLumpValue);
        yield return null;
    }

    this.GetComponent<Renderer>().enabled = true;
    SetNewPosition(height);
}

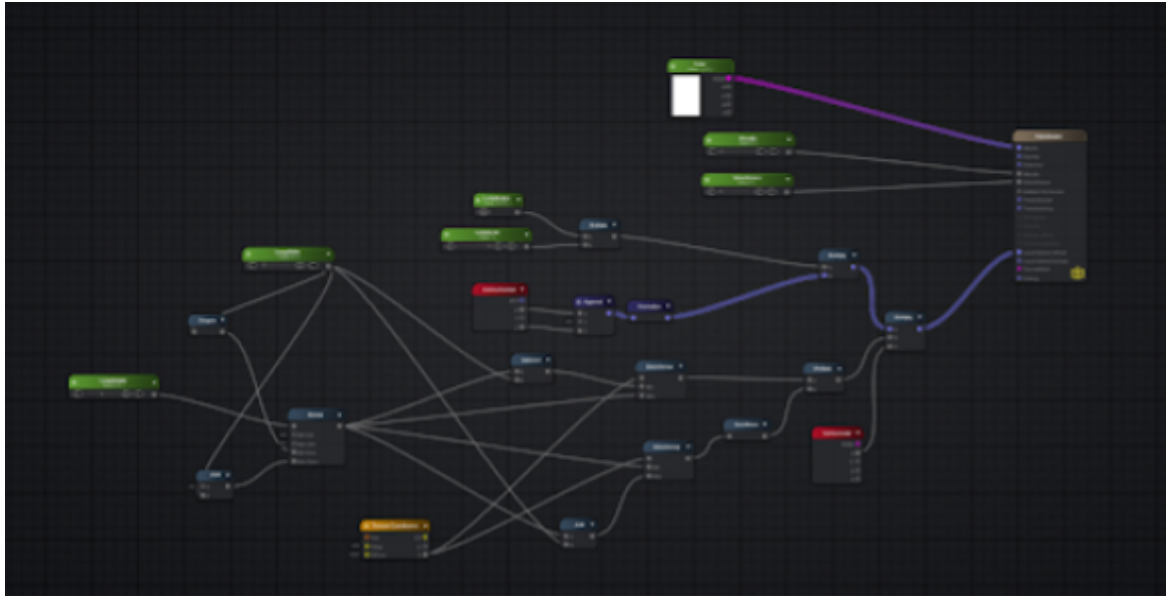
```

[Figure 24: Example of a Coroutine]

3.3.5 Shaders

Throughout this dissertation, the term 'shader' was mentioned a couple times. Below examples contain the shader that handles how the tube can expand and move. Shaders have their base in code, but are often developed with node editors to simplify the development process.

Even in a node-based view[Figure 25], shaders can become quite complicated, often resulting in what looks like a mess of connections and mathematical equations, yet every connection here is necessary to ensure the shader provides the right output.



[Figure 25: Shader that provides the movement for the tubing]

4 Conclusion and recommendations

4.1 Conclusion

“Within the timespan of 10-15 minutes, what steps can we let children between the ages of 10-14 take within a virtual experiment to give them an abstract understanding of the concept of density?”

Looking back at chapter 2, the steps needed in the experiment need to hold up to a couple requirements. These requirements are that size cannot determine the density of the element. Furthermore, a visual representation of the ‘amount of particles’ in an object should be presented to the end users. Denser objects will also float less, but this has nothing to do with weight or mass but simply again, with the amount of particles.

Based on these rules, two big interactions are present. The first one relates to denser objects sinking deeper in the canister. End users ‘vacuum’ up an object and it lands in a certain color. This color indicates a material density. A second interaction makes sure the end user can also get a visual representation. A scanner will show a grid that contains an amount of particles for every grid space. Additionally, smaller interactions needed to have smooth gameplay are a teleporter, to get the object back into the right place, and a button to spawn in new objects.

Separately these interactions can lead our end users to the wrong conclusion, but executed in tandem, they should lead a user to the correct understanding of these core concepts. Providing the user with an object with a certain ‘density’ and creating a pattern that will show them an object with a smaller size but the same ‘density’, one after another, will provide them with all the information needed to develop a feeling for what density is. Furthermore, having them compare these objects with each other and providing them with visual cues whether or not they are correct, will further reinforce this knowledge.

On the more practical side of things, when providing the experiment to a tester, mostly members at LuGus, the testers often started playing around. The hand tracking seems to play a big part in this. By removing the controller and having users use their own hands, they are able to mimic movements they would use in real life, thus making them feel more involved in the process.

4.2 Recommendation

At the time of writing, the experiment is based on randomness. The elements are spawned at random and there is no pattern. There are several ways an end user can 'cheat' the system. For instance, some users may associate the color of the object with the color of the liquid in the canister. Therefore, it would be advisable to carefully design the pattern in which the cubes spawn and investigate in what order the interactions are executed. I'd advise changing the color of certain elements in this case. Furthermore, how many times the end users needs to be reinforced whether or not they are correct is something that can be tested on the target age group. Most of these questions however, are beyond the scope of this project and align more with the research project, in which Sanne van Loenen is currently working.

Another aspect that could benefit from further testing with the age group, are the provided visual cues. Are they sufficient to guide the user through the process? Do users understand the end goal of the experiment without verbal guidance from an adult? Does the flow within the game work? These are aspects that haven't been tested within the age group yet.

In general, the recommendations are more related to the didactic value of this experiment and should be considered before deploying this experiment in a classroom.

4.3 Personal analysis

As someone passionate both about games in general and the development of games that have an added value to society as a whole, this was a wonderful experiment to develop and I feel like I've learned a lot over the past months working on this. To provide some structure to this reflection, I'll be discussing the several roles I've embodied during this entire process. What went well and what were the challenges I faced?

4.3.1 Designer

When designing this experiment, it was both important to consider what the literature told me, the recommendations made by the team at LuGus Studios and the recommendations made by Sanne van Loenen.

The designing aspect is something I definitely still have much to learn about. LuGus Studios believes firmly in 'design by doing', something with which I agree as it often makes the process of designing more in line with my own creative flow. I enjoyed being able to design and develop at the same time, often allowing me to try diverse ways to solve an issue. However, 'design by doing' also has its own challenges. For example, the boxes were an addition that were made quite late along the process, when I realized I had a lack of flow in the experiment and little to no incentive for the end user to want to understand the concept. Due to this late addition in the design of the experiment, changes needed to be made to the code that could've been prevented if I had designed the flow earlier on.

However, in designing I did manage to rule out things that were beyond the scope of my experiment and that would have otherwise been an unnecessary timesink.

4.3.2 Developer

As a developer I've grown in many ways. I now have a firmer grasp of things like interfaces, static classes and how to manage several objects within Unity. I also feel like I've become better at recycling earlier methods and thinking of creative ways to deal with problems that arise during the process.

One of the biggest challenges here was working with Virtual Reality and the Oculus Quest. The Oculus Integration SDK, that was used to implement the interactions, has little to no

documentation available and often I had to figure things out on my own or as a team. The SDK is also fairly new technology, with regular changes that are often not well documented. It's often up to the developer to figure these things out as they go.

The biggest frustration I felt however, was when dealing with the setup of the hardware. Meta has delivered surprisingly unstable software to work with and the connection with Unity often fails, making the development process a lot less streamlined than it could be. I channeled this frustration into wanting it to work, even if I had to wait half a minute for a build every time I wished to test the application. Overall, I'm very pleased that I didn't give up on the project and I managed to stay calm and rational, providing my own solutions to these problems.

By the end of the development process however, I felt like I could now solve problems that seemed impossible before. At the beginning I might not have thought of a Coroutine to solve a problem, but by the end, they became one of the best tools in my toolkit.

4.3.3 Tester

When going through the process of designing and developing this I spent hours in my own experiment, I've gone through every interaction, and this is a guess, probably dozens if not hundreds of times. Things break along the way, and by 'monkey testing', a form of testing where you try to break the application by pushing every button, repeating actions, etc. issues always come forth.

Testing in game development often doesn't rely heavily on systems and can be seen as a more natural part of the process. The biggest challenge here was developing something without being able to test it in the real life fume hood until late in the process

4.3.4 Communicator

During this entire process I reported to both LuGus Studios and Sanne van Loenen. It was a great learning experience to be in contact with and report to both parties involved. Respecting all parties wished in this case has made me develop my soft skills even further.

References


- [1]ETN - Charming, "CHARMING Project – European Training Network for Chemical Engineering Immersive Learning," *ETN-Charming*, 2019. [Online]. Available: <https://charming-etn.eu/charming-project/>. [Accessed: 07-May-2022]
- [2]Encyclopaedia Britannica, "Density," *Encyclopedia Britannica*. 20-Jul-1998 [Online]. Available: <https://www.britannica.com/science/density>. [Accessed: 07-May-2022]
- [3]inspirit, *Density*. 2022 [Online]. Available: <https://www.inspiritvr.com/chemistry/measurements/density-study-guide>
- [4]S. Hawkes, "The Concept of Density," *Journal of Chemical Education*, vol. 81, no. 1, Jan. 2004.
- [5]M. Dumin, "Curriculum Belgian Schools," 03-Apr-2022.
- [6]Stanford, "Phenomenology," *Stanford Encyclopedia of Philosophy*. 16-Nov-2003 [Online]. Available: <https://plato.stanford.edu/entries/phenomenology/>. [Accessed: 07-May-2022]
- [7]S. Dole, D. Clarke, T. Wright, and G. Hilton, "Developing Year 5 Students' Understanding of Density: Implications for Mathematics Teaching," PDF, Merga, 2009 [Online]. Available: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.510.6439&rep=rep1&type=pdf>
- [8]S. Kavanagh, A. Luxton-Reilly, B. Wuensche, and B. Plimmer, "A systematic review of Virtual Reality in education," *Themes in Science & Technology Education*, vol. 10, no. 2, 2017.
- [9]M. Lewallen, "Understanding and Using Visual Cues in Videogames," PDF, Savannah College of Art and Design, 2013 [Online]. Available: <https://ecollections.scad.edu/iii/cpro/DigitalItemPdfViewerPage.external?id=6664483794872025&itemId=1002108&lang=eng&file=%2Fiii%2Fcpro%2Fapp%3Fid%3D6664483794872025%26itemId%3D1002108%26lang%3Deng%26nopassword%3Dtrue%26service%3Dblob%26suite%3Ddef#locale=eng&gridView=true>
- [10]Eurostat, "Being young in Europe today - digital world," *Eurostat*, Jul-2020. [Online]. Available: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Being_young_in_Europe_today_-_digital_world. [Accessed: 07-May-2022]
- [11]Vlaamse Overheid, "Schooljaar, schoolweek, schooldag in basisonderwijs," *Onderwijs.vlaanderen.be*, 2021. [Online]. Available: <https://onderwijs.vlaanderen.be/nl/ouders/organisatie-van-de-school/schooljaar-schoolweek-en-schooldag/schooljaar-schoolweek-schooldag-in-het-basisonderwijs>. [Accessed: 07-May-2022]
- [12]Contributors to Wikimedia projects, "Unity (game engine)," *Wikipedia*, 21-Apr-2022. [Online]. Available: [https://en.wikipedia.org/wiki/Unity_\(game_engine\)](https://en.wikipedia.org/wiki/Unity_(game_engine)). [Accessed: 07-May-2022]

- [13]B. Foundation, "About — blender.org," *blender.org*, 2020. [Online]. Available: <https://www.blender.org/about/>. [Accessed: 07-May-2022]
- [14]G. L. H. H. H. Hector, "Oculus Quest 2," *TechRadar*, 26-Jul-2021. [Online]. Available: <https://www.techradar.com/reviews/oculus-quest-2-review>. [Accessed: 07-May-2022]
- [15]S. Butler, "How Oculus Quest Hand Tracking Technology Works," *Online Tech Tips*, 10-Mar-2021. [Online]. Available: <https://www.online-tech-tips.com/gaming/how-oculus-quest-hand-tracking-technology-works/>. [Accessed: 07-May-2022]
- [16]Meta, "Oculus Developer Center," *Developer.Oculus.com*, 27-Apr-2022. [Online]. Available: <https://developer.oculus.com/downloads/package/unity-integration/>. [Accessed: 21-May-2022]
- [17]U. Technologies, "Unity," *Manual: Unity User Manual 2021.3 (LTS)*, 15-May-2022. [Online]. Available: <https://docs.unity3d.com/Manual/index.html>. [Accessed: 20-May-2022]

Appendix

a. Michael Dumin, 2022 - Powerpoint

C3 Belgium (both) – Key figures



0

	Primary (Science)		Secondary (Chemistry)		Sum	
Top 3 keywords [%]	Material properties	20.8	Particle model	18.0	Material properties	13.6
	Environmental awareness	15.1	Quantities and measurements	12.0	Quantities and measurements	8.7
	Recycling	15.1	Energy concepts	12.0	Particle model, Energy concepts	8.7
Content/Non-content ratio	36/64		64/36		50/50	

Table 1. Top 3 keywords in [%] from all Belgian curricula in primary and secondary schools (Katholiek and GO schools; until 8th school year).



1

Keyword	Short explanation
ACIDITY AND ALKALINITY	Acids, bases and indicators with examples, and the pH scale
ATOMIC MODEL	The atomic models of Rutherford and Bohr with their subatomic components
BONDS AND INTERACTIONS	The covalent and ionic bond, their formation and breaking with exemplary compounds
CARBON CHEMISTRY	Rudimentary aspects of organic chemistry (alkanes, functional groups)
CHEMICAL REACTION	Change on a molecular level as the most basic chemical phenomenon, and the distinction of different reaction types
DETECTION REACTIONS	The verification of chemical species like hydrogen or oxygen by common reactions
ELEMENTS AND COMPOUND PROPERTIES	Metallic or non-metallic elements or compounds and their specific chemical properties at the particulate scale
ENERGY CONCEPTS	Fundamental laws of energy, its different forms and transformations in nature
EXPERIMENTS AND SET UPS	Active, experimental actions with according laboratory devices which results in qualitative observations and knowledge gain
MAGNETISM	The concept of magnetism and magnetic effects as unique characteristics of some materials
MATERIAL PROPERTIES	Phenomenological and diverse qualitative characteristics of various materials and their use
PARTICLE MODEL	The atomic model of Dalton as smallest, indivisible particles of matter
PERIODIC TABLE	The table of Mendeleev as an order and classification system of all elements
PURE SUBSTANCES AND MIXTURES	Different forms of pure substances, hetero- and homogenous mixtures, their dissolving and mixing characteristics under various conditions
QUANTITIES AND MEASUREMENTS	Various magnitudes, their SI units, ways to measure them, and other determinations and calculations related to chemistry
SEPARATION TECHNIQUES	The introduction of different ways to separate substances for all aggregation states
STATES OF AGGREGATION	The three states of matter (solid, liquid, gaseous), their characteristics and transition processes
SYMBOLS AND FORMULA	Chemical notation of elements and compounds, and basic systematic nomenclature

Keyword	Short explanation
AIR	Air as a special gas component mixture of fundamental importance for nature
CHEMIST	The profession as a chemist and its significance in STEM fields and society
ECONOMIC AND INDUSTRIAL IMPORTANCE	Important aspects of chemical fields or specific chemicals in industry
ENVIRONMENTAL AWARENESS	A reflective mental process by pupils to recognize and value the direct environment of dynamic nature
HOUSEHOLD AND DAILY LIFE	Practical contexts of chemistry and use of chemicals in everyday life and technology
HEALTH, SAFETY AND ENVIRONMENT (HSE)	Knowledge and training for responsible use of hazardous substances and general safety-conscious behavior
RECYCLING	Learning about and being encouraged to take part in recycling processes, waste disposal, waste prevention or reduction
SUSTAINABILITY & RESOURCES	Knowledge, evaluation and sustainable use of different kinds of resources and energy
WATER	Water as an exceptional substance of nature, its various forms, and the use and value for humans

Cycle 3 - Belgium both

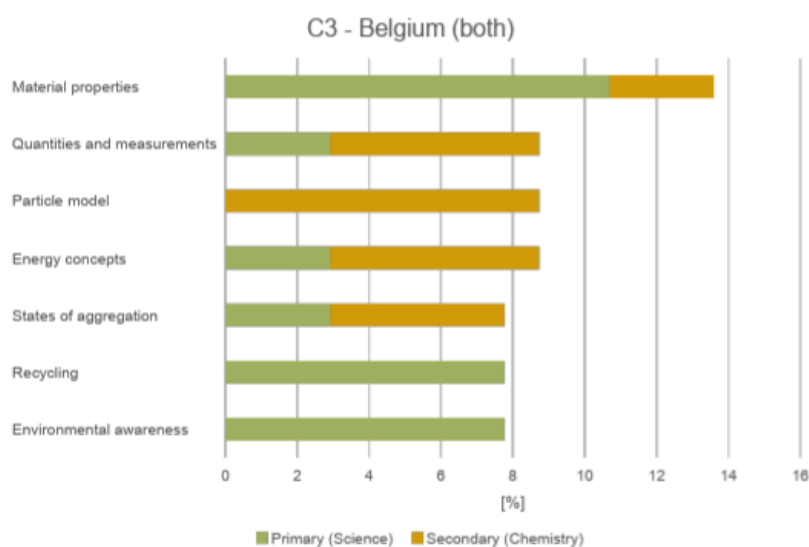


Figure 14. Top 10 fraction of chemical key concepts in [%] after cycle 3 analysis of Belgium GO schools (54 key word entries total).

Cycle 3 - Belgium GO



4

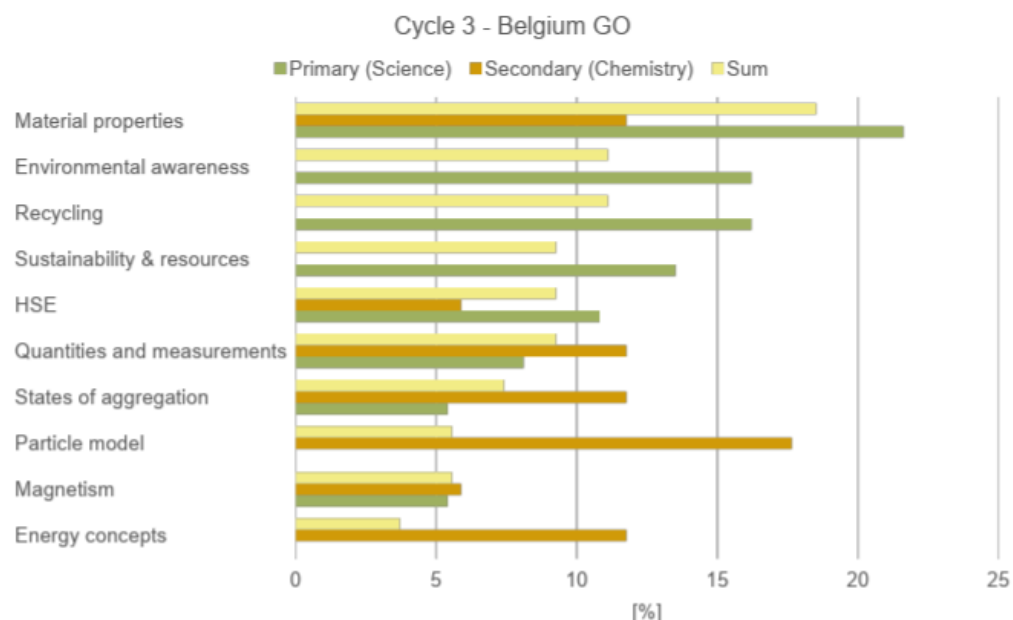


Figure 14. Top 10 fraction of chemical key concepts in [%] after cycle 2 analysis of Belgium GO schools (54 key word entries total).

Cycle 3 - Belgium Katholiek



5

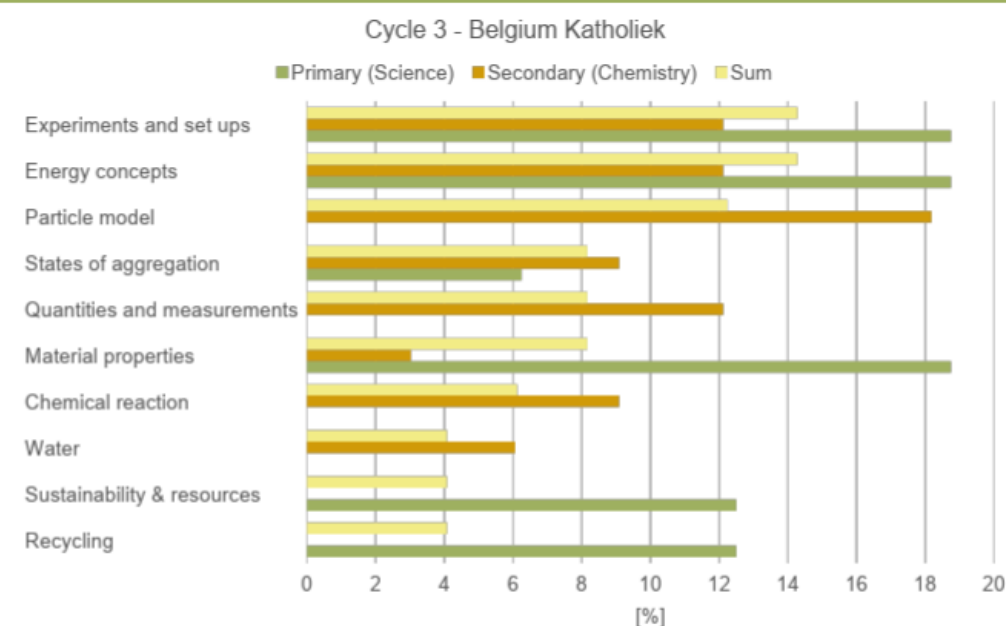
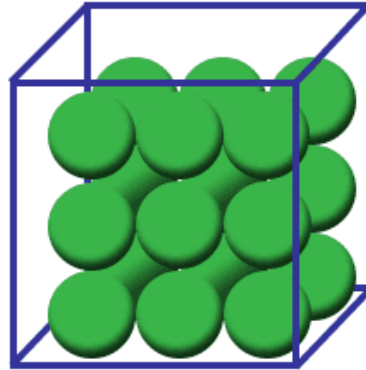
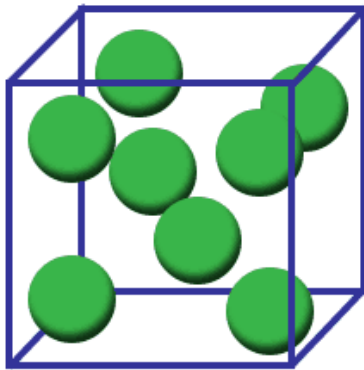


Figure 15. Top 10 fraction of chemical key concepts in [%] after cycle 3 analysis of Belgium Katholiek schools (49 key word entries total).

b. Figure explaining density

Density



TheEngineeringMindset.com