DATA PHYSICALIZATION FOR CLIMATE CHANGE DATA USING AN INTERACTIVE GLOBE: EXPLORATION OF ENCODING VARIABLES INTO LEDS

Bachelor of Science Thesis Creative Technology

Lukas Korné van Hoeven
S2849119

July 2024

Faculty EEMCS, University of Twente

First supervisor: dr. Champika Epa Ranasinghe

Second supervisor: Auriol Degbelo

External supervisors: dr. Yuri von Engelhardt

and dr. Paulo Raposo

Critical observer: Nacir Bouali

ABSTRACT

This thesis explores the potential of data physicalization for climate change data using an interactive spherical display. By integrating individually addressable LEDs, the project investigates the effectiveness of different encoding variables; colour and animation (blink speed) in enhancing data comprehension and user engagement. The study employs a combination of background research, system design, and user evaluation to assess the impact of these visual modalities on accuracy, mental load, satisfaction, and overall user experience. Results indicate that while animation modes yield higher accuracy in data interpretation, they also pose significant usability challenges. Blink speed, although accurate and engaging, was perceived as obtrusive and demanding. Colour-based representations were preferred for their ease of use and lower mental load. The findings underscore the importance of balancing visual effectiveness with user comfort in the design of spherical displays.

ACKNOWLEDGEMENTS

I'd like to thank Champika Epa Ranasinghe and Auriol Degbelo for supervising the project and for their enthusiasm regarding the globe we built. I'd also like to thank María Cobo, my co-researcher and good friend, for the work she has done for the project.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. CONTEXT AND RELEVANCE

Climate change is an important topic nowadays. Climate experts suggests that the effects may threaten the way people live [1]. Communicating about climate change is important. Traditionally this is done using maps with geographic information [1]. Communicating in these in 2d maps reduce accurateness, since there is no accurate way to turn a spherical form into a flat 2d representation [2]. Globes are the most accurate representation of earth since it is a miniature version of the actual earth. Representing this data in spherical form. There are multiple metrics of showing climate change data, which include land precipitation, Arctic ice coverage, air temperature, and sea temperature [3].

There have been installations showcasing climate change or geographic information in various ways, like utilizing 2d charts. Visualisations in AR and VR have also been explored; users were presented with a digital twin of the earth that showed various ways of displaying geographic data. The users could rotate and manipulate the digital twin [2], [4]. However, these lack physicality and do not enable multiple users to interact with them. This leaves users with an experience that is not tangible and that does not enable users to share the experience amongst themselves. Spherical displays are also used, but not available for a wide public because of their price.

Utilizing tangible interactions in these type of physicalizations can help make the data more perceivable [5]. Tangible interactions may make this data more fun, engaging and more approachable [5]. The approachability and ease of use can assist in improving the knowledge of the general public about climate change through interacting with relevant data.

The field of physicalizations relatively new and upcoming and shows a lot of potential for representing data in fun and engaging ways. It can offer perceptual and communicative value that computer visualizations do not offer [6]. Combining the possibilities of spherical displays and concepts of data physicalization may offer a possible solution for showing climate change in a fun and engaging way. Since the physicalization should show data, it is of importance that relevant technologies are used. Individually addressable light emitting diodes (LED's) are a way to show data in a relatively easy manner and offer easily programmable methods that enable rapid prototyping.

1.2. CHALLENGES AND RESEARCH QUESTIONS

Although there are four main indicators of climate change, the project focusses on just air temperature. LED's have been explored in data physicalizations, they offer multiple ways of showing data [7]. They seem a fitting technology to represent temperature values in different ways, by for instance using colour scales. However, there is one more property of LED's that can be utilized for communicating data in physicalizations, animation or blink speed. The research gap in comparing different properties of LED's for communicating data in physicalizations is the motivation to explore the combination of blink speed and colour modalities for the communication of air temperature data. This study explores the efficiency, accuracy, mental load, satisfaction, and user experience of different LED output modes on an interactive globe. Therefore, the research question is: ***How do different combination of output modes: colour and blink speed, and each modality separately, compare***

in data perception and user experience of a data physicalization conveying climate change data?

2. Background Research

2.1. Literature review on spherical displays

This literature review will focus on the advantages of a spherical display, so these advantages can be fully utilized to keep the users interacting, meaning that they continue to make use or learn from the information provided on the globe. Since part of the data physicalization will be spherical, the design space of spherical displays may offer valuable insights that can be utilized in the physicalization.

2.1.1. Advantages and disadvantages of spherical displays to stimulate interaction

Two important advantages and two important disadvantages come forward when reading literature that has been published on the advantages and disadvantages of spherical displays. A first advantage is the fact that a spherical display offers a unique and entertaining presentation format where data can be seen from multiple angles [4], [8]. This results in only about half of the total content displayed which Soni et al. [8] describes as an advantage, but also a disadvantage. Satriade [4] sees it also as a double edged sword. Williams et al. [9] emphasize that this can create private or shared experiences based on where a user is standing, which shows he agrees with Soni et al. [8] and Satriade [4]. A second advantage that comes forward is the fact that users can freely move around the sphere [10], [11]. Balakrishan et al [10] describes that spherical displays do not have no master user. Users are free to walk around and interact without there being a more important spot available. Hamilton et al [9] describes that the sphere enables continuous content to be shown, also emphasizing the fact that there is no master user thus agreeing with Balakrishan [10].

A first disadvantage is the interaction and content design on spherical displays can be challenging. Soni et al. [8] describe that content on spherical displays can be challenging. Bakakrishan [10] describes also the difficulty of continuous content, supporting the statements of Soni et al. [8]. The interaction should be clear and seem easy to the user, or confusion will arise [12]. A second disadvantage is the fact that the spherical display is unconventional [8]. Soni et al. [8] did an experiment where they let adults and children interact with a public display. They found that the adults had a more held back approach when interacting with the sphere. The adults also used more conventional touch-based gestures on the sphere in comparison to the children in the experiment.

All in all the fact that you can display content at every angle is a double edged sword. In addition to this, the unconventionality can work against the effectiveness of a spherical display. This means that when designing for public displays we must ensure that the user is not bothered by the missing information and feels free to interact and walk around the sphere.

2.1.2. Ques to lower barrier of interaction with a public display

The advantages and disadvantages of spherical displays offer unique and entertaining presentation formats, however the navigating user experience can be challenging. Getting an audience to interact with a public display is a task that has many variables. Social anxiety plays a big role in interaction with public displays. Wiliamson et al. [9] held an experiment with a public spherical display that said displayed the words ‘touch me’, people could have different types of touch-based interactions. They found multiple people touching and quickly moving along the sphere, which they speculated was to avoid the social embarrassment of the sphere not responding to inputs. Rogers et al. [13] also emphasize on the fact that users experience embarrassment when in contact with public displays.

The ‘honey pot’ effect is a common effect that happens at public installation. Which entails that when people are around a display, more people start to also look at the display and start interacting with it. Williamson and Sunden [14] observed exponentially more people standing around the sphere if there were already people. Brignull and Rogers [13] emphasize the observance of the honey pot effect, adding onto the commonality of the effect. Williamson et al. [9] and Hornecker and Stifter [15] also found the honey pot effect in their experiments with public installations. The fact that more people are standing around could also lower social anxiety. Hornecker and Stifter [15] also emphasize that when interactions are more embodied (use natural movement, no mouse or keyboard etc.) a wider audience is inclined to start interacting. In addition to this the perceived ease of use of a task also lowers the barrier of interaction [9], [13].

The element of challenge also contributes to lowering interaction. Hornecker and Stifter [15] found that interactive museum installations with an element of challenge increased the amount of interactions. Balakrishan et al [10] found that when elements on a spherical display were messy users would be more inclined to start interacting, the audience may perceive this as the challenge to organize. Thus supporting the observations of Hornecker and Stifter [15].

All in all, while spherical displays provide engaging presentations, navigating user interaction can be challenging. Addressing factors like social anxiety and perceived ease of use can help lower barriers to engagement with public displays. To lower the barrier of interaction one must make the perceived ease of use as low as possible and try to make users feel smart from the start to boost their confidence for interacting.

2.1.3. Reasons for users to keep interacting with a public display

To develop further on the concept of navigating user experience and interacting with public displays it is important to keep the users engaged when they start interacting with a public display. In the literature of this paper there were ten reasons that added to users interacting with a public display. Discussing them all is outside the scope of this paper. The three reasons that have overlapping similarities with other papers are discussed. First of all, predictability and usability of the system is a major reason to keep interacting. Hornecker and Stifter [15] observed that if there were problems with interaction and disfunction of the system the users would quickly stop interacting. Williamson et al. [9] observed users tapping once on their system and seeing nothing happen and walking away, adding onto the observations of Hornecker and Stifter [15]. This shows that the users want to use a system that is predictable and the amount of interaction lowers if this is not the case.

Secondly, perceived ease of use and priority play a role in interacting. Brignull and Rogers [13] state that users value their time and aren’t willing to spend time learning a system. The users in their study found it important that the system was easily usable. This shows that when a system is perceived as easy to learn users are more tempted to interact with it. This also reflects on Williamson et al. [9], if a user is confused, they stop trying to test the system further. A system must be easy to use so it can entice users to interact.

Thirdly the type of interactions plays a role when it comes to how long users will interact. Williamson et al. [9] experimented with a globe and different types of interactions. The first was the ability to only spin the globe, the second was the ability to both spin and tilt the globe. They found that the spinning and tilting let to longer interaction times. Brignull and Rogers [13] found that short but mesmerizing and playful interactions are more engaging for the user. This also explains that more types of interactions make it more playful and thus more engaging. Williamson and Sunden [14] also focused the interaction of their installation to be playful but did not specify the results of the playfulness.

All in all, interactions should be attention grabbing and playful to make it engaging. Designing spheres for public interaction should be about making a short interaction as mesmerizing as possible and accepting users in public setting do not want to invest a significant amount of time and effort learning the ins and outs of an installation.

2.1.4. Leveraging spherical displays to encourage public engagement

Throughout the research for this literature review it has been observed that there are three concepts of structural elements that can be utilized in order to leverage the full utility of spherical displays. First of all the landmarks of a display play an important role in users orienting themselves and thus fully recognizing elements of the display, making them comprehend more information. Balakrishnan et al. [10] found that making clear where the equator, north pole and south pole can help users orient themselves. However, Hamilton et al. [11] found that content placed near the north and south pole is not looked at as much as content placed near the equator. This could mean that users orient themselves, but they disregard some of the data that is given to them.

Second, the surrounding elements of the spherical display influence the way people perceive and interact with the spherical display. Brignull and Rogers [13] found that activities surrounding the display that seemed easy had a positive effect on the amount of engagement with the public display. Williamson and Sunden [14] found that attempting to make the globe seem part of a bigger system opens more ways of interaction. This could mean that when a system seems part of something bigger than just a display opens more creativity and comfortableness for users, in line with Brignull and Rogers [13] statements that if these activities seem easy it has a positive effect on the public engagement.

Third of all clearly leading the users and showing them explicitly is important for engagement and bringing information on a display in clear ways. Striadi et al. [4] state that it is important to highlight elements that are not instantly comprehendible by users, for instance highlighting bar charts with what they mean. This shows that users want to be told and not have to actively persuade information. Hamilton et at. [11] found that horizontal lines along the display encourage users to walk around it, showing behaviour of users being led around the display. This is in line with the statement that users want to be told or insinuated on what to do. Brignull and Rogers [3] foundings on perceived ease of use from the surrounding activities also add onto this statement that users don not want to think about interaction.

All in all, to fully make use of spherical displays one must understand that users want to be told what to do, this can come in the form of explicitly telling them or telling them through designing experiences that are perceived as easy.

2.1.5. Conclusion and discussion

The goal of this literature review was to explore how to persuade general public to interact with a public spherical display, so they may playfully learn about what information the display has to offer. From the research it has been found that displaying the right content might be a challenge for physical spheres. Continuous content can persuade the user to walk around the display, but choosing the continuous content can be a challenge. The fact that these displays are in public comes with the phenomena of social anxiety, which limits people to interact with the sphere. Multiple methods were tried to keep users engaged and perceived ease of use seems to be one of the most important elements when it comes to getting users to engage. Interactions should not aim to be long, keeping the user mesmerized and enabling playful interactions is more important. Multiple papers tried to put a text with 'touch me' on their display [8], [9], however a limitation is that they did not investigate the way this influenced the interactions.

A limitation of this research was the fact that the number of studies was limited. Studies from general public displays were included as well, which may contrast with some findings that are applicable only to spherical displays in public. Furthermore, only academically reviewed sources were used, but non-academic literature might also offer new insights. In addition to this the purpose of the reviewed sources was also differentiated, some were focused on non-public interactions whilst other on just public interactions. The sources also did not discuss the amount of information comprehended from the interactions. Studies that put their globe in public did not interview users after the interaction, missing out on important qualitative insights.

An interesting future research direction is to see if the spherical public displays contribute to more learning and information comprehension in comparison to traditional displays. Direct comparison has not been done. Future research in setting up a framework to analyse variables to gain an effectiveness score for spherical displays may also help create a more congruent work field related to public spherical displays. This framework and comparison can be used to study the effectiveness of spherical displays effectively and concretely.

2.2. State of the art

2.2.1. Data physicalizations

Data physicalizations aim to help people explore and get a better understanding of data using physical representations of data [6]. Physical representations can be touched and perceived in other ways than just with the eyes. This can lead to users interacting with data in a embodied way, which could potentially help increase perception and perceivability [7]. In addition to this experiencing data in 3d is perceived as easier compared to 2d representations of data [16].

Physicalizations can assist in making data more accessible as well. Physicalizations are always on and could be used as ambient displays [6]. This could bring these to the public, so the public can get a better understanding of what the physicalization is showing. This could entice them to explore the concept more in depth or leave with a greater understanding. Putting it in public places and making it easy to interact with increases the accessibility and purpose of physicalizations.

An example of a physicalization is the CairnFORM system [17], a system showing energy availability in public spaces. It is a ring-based data physicalization that encodes renewable energy variability using brightness and ring position using LEDs. It also includes expanding and receding animations on the system.

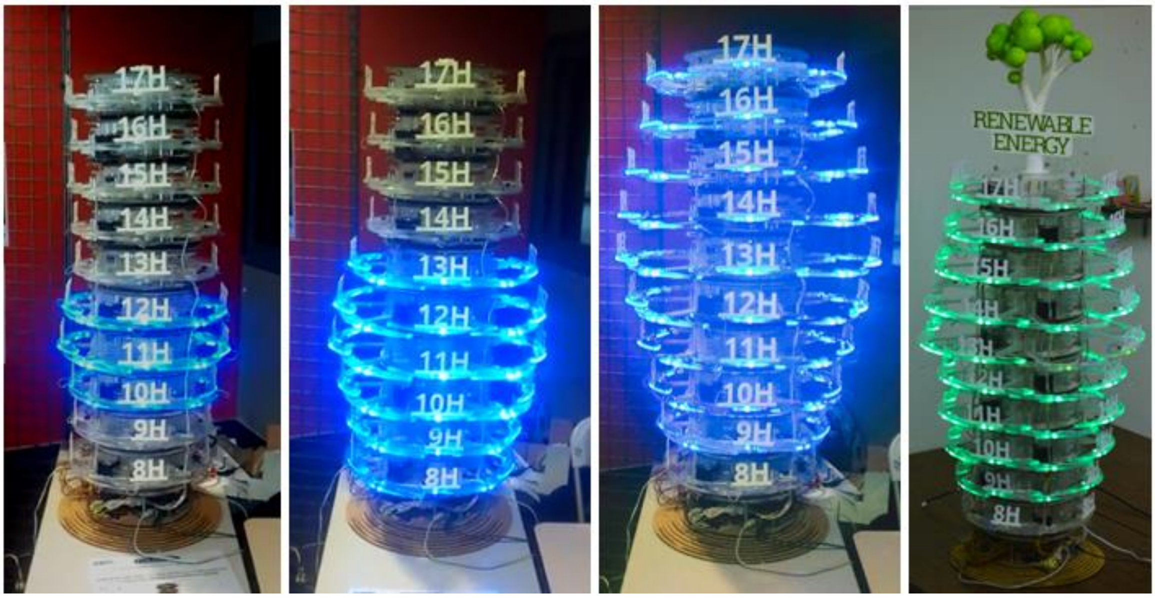


FIGURE 1:CAIRNFORM PROTOTYPE

The above figure shows the prototype that they have built, where the blue rings represent the availability of renewable energy. Their prototype was build to be used in ambient public setting. Including animations that would notify the user without annoying them was a challenge for designing the prototype[17].

Another physicalization example is the EmoClock [18]. The EmoClock attempts to physicalize emotional states using LEDs and detecting these states using biodata.

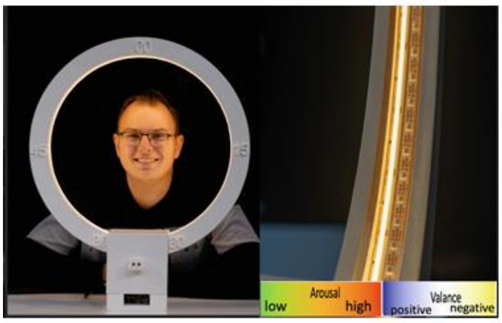


FIGURE 2: THE EMO CLOCK

In the figure above one can see the emo clock, on the right one can see the LED strip used for physicalizing the user's emotional state. It uses colours to represent arousal and valance [18].

2.2.2. Modalities, encoding, interactions

Encoding variables is a big part of data physicalizations. In the field of physicalizations numerical values are transformed into physical modalities; where modalities are sensory channels or ways te perceive data. Ranasinghe and Degbelo [19] state that understanding encoding variables is important for making physical representations of data inclusive and accessible.

The modalities that variables can be encoded into are defined clearly by Ranasinghe and Degbelo [19]. . The modalities can be encoded into physical, visual, haptic, olfactory, gustatory, sonic, and dynamic variables. One of the options for haptic variables is tangible orientation, tangible alignment. Changing the perspective of the user or changing or rotating a physicalization can influence what the user sees or perceives about the physicalization. Rotating or changing perspective is also an embodied interaction that can be utilized to make it more engaging. Leifer [20] also advocates for implicit interactions, meaning that objects should implicitly tell users how they should be used. Physicalizations that consist of rotation could benefit from implicit interaction design to make them more user friendly and accessible.

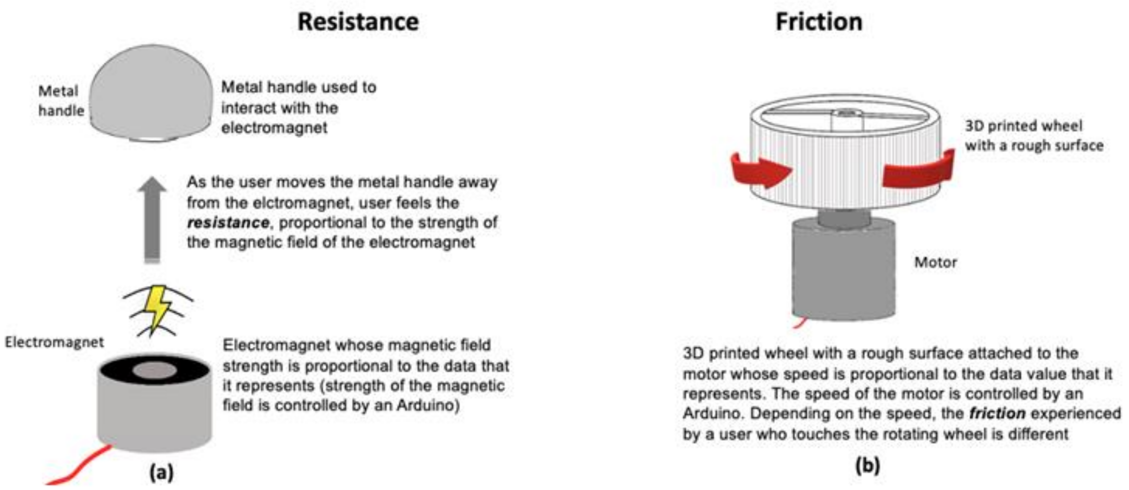


FIGURE 3: RESISTANCE AND FRICTION MODALITIES [21]

The above figure shows examples of two quite unique modalities, kinesthetic modalities to be more precise. Dullaert et al. [21] utilize electromagnets and friction cause by a motor to encode geographic data in a physicalization in order to create a more embodied experience.

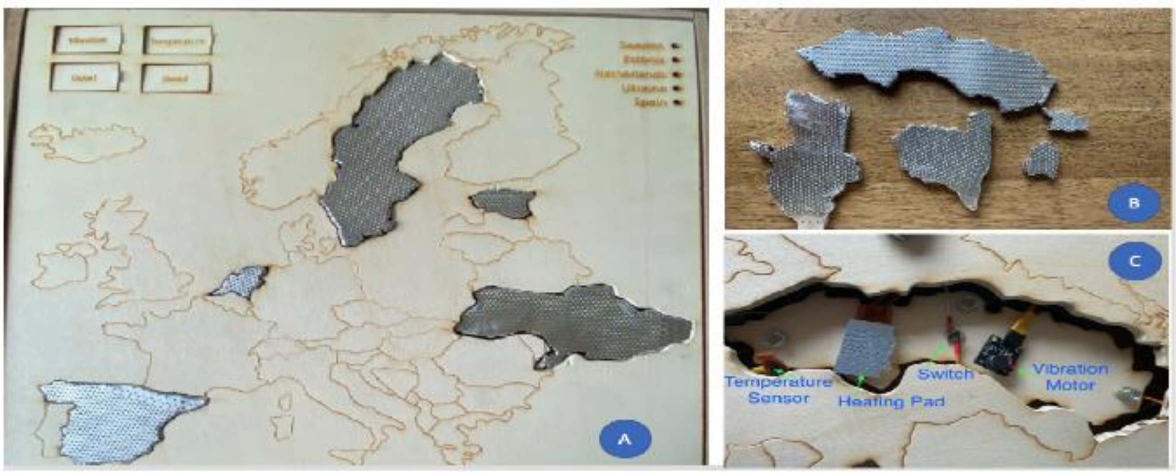


FIGURE 4: VIBRATION AND TEMPERATURE MODALITY [22]

Another example of unique modality use is the example of encoding vibration and temperature in order to show temperature data. Loenhout et al. [22] utilized this method to convey facts related to the sustainable development goal of affordable and clean energy.

FIGURE 5: INTERACTIVE GLOBE USING LEDS [23]

The above figure shows a physicalization that utilizes LEDs to communicate values. De Kreij et al. [23] made an interactive globe where one can control a slider to set the desired air pollution level. The countries on the globe that have the set pollution level will light up so the user can see the air pollution in countries. The modality of light is also discussed earlier, for the emo clock [18] uses a colour scale to show data. This is different than the application of LED usage that is used for the above globe.

FIGURE 6: AIR FLOW MODALITY [24]

Data can also be encoded in airflow, this is what Houben et al. [24] did, they developed a system which one can see in the figure above. The system has a vent and data is encoded into the speed of the vent. Causing users to feel air flowing faster if the value of the data is higher.

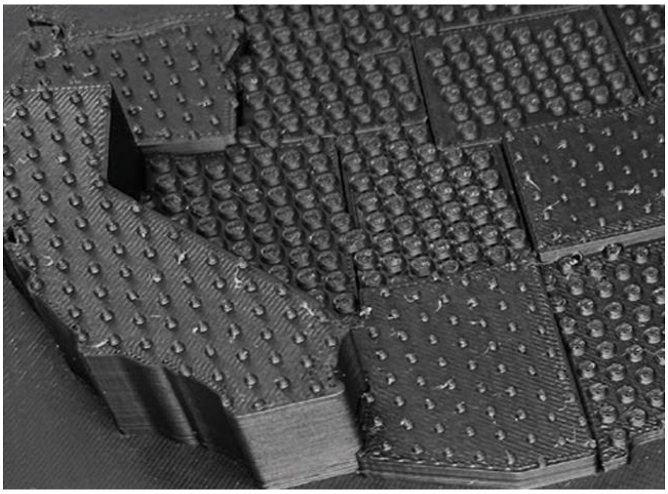


FIGURE 7: THE MODALITY OF ROUGHNESS [25]

The above figure shows a part of a physicalization that uses roughness to encode data. Du et al. [25] explores the possibility of using textures with different textures to encode data. This modality is not frequently used but shows the possibilities that physicalizations provide to show data in creative ways.

2.2.3. RESEARCH GAP

Current state of the art research on data physicalization do make use of LEDs[7], [17], [18]. However, these physicalizations do not explore animation as a way of encoding variables. In addition to this evaluating the effectiveness of the modalities that are enabled by LEDs are not explored in depth. Furthermore, a ring-based LED globe that is accessible has not been built and researched before.

3. METHODOLOGY

This chapter will explain the method which was used to generate the design of the physicalization as well as explain the initial requirements that came forth as starting point for the project.

3.1. METHOD

Initially the design of the physicalization was developed on basis what was needed for the user evaluation. In addition to this an iterative approach for the design inspired by the Creative Technology (CreaTe) design cycle [26] was used to iterate over different concepts for the final design of the physicalization.

The CreaTe approach consists of ideation, specification, and realization. It has an iterative approach, meaning that when designing the designer regularly switches between different stages to iterate on ideas or concepts. For the ideation stage of the design process, it is crucial to identify technology which can be used to satisfy the initial requirements for the user test.

During the specification phase early prototypes are made. Regarding physicalizations multiple input and output systems can be explored and evaluated, after which the product specification can be given. With this product specification a full prototype can be built in the realization phase.

In the evaluation phase the prototype is tested to see if it meets the requirements. During the user test different modalities will be evaluated in terms of the user experience, but also efficiency, accuracy, mental load, and satisfaction. Furthermore, the evaluation phase also includes the reflection and discussion of the user test.

3.2. INITIAL SYSTEM REQUIREMENTS

The initial requirements were the starting point of the ideation phase of the physicalization. The design started with the following requirements:

- The physicalization should represent climate change.
- The physicalization should have physical input types for changing the year or modality.
- The physicalization should be a globe and thus not be a flat map.
- The globe should be able to rotate.
- The physicalization should facilitate different output modes.
- The physicalization should be intuitive for the users.
- The physicalization should not have significant delays that influence the user experience.

4. IDEATION

4.1. BRAINSTORM

An initial brainstorm was done by utilizing making a mind map of possible interactions, output modes and other important elements for the design. This mind map has been digitalized and the grouping has been done by putting the different branches of the mind map into the same-coloured squares.

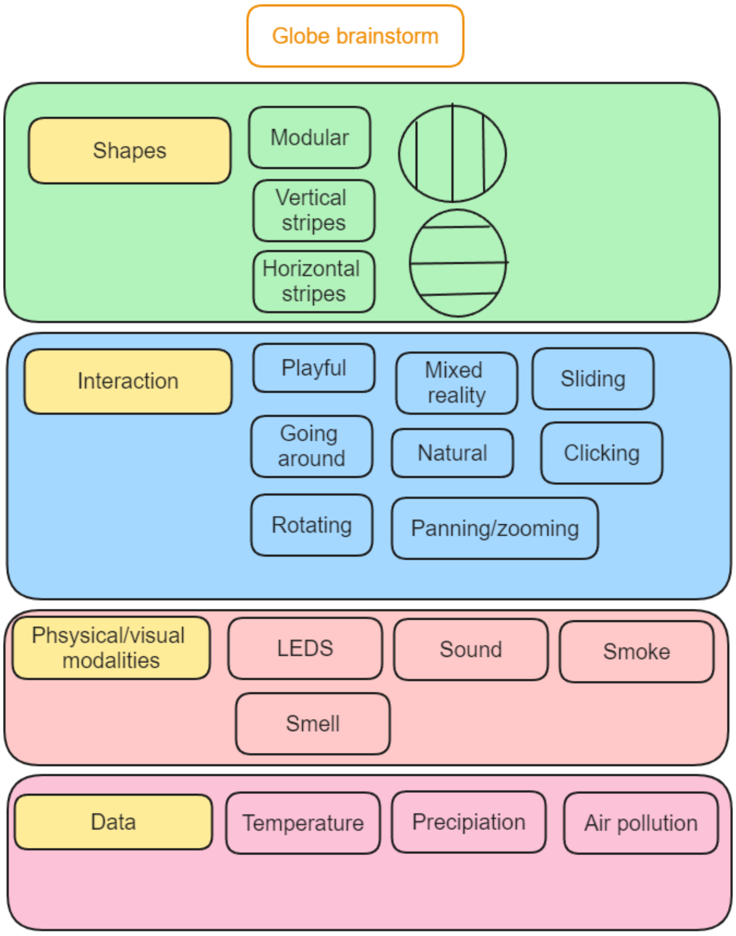


FIGURE 8: DIGITALIZED BRAINSTORM SESSION

After this brainstorm concepts were developed that tried to incorporate the different concepts into a preliminary design. The complexity of implementation was not taken into account with the following designs.

4.2. CONCEPTS

To get a general idea what the appearance and functionality of the globe would look like some simple sketches were made. These sketches gave a rough idea of where the interaction points could be placed and how the set-up of the physicalization will look like.

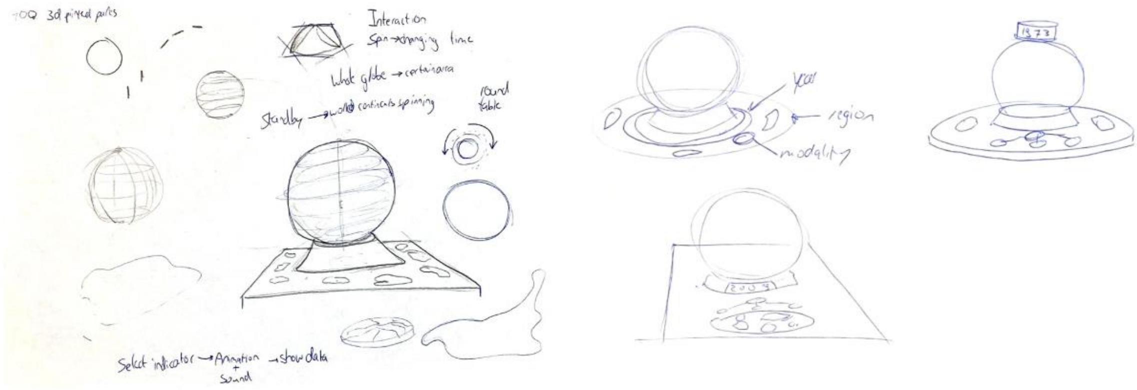


FIGURE 9: EARLY SKETCHES OF POTENTIAL LAYOUT.

In the above figure one can observe a globe with different types of tables; round and squared around it. These designs include different approaches for interactivity with the installation. One approach is to let users interact with the sphere from one side of the installation. This could discourage users to walk around the physicalization. Another approach is to put the interaction points around the installation so multiple people can interact with it at the same time and the users get more prompted to walk around the physicalization.

Because the design should be able to be tested by single users, but it was not specified if the experiment was a within group study or not, multiple options were explored and thought about.

FIGURE 10: DESIGN CONCEPTS

Five possible outputs (PO) and five possible inputs (PI) were defined and sketched out. All the above outputs were based on an LED globe where the different continents got represented in an abstract way. The globe was sliced vertically into different sections. Each continent had a different section attributed to it and was marked with letters so the user could see where the continent is at.

Modalities apart from light were also explored, like adding music or sound. In addition to this modality for selecting different type of data were also considered, however implementing this was outside the scope of the project. There is a brief description of the different input and output mode concepts:

- PO1
  - Different temperatures are represented by using a colour hue scale, so the redder, the warmer.
- PO2
  - Different temperatures are represented by using a brightness scale, so the brighter, the warmer.
- PO3
  - A threshold for air pollution is set, once the data point goes over this threshold, smoke is emitted from under the globe.
- PO4
  - Music is mapped to temperature, when the temperature becomes higher, the music starts playing louder or faster.
- PO5
  - Precipitation is represented using animation, the faster certain parts blink, the higher the precipitation.
- PI1
  - The globe is able to rotate by using one’s hands.
- PI2

    - There are miniatures of a thermometer, ice cube, a cloud, and the text “CO2”. The user can pick up a miniature from the hub and place it on a central pad, the globe will now switch modalities and you will see different data related to the miniature you put on the main pad.
- PI3
    - A sort of touch screen slider is utilized so a user can select a year by touching the corresponding spot on the slider. The years you can select are visible above the touchable slider.
- PI4
    - A rotating knob is utilized so a user can scroll through the years. The current selected year is displayed on an LCD screen.
- PI5
    - A rotating wheel is used to show the causality between CO2 emissions and temperature. When someone rotates the wheel, a simulation will start where more CO2 is added, and the globe presents a possible future scenario in which CO2 emissions and thus temperature rise.

4.3. DATA

The Copernicus Climate change database is a free database that contains multiple climate change models and historic data. This database was selected for the data that was being represented on the globe because of the ease of accessibility and the diverse datasets that Copernicus offers.

4.4. DESIGN INSPIRATION

Making an LED ball has been done and seen before. Some examples of LED spheres were looked up and gathered to ensure the feasibility of realization. This was gathered at the same time of making the sketches seen in figure 10 to validate the designs in terms of feasibility.

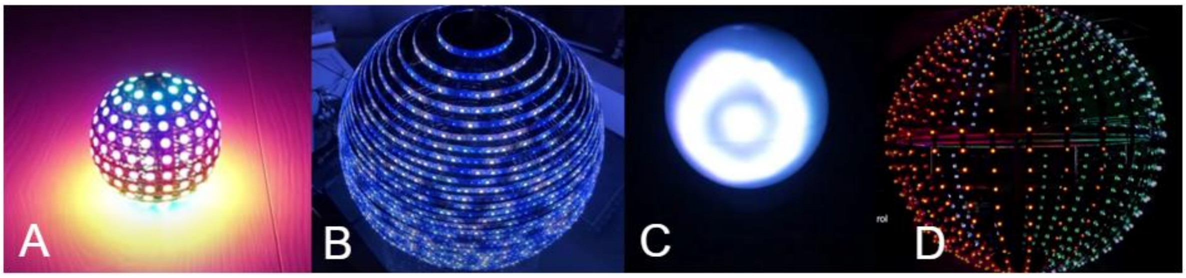


FIGURE 11: EXAMPLES OF POTENTIAL DESIGNS A: [27] B: [28] C: [29] D: [30]

In the figure above we observe four different designs. Figure A is a design where LEDs are hand soldered into spherical form. This takes a lot of manual labour. Figure B is a latitude-based design which uses LED strips glued to circles. This way the LED strips can be left intact, and this decreases the amount of manual labour. Figure D is a meridian based design. If one compares this to the latitude-based design, it is observed that the density of LEDs is significantly less in the meridian based design. This would also mean that the LEDs in the middle represent a significantly bigger area than the LEDs above, which can confuse users and decrease accuracy.

All designs include LEDs without diffusion, but diffusion would be possible by encasing the globe like in figure C. This could make the globe seem smoother and more touchable, since no electronic parts will be showing.

4.5. CONCLUSION

Some concepts were dropped because of irrelevance to the research question. It was decided to use LEDs to represent data and leave out smoke and sound so the user can focus on the output of the LEDs. The importance of accurately modelling a globe is considered. However, cannot be implemented fully due to the limitation of the LED strips that were utilized during the realization.

As for the inputs, three different types were chosen to research which one works best. These will include sliders and knobs to cycle through the years of the data. Comparing which input method works best is worked out in another research paper.

5. SPECIFICATION

This chapter will explain the requirements and potential interactions and output modes that have come forth from previous chapters.

5.1. REQUIREMENTS

The designs in figure 10 were presented to geographic researchers who wanted to give feedback. They presented a few points of feedback that led into another iteration of the physicalization. The main point of feedback was that by representing a globe in an abstract way, hereby meaning that dividing it up into sections, defeats the purpose of a globe. A globe is the most accurate representation of earth [2]. Abstracting this away would defeat the accuracy of the globe. Representing data accurately on a globe comes with some challenges so the way one assembles a globe using LEDs has influence on the accurateness of the data being presented. However, it was concluded that accurately building the globe is outside the scope of the project. This required too much research and hand soldering and fell outside of the scope of the project. After the feedback the concept was tweaked and the requirements were set up that the next iterations should follow.

As for the functional requirements; the user should be able to:

- Select the year from which the data is shown on the globe.
- Navigate through data from 1980-2023.
- Read the data from the globe.
- Distinguish different levels of temperature on the globe.
- Spin the globe.
- Know what continent they are looking at.

The system should allow for:

- Switching between output modes (Blinking, colour, blinking and colour)
- Making easy to implement changes in the way the variables are mapped and displayed.
- Fast updates of the data

As for the non-functional requirements; the system should:

- Have natural interactions.
- Embodied control so one can physically interact with the physicalization.
- Be intuitive and easy to use.
- Be able to be user tested to see the difference between different output modes.

5.2. ITERATION AND RATIONALE

5.2.1. ROTATION AND WIRING

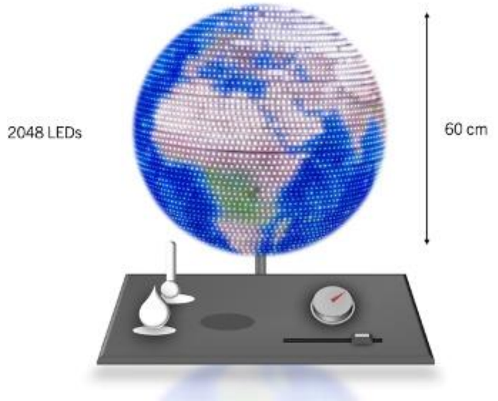


FIGURE 12: INITIAL VISUAL DESIGN OF GLOBE AND DASHBOARD

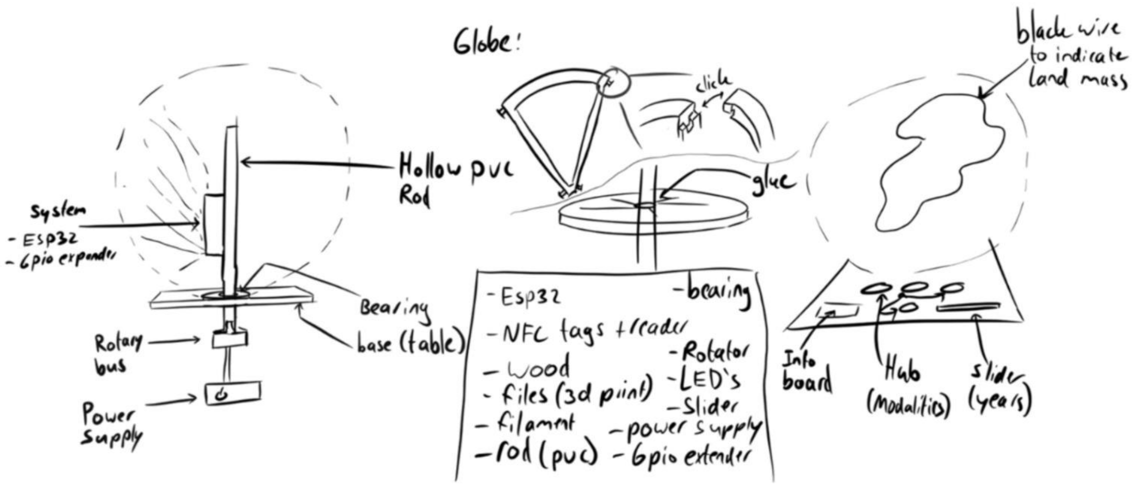


FIGURE 13: A SKETCH OF THE GLOBE DESIGN TO ENABLE ROTATION

To still achieve a bit of accuracy for the globe the ring longitudinal based design was chosen. Rings with varying diameters are put on a pole so a sort of spherical form has been made. This way the distribution of LEDs is the most equal whilst maintaining the easiness of using LED strips.

FIGURE 14: THE LONGITUDINAL BASED DESIGN

In the above figure one can observe the fusion layout of the globe using rings with varying diameters. A python script was made to calculate the number of LEDs needed for a ball. It does this by giving it the desired diameter, the desired height spacing between strips and LEDs per meter on the strip. It was decided out of budgetary reasons that only 35 meters of LED strips would be bought. Since the density was desired to be high so it would look more like a spherical display instead of rows of LEDs with a long distance in between a height margin of 1.5 cm was selected. The results yielded fitting LED amounts with a diameter of 45 cm and 32 rings. The script can be found in **appendix J**. The ring sizes that were the output of the script were put into Fusion and a laser cut template was made that fits the design.

To achieve the globe being rotated whilst maintaining untangled electrical wires he following techniques and components were implemented; a slip ring is used so the wires do not get untangled. This slip ring is put inside a bearing (see left side of the above figure). The pole that the rings will rest on will be assembled onto the bearing, so it is able to rotate together with the bearing.

5.2.2. Input methods

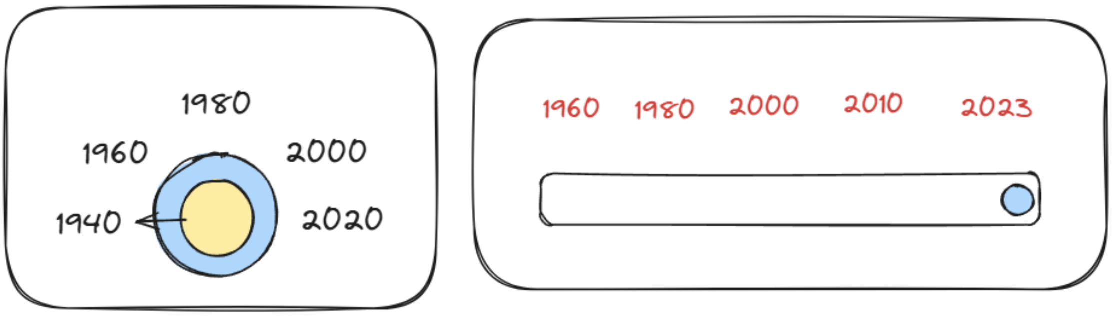


FIGURE 15:TOP-DOWN VIEW OF SLIDER AND KNOB DESIGN

The globe should have a knob that a user is able to rotate to select the year. There is a clear indication (In the figure one can see the arrow) which year is selected. In addition to this, 2 knobs will be implemented, one with haptic feedback when one selects the next position. This means that it makes a clicking sound and feel once you turn it. The other knob will not have this clicking sound. Also the slider is implemented for users to interact with. The position on the slider will correspond with the position of the year.

5.2.3. Encoding variables

It was decided that the variable of temperature anomaly was going to be encoded into LEDs. This variable had ranges from -7 until 7, but since there were a lot of outliers the range from -3 till 3 degrees Celsius was chosen. These values were going to be visually encoded in the LEDs by using a traditional colour scale, but also by using blinking.

The LEDs animation would entail blinking, where an LED goes off every X amount of seconds and then goes back on. This can be done in a variety of different patterns, like linearly increasing the brightness until it has reached its full potential or using a sine wave to smooth out the going on and off. What type of method is still to be experimented with to see what soothes the eyes most. The LED lights on the sphere should blink at it should be noticeable to see the difference between different blinking patterns. The blinking should be frequency based and thus the frequency of blinking should be mapped to variables.

The LED lights on the sphere should be able to be mapped to colour scale. This colour scale should make use of data bins for showing different temperature, thus setting up ranges of data that belong in a certain 'bin' and coupling a value to these ranges. Linear encoding is not ought to be used, since colour bins have shown to make small differences more noticeable for users of physicalizations. In addition to this making it more perceivable also means simplifying the data a bit by putting it in containers.

5.3. Designing colour maps and blink speeds

Since the user test will revolve around the perceivability of different types of output modes it is of utmost importance to design these so they may be perceived properly by users. This section explains what key findings were found during the development of these output modes. An important note is that the min and max values of the data were set at -3 and 3 degrees respectively. This was done because this made the information more perceivable, and the Copernicus Climate Atlas implemented the same min and max values as well for showing their temperature anomalies.

5.3.1. Blinking perceivability

The blinking animation was tested by setting the ball to a certain colour and then applying data bins and frequencies. Like described earlier, the frequency is a number that gets multiplied by 100, the number that comes out is the time in milliseconds between each blink. The table below will showcase findings during the iteration process of configuring the blink speeds. The format of encoding data bins and frequency is the following:

[bin_start, bin_end, frequency]

The colour has a maximum brightness of 25, unless specified that it has not. The temperature anomaly can be both positive and negative, however for the frequencies we take the absolute value, since showing negative numbers with blinking would be confusing. This can be encoded in binary colour information, limiting colour scale but still giving users necessary information.

Bins	Colour	Observations
[0,1,100],[1,2,70],[2,3,50]	White	When distinguishing between the 3 bins it is easier to observe different groups of LEDS going off at the same time, rather than looking at the frequency.
[0,1,150],[1,2,70],[2,3,10]	White	The fastest blinking is unpleasing to the eye
[0,1,170],[1,2,130],[2,3,50]	White	Spreading the frequencies do not make the frequencies easier to look at, it is just about looking what regions turn on and off at the same time.

[0,1,170],[1,2,130],[2,3,50]	Red	The red colour is less contrasting than white, so it is harder to see what regions turn on and off. This could be caused by the red light emitting less lumen than the white light.
[0,1,170],[1,2,130],[2,3,50]	Green	The green colour is quite clear, clearer than red. It is the easiest colour to see contrasting.
[0,1,170],[1,2,130],[2,3,50]	Yellow	It is brighter than red/green. This is good for contrast, however whilst the LEDs are in the process of dimming it is hard to distinguish between the groups/bins.
[0,1,170],[1,2,130],[2,3,50]	Blue	The contrast with blue turning on and off is about the same as red.
[0,1,170],[1,2,130],[2,3,50]	White (100 brightness)	This white is more contrasting, because it is brighter. This impacts the contrast more than other colours. It would mean that brightness plays a bigger role than colour.

TABLE 1: DESIGN PROCESS BLINKING PART 1

The above table mainly compared different colours; the conclusion can be taken that changing the maximum brightness. In addition to this it is still quite hard to distinguish between blink speeds. It is easier to look at what regions turn on and off at the same time. The next table explores options for making the frequency distinguishable. Its colour is just white at a maximum brightness of 25.

Bins	**Observations**
[0,1,200],[1,2,100],[2,3,50]	In theory these bins happen in sync with each other, because they are multiples. This causes the whole ball to be lit at the same time, you can clearly see what frequency happens most often. The slower frequencies are hard to distinguish from each other.
[0,1,250],[1,2,100],[2,3,50]	Now the frequency of the first bin is out of sync with the second bin, but in sync with the third bin. It is clear which regions belong to the last bin and if you concentrate hard, you can see that the last first bin turns off very slowly. If you concentrate well, you can also see the middle frequency turning off once and again.
[0,1,250],[1,2,118],[2,3,50]	Now the middle bin is out of tune with the rest. The first and last bin are in tune with each other. The middle bin turns off at its own time, making it quite distinguishable. However it is observed that the first bin (slowest) is a bit unclear, since you have to wait quite some time for a blink.
[0,1,120],[1,2,40],[2,3,10]	The frequency of these bins are spread exponentially. This means that they are really far apart. The fastest frequency is

	unpleasing to the eye, however it does not occur that often so it gets compensated by less intense blinking.
[0,1,180],[1,2,60],[2,3,20]	Now the bins are spread exponentially, but the slowest frequency takes some time to turn off. Making you have to wait for it and this requires a lot of concentration.
[0,1,100],[1,2,60],[2,3,20]	Now waiting time for the slowest frequency is reduced, this improves clarity.
[0,0.5,100],[0.5,1,70],[1,2,40],[2,3,20]	Experimenting with multiple bins, you can clearly see the fastest and the slowest bin, however the middle ones are hard to distinguish.
[0,0.5,150],[0.5,1,80],[1,2,30],[2,3,20]	It is unclear how many different frequencies there are.
[0,1,100],[1,2,60],[2,3,20]	In the end it was decided that this mix of frequencies are most distinguishable and not agitating to the eyes.

TABLE 2: DESIGN PROCESS FOR BLINKING PART 2

The above table concludes that frequencies that are far from each other work better than frequencies close to each other in terms of recognizability. Differentiating with the frequencies is hard, hence only three different ones were chosen. It is easier to see which ones are going on and off at the same time rather than watching the individual blink speeds. Also, slow blink speeds require concentration and some more time before being able to recognize them. Because the users in the user study require to answer questions related to increase or decrease of temperature it was decided that for the 'animation' mode, so no big colour scale, a binary colour scale was implemented. So, if a value is higher than 0 it turns red, if lower blue. This lets the user still see the difference between warm and cold, however the user needs to get reliable information from the blink speed.

5.3.2. COLOUR SCALE

After the animations were decided the colour scales were developed. It was decided to put the colours into 5 bins. The blinking was put into 3; almost no change, a bit of change and a lot of change. When using 5 colour bins one can simultaneously use them together with blinking. At first the bins were set as follows:

[-3, -2], [-2, -1], [-1,1], [1, 2], [2, 3]

However, this caused a problem: the majority of the globe turned white and it was not usable for user testing. Hence the bins of colours were changed to:

[-3, -2], [-2, -0.5], [-0.5, 0.5], [1.5, 2], [2, 3]

The bins for animation were therefore changed to:

[0, 0.5], [0.5, 2], [2, 3]

The change was done to ensure that when combining colour and blinking the user wouldn't be mislead because the frequencies were put into a different bin than the colours. In the end a certain set of colour bins were decided. They are in the following format:

[red value, green value, blue value]

The maximum value is 255 for RGB, however do keep in mind that there is a maximum brightness of 25 for the globe to make it bearable to the eyes, thus all values are multiplied by **25/255.**

[0, 0, 255], [50, 50, 150], [50, 50, 40], [75, 75, 0], [255, 0, 0]

The colours from left to right are: Dark blue, light blue, white, yellow, red. It is important that the contrast between the dark blue and light blue was visible. A significant amount of red and green light was therefore added to the light blue so it would appear drastically lighter than the dark blue. This caused some contrast issues with the white light; thus, the blue value of the white light was tuned down. This caused white to still seem white, however the contrast with light blue was increased a bit. The yellow light was easier to distinguish with red. Orange was tried as well, but it was hard to distinct red and orange. Furthermore, the red on the LED lights themselves have a bit of an orange tint to them as well.

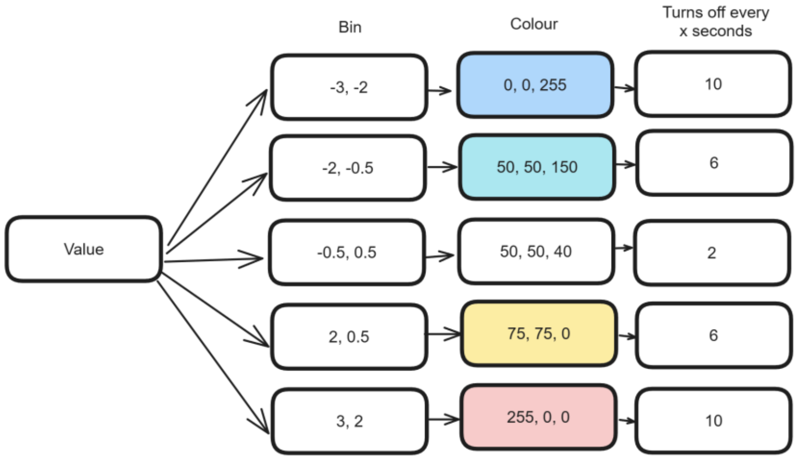


FIGURE 16: MAPPING OF DATA TO OUTPUT MODES

In the above figure one can observe the mapped data, bins, colour and frequency. The above table shows the mapping that was used in the experiment. In the experiment 3 modes were used; just colour, colour and blinking or only blinking. When the lights are only blinking a binary colour scale is used with red indicating warmer and blue indicating cooler.

6. REALIZATION

6.1. SYSTEM

This section will explain the realization of the system. It starts with a general overview of the physicalization and goes more in depth on the specific parts of the system.

6.1.1. GENERAL OVERVIEW

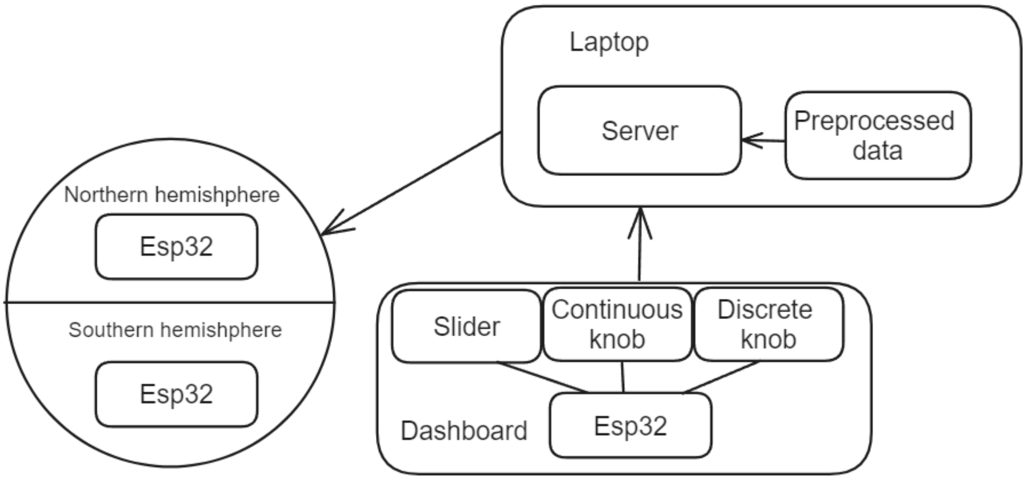


FIGURE 17: SCHEMATIC OF SYSTEM DESIGN

In the figure above we can see a layout of the physicalization systematically. The physicalization consists of four parts; the two hemispheres that make up the globe, the dashboard, and the server. The hemispheres are connected to rows of LEDs. Both the dashboard and hemispheres are powered by a computer power supply supplying 10 Amps at 5V and communicate wirelessly with the laptop. On the laptop there is some pre-processed temperature anomaly data that is able to be directly outputted to the hemisphere. Note that the globe does not communicate back to the server, it only wait for data to be sent.

6.1.2. CONTROLLERS

The system is powered by three microcontrollers and a laptop. The microcontrollers that were chosen are the ESP32-WROOM microcontrollers, because of the built-in wireless communication support. One microcontroller drives the upper hemisphere, the other microcontroller drives the lower hemisphere of the globe. The third microcontroller is reading the inputs from the various knobs and the slider. The approach of multiple microcontrollers was chosen to make the system more modular. Also reducing the number of tasks one microcontroller must do could improve the wireless communication speed. All these microcontrollers send information to the laptop and receive information from the laptop. They do not directly communicate with each other.

6.1.3. Physical design

The physical globe consists of a skeleton which the LED strips are glued onto. This is based on the latitude design. The skeleton is made by laser cutting 10mm thick wood. Designs were made using Fusion. In the end the skeleton had a diameter of 45cm.

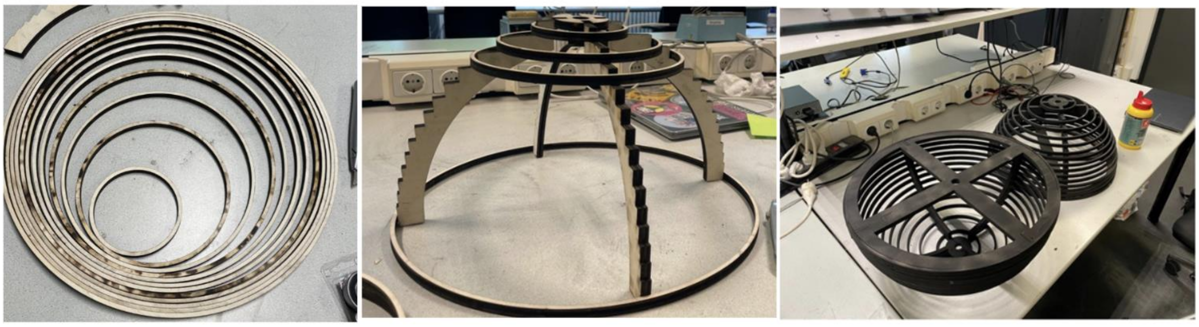


FIGURE 18: RING SKELETON FOR THE PHYSICAL DESIGN

In the figure above one can see the rings of different sizes, stacked at different heights that make up the hemisphere. The hemispheres were spray painted black when finished. In the next stage of the realization the LED strips were put on the skeleton and wired to the microcontroller using adhesive strips that the LED strips come with. For the LED strip the WS2812b strips were chosen, because of their ease of use and modularity. They can be cut into different lengths and are easy to program. Each hemisphere consists of 991 LEDs and thus the entire globe consists of 1982 LEDs spread over 32 rings.

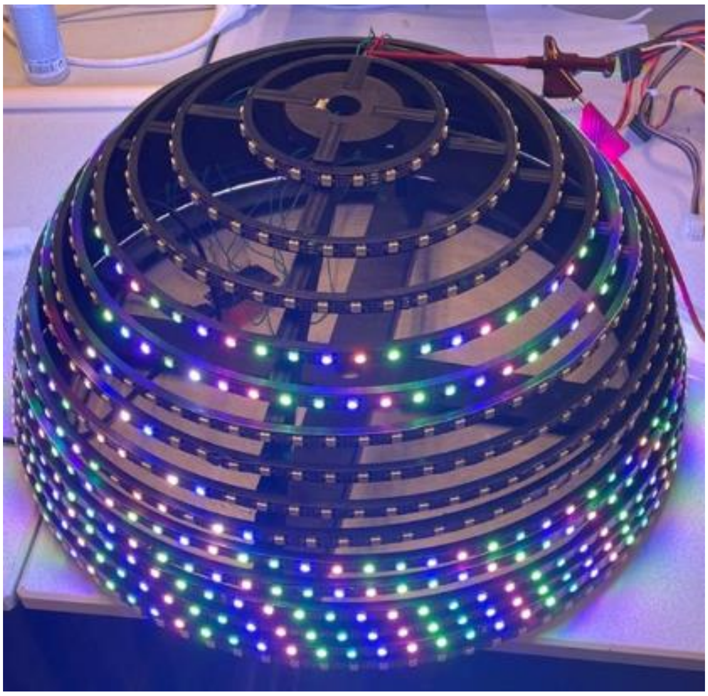


FIGURE 19: ASSEMBLED HEMISPHERE

In the above figure one can observe the assembled hemisphere with connection to the microcontroller.

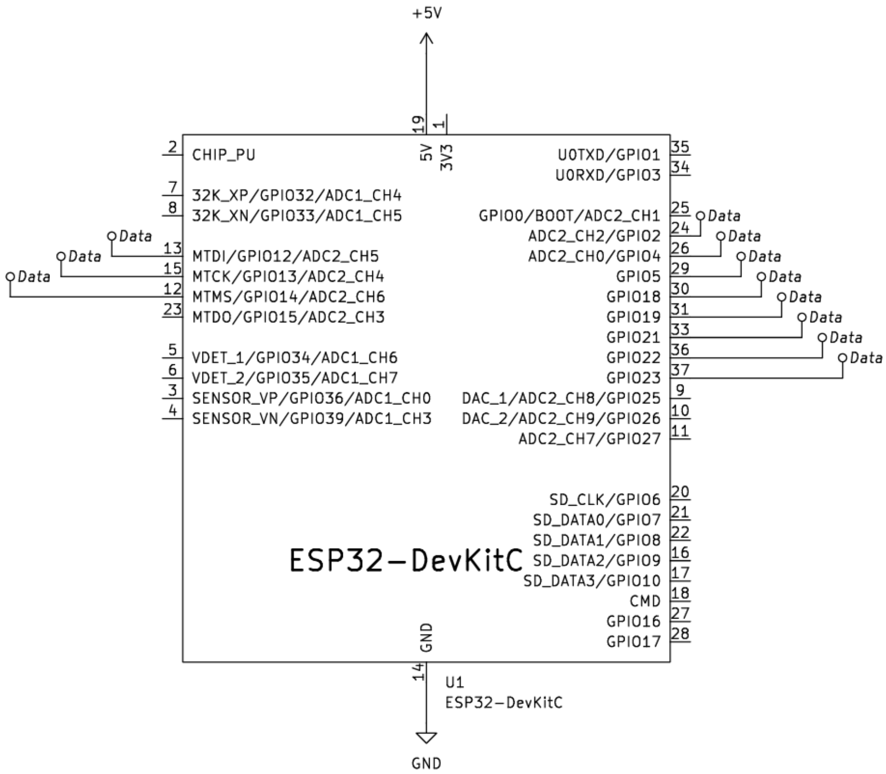


FIGURE 20: SCHEMATIC FOR HEMISPHERE

The above schematic is used for a single hemisphere, used in both hemispheres. The ground and power are connected to the ground and power of the power supply. There are 11 data wires connected to different LED strips.

There are 16 rows of LEDs on each hemisphere, but only 11 data wires are used to control them. This is because some of the LED strips are connected to each other to minimize the number of pins required. The Adafruit Neopixel library controls the LEDs. Each LED strip needs its own data wire. However, it’s unclear from the documentation how many LEDs can be connected in a single strip while still functioning properly with the library. To ensure reliable operation, the design uses multiple shorter LED strips instead of one long strip.

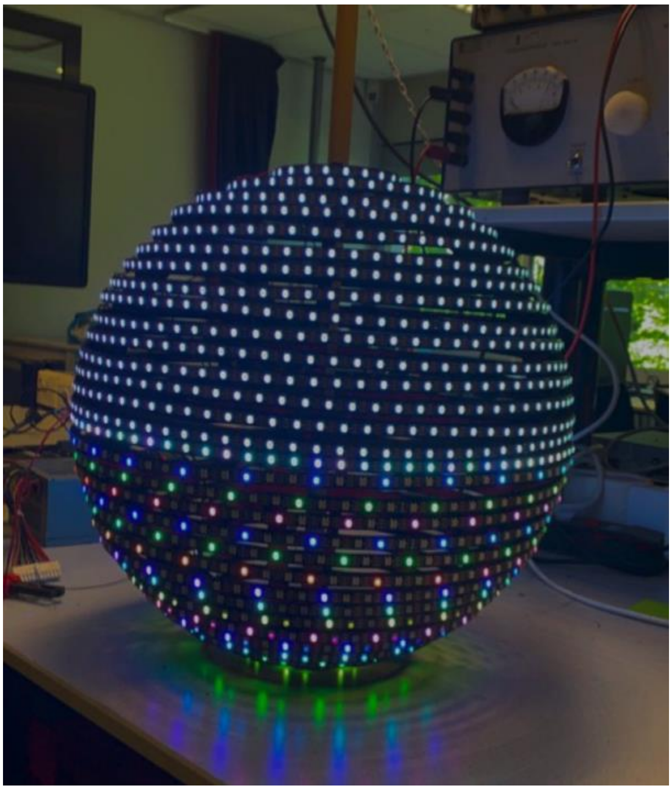


FIGURE 21: ASSEMBLED GLOBE WITHOUT DASHBOARD

The hemispheres are put together and assembled, being held in place by a PVC pipe. This concludes the globe for the most part, however it needs to rest on a dashboard. A box was cut from wood with a laser and contained a mechanism with a bearing and slip ring to ensure that the globe can be rotated, and the globe is able to be powered with a supply and thus does not rely on batteries.

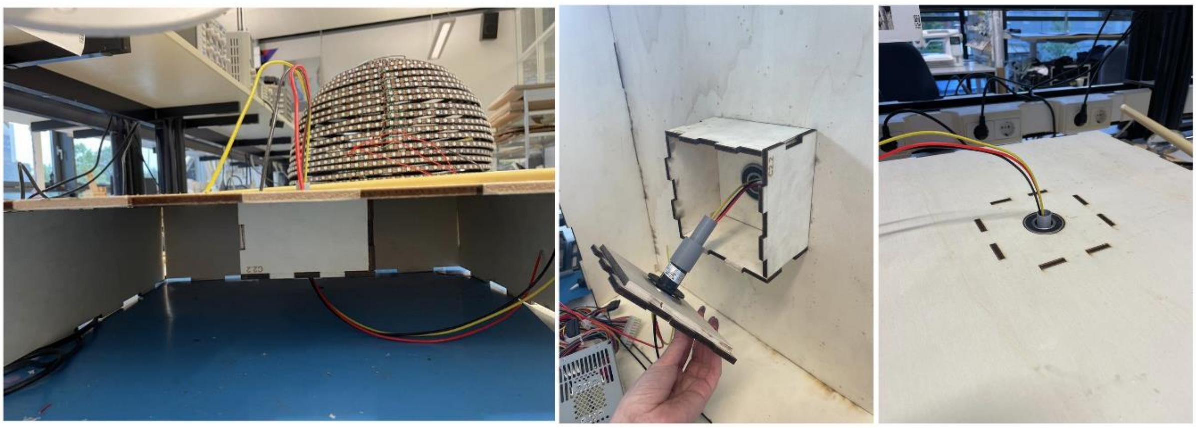


FIGURE 22: ROTATION MECHANISM

In the above figure on the left one can see the box that serves as the dashboard, side panel open, no bottom. The cables go in and out and can rotate because of the component in the middle picture. There is a 3d printed cylindrical component attached to the slip ring to ensure it can fit in the bearing. The bearing is attached on the top of the dashboard (see right picture). This way the PVC pipe can be attached to the bearing and slip ring, wires can be fed through the tube. The tube will be able to rotate without causing tangles.

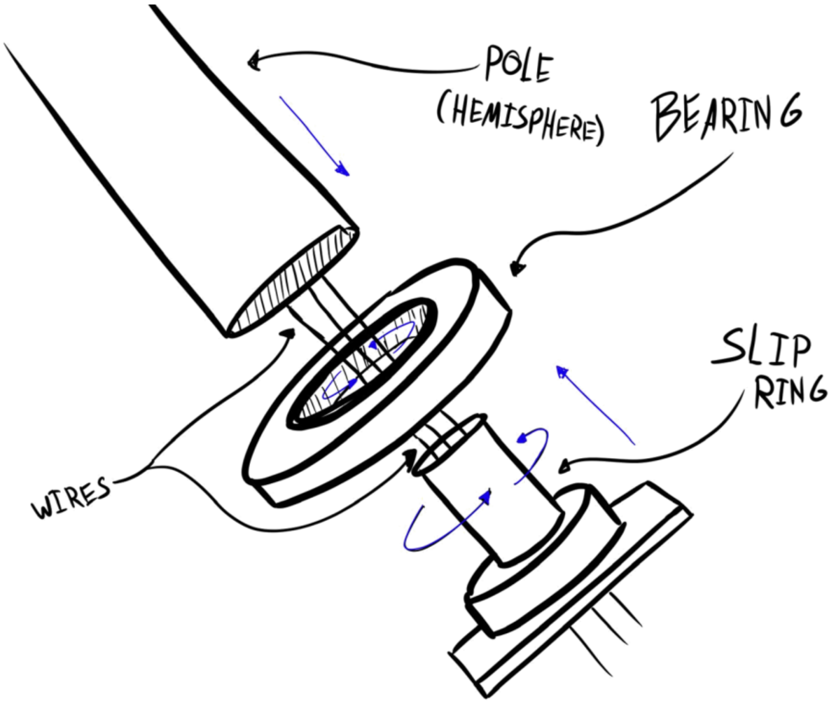


FIGURE 23: THE MECHANISM FOR ROTATING DRAWN OUT

The mechanism ensures that the globe can be rotated indefinitely. This mechanism serves a major requirement; users must be able to rotate the globe, since this is the natural embodied interaction that are enticed by globes.

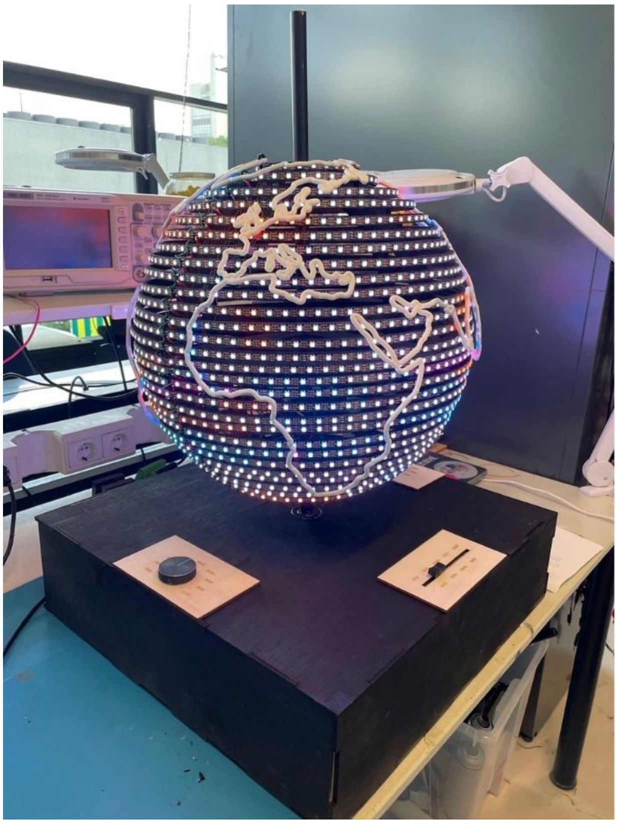


FIGURE 24: THE ASSEMBLED PHYSICAL GLOBE

Continents were 3d printed using PLA material and placed on the globe to ensure one can see what continent they are looking at. The dashboard has holes so the dashboard items (slider, continuous knob and discrete knob) can be connected properly. The different elements work like potentiometers and are powered by the 3.3v output of the power supply, since the ESP32 analog-input channels will only measure up to 3.3 volts.

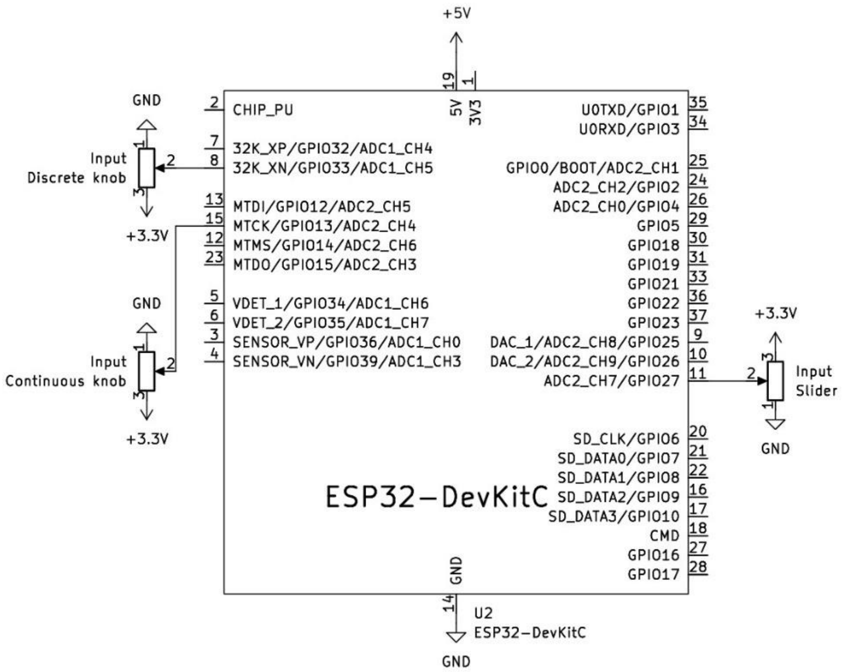


FIGURE 25: SCHEMATIC FOR DASHBOARD

The code for the dashboard controller can be found in **appendix E.**

6.1.4. WIRELESS COMMUNICATION AND SERVER

To incorporate smooth communication between the microcontrollers and the laptop WebSockets are used. Websockets is a communication protocol that enables real-time, two-way communication between server and client. The laptop is the server in this case.

The microcontrollers for the upper and lower hemisphere only receive data and output this directly to the LEDs that are connected to the microcontrollers. The laptop sends 8-bit integers directly over web sockets as bytes, which the microcontroller receives and saves. Control mechanisms have been put into place, so the microcontroller only saves the array of integers if it receives a full array of 3984 integers. See section 6.1.6 for how these numbers are used inside the hemispheres. The microcontroller that is used for the input methods sends the selected year of all the knobs to the laptop. See section 6.1.6 for how the microcontroller processes inputs.

A laptop hosts a websocket server using the Flask-Sock library. This library is an addition to the Flask library, which is a simple web framework for hosting webservers in python. The server not only hosts the communication between laptop and esp. But it also hosts a site which lets the researchers tweak variables that affect the physicalization. These variables include:

- Data bins for frequency.
  - Which ranges correspond to what value of frequency?
- Data bins for colours.
  - Which ranges correspond to what colour bin?

- Colours per bin.
    - What is the colour of a certain bin?
- Which input mode is activated.
    - Is the slider, continuous knob, or discrete knob active?
    - A between study approach was decided, thus it was necessary to disable other forms of input modes.
- Which output mode is activated.
    - Is the blinking, colour or both active?
    - A between study approach was decided, thus it was necessary to disable other forms of output modes.
- Maximum brightness
    - What is the maximum brightness of LEDs?
    - When using maximum brightness, it is blinding and thus annoying and user unfriendly to look at
- Max and min values
    - The dataset contains different values, however when operating on the whole range the data becomes unreadable. To filter out outliers that are negligible one can input a minimum and maximum value that each data point must have.

FIGURE 26: THE DASHBOARD TO CONTROL VARIABLES

In the above figure one can observe the webserver and its different inputs. It also includes a slider to control the years. This was used to debug and test if the outputs were working before the building of the dashboard was finished. Once settings are set the researcher can press send and the globe will be updated and configured accordingly. Code for the webserver can be found in **appendix A**.

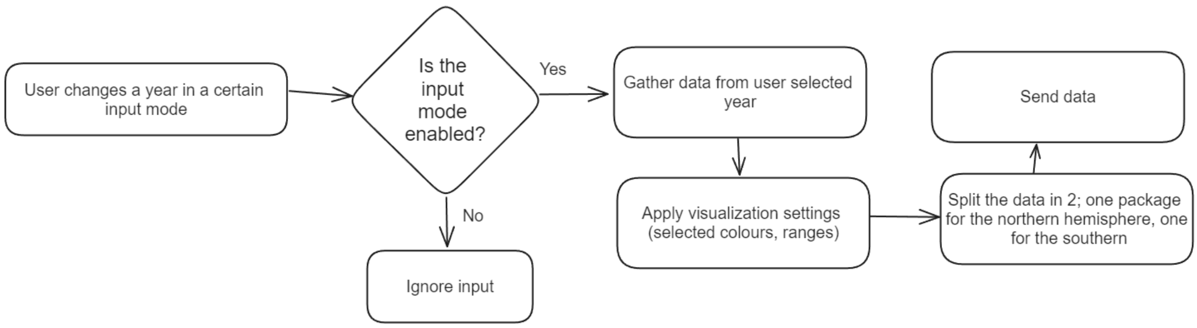


FIGURE 27: SCHEMATIC OF SERVER LOGIC

The server functionality is simplified in the above figure. It waits for input from a user, checks if that input is enabled. When the input is enabled, it gathers data from the year that the user just selected. It then applies the different settings to the data (colours and bins etc.). It also converts the data to an array of RGB and blink frequency values, so it does not send raw values to the globe. All visualization logic is handled on the server. It works with data for the entire earth, but since the globe consists of 2 hemispheres the data needs to be split. Each packet is sent to the corresponding sphere.

6.1.5. DATA CONVERSION

For the database the Copernicus Climate Data store (CDS) was used. This provides a way to download temperature data. In the end it was decided to implement the ERA5 temperature anomaly database from 1980-2023. ERA5 is a climate change model and system that contains data. Temperature anomaly is the deviation from a mean temperature. So, if a value is 3, that specific location is 3 degrees warmer than the mean. It can also be colder than the mean. The mean is calculated over a 30-year period from 1980 until 2010. This period was chosen for this specific database, but Copernicus supplies data which have a different mean.

The yearly mean anomaly was downloaded from every year in the database. Every year is one file. This came in .GRIB format, which is a common format for weather and earth data. A grib file from the CDS contains some metadata and a 2d array with the temperature anomaly values. This array had latitudes on the x-axis and longitudes on the y-axis. The longitudes and latitudes had increments of 0.25 degrees.

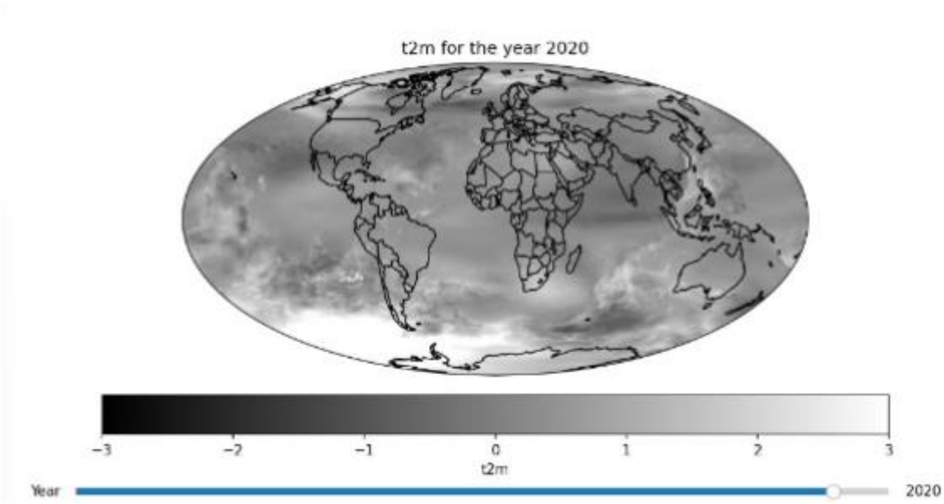
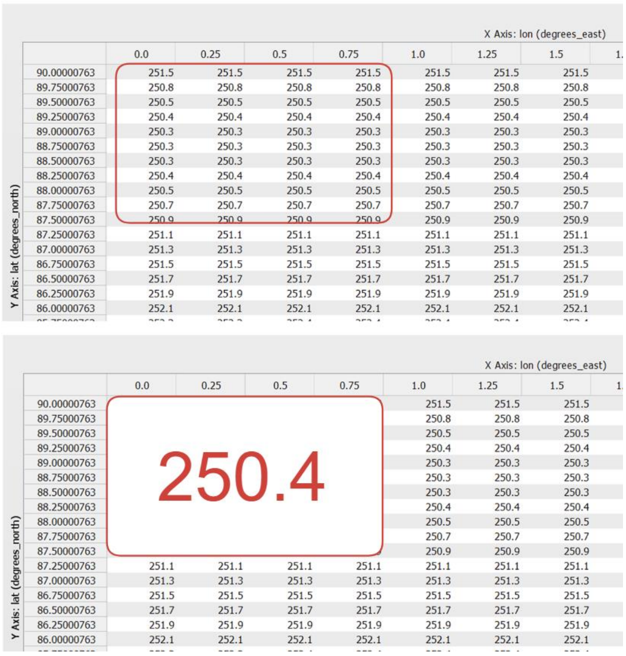


FIGURE 28: THE GRIB FILE DATA IN A VISUALIZATION

In the above figure one can observe a visualization of the data using matplotlib. Using the slider below one can manipulate the year of the data. The python script for this is provided in **appendix D.** The processing power to visualize this and scroll through the years was so high that there was a lot of delay when moving the slider. Luckily the low resolution of the globe would mean that near real time interaction is possible. Although this means that converting the grib files from resolution in real time would not be possible. This ‘scaling down’ would be too computationally intensive.



X Axis: lon (degrees_east); Y Axis: lat (degrees_north)

	0.0	0.25	0.5	0.75	1.0	1.25	1.5
90.00000763	251.5	251.5	251.5	251.5	251.5	251.5	251.5
89.75000763	250.8	250.8	250.8	250.8	250.8	250.8	250.8
89.50000763	250.5	250.5	250.5	250.5	250.5	250.5	250.5
89.25000763	250.4	250.4	250.4	250.4	250.4	250.4	250.4
89.00000763	250.3	250.3	250.3	250.3	250.3	250.3	250.3
88.75000763	250.3	250.3	250.3	250.3	250.3	250.3	250.3
88.50000763	250.3	250.3	250.3	250.3	250.3	250.3	250.3
88.25000763	250.4	250.4	250.4	250.4	250.4	250.4	250.4
88.00000763	250.5	250.5	250.5	250.5	250.5	250.5	250.5
87.75000763	250.7	250.7	250.7	250.7	250.7	250.7	250.7
87.50000763	250.9	250.9	250.9	250.9	250.9	250.9	250.9
87.25000763	251.1	251.1	251.1	251.1	251.1	251.1	251.1
87.00000763	251.3	251.3	251.3	251.3	251.3	251.3	251.3
86.75000763	251.5	251.5	251.5	251.5	251.5	251.5	251.5
86.50000763	251.7	251.7	251.7	251.7	251.7	251.7	251.7
86.25000763	251.9	251.9	251.9	251.9	251.9	251.9	251.9
86.00000763	252.1	252.1	252.1	252.1	252.1	252.1	252.1

X Axis: lon (degrees_east); Y Axis: lat (degrees_north)

	0.0	0.25	0.5	0.75	1.0	1.25	1.5
90.00000763					251.5	251.5	251.5
89.75000763					250.8	250.8	250.8
89.50000763					250.5	250.5	250.5
89.25000763					250.4	250.4	250.4
89.00000763					250.3	250.3	250.3
88.75000763					250.3	250.3	250.3
88.50000763					250.3	250.3	250.3
88.25000763					250.4	250.4	250.4
88.00000763					250.5	250.5	250.5
87.75000763					250.7	250.7	250.7
87.50000763					250.9	250.9	250.9
87.25000763	251.1	251.1	251.1	251.1	251.1	251.1	251.1
87.00000763	251.3	251.3	251.3	251.3	251.3	251.3	251.3
86.75000763	251.5	251.5	251.5	251.5	251.5	251.5	251.5
86.50000763	251.7	251.7	251.7	251.7	251.7	251.7	251.7
86.25000763	251.9	251.9	251.9	251.9	251.9	251.9	251.9
86.00000763	252.1	252.1	252.1	252.1	252.1	252.1	252.1

FIGURE 29: SCALING DOWN FILE

The globe has 32 rings with each of the rings containing a different number of LEDS. A script was built to divide the latitude into 32 different bins. Each bin would also be divided into longitude bins, there are as much longitude bins for a certain ring as number of LEDs on that ring. This way there are ‘boxes’ in which its different values from the high-resolution data can be fit. The values from the GRIB file were scaled down and linearly interpolated to ensure smoothness. In the above figure the value in the red lined square represents the scaled down value. Going to the right there would be as many boxes as there are LEDs on the first row. Going down a box you would end up in the second row, there will be as many boxes as there are LEDs in the second row positioned on the right. This downscaled data could be visualized to see the consistency with the high resolution data.

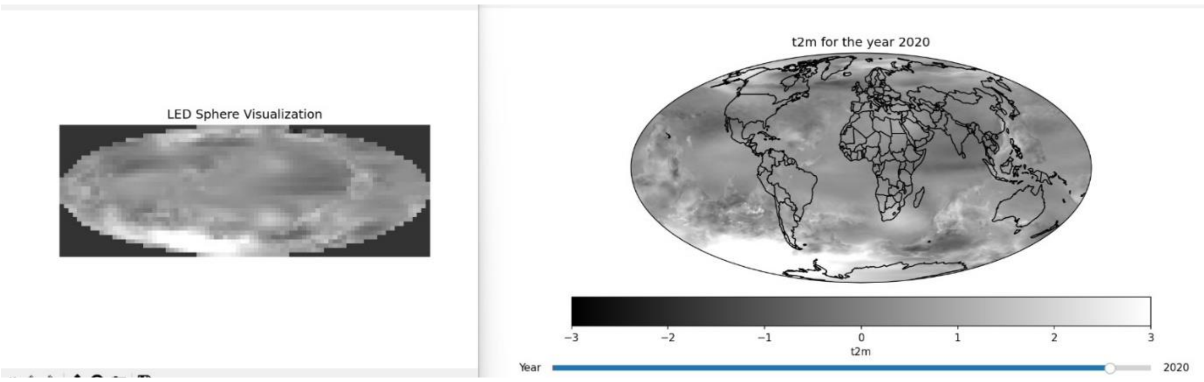


FIGURE 30: VISUALIZATION OF THE SCALED DOWN DATA

A script has been built to showcase the downscaled data, found in **appendix C**. In the above figure one can see that the images look quite similar, although the resolution is different. Each square in the right side represents an LED on the physical globe, with exception of the dark grey areas in the corners. If one were to rip out the led strips of the globe and put them below each other and centrically align them it would look like the left side of the above figure. The downscaled data was then saved as a JSON file containing python dictionaries with values. The script that downscales data and saves it as JSON can be found in **appendix B.** This downscaled data could be loaded very fast which meant real time interaction is not limited by the size of data. It is saved as JSON files in the server folder, so the server does not use a database but accesses the data using the local storage directly.

6.1.6. SETTING COLOUR AND BLINK SPEED

As stated earlier, the server processes the data values and gives RGB and frequency values to the controller. The server sends a big array of 8 bit integers, each LED gets 4 integers; a value for red, green, blue and a value for frequency. The frequency value is a value from 0-255 and is multiplied by 100 on the microcontroller. This value is the amount of time between blinks. 0 means it is always on, 10 would mean it goes on and off every 1000 milliseconds, 200 would mean it goes on every 20000 ms.

The way that the controller handles the dimming is with the following function:

```
int r = ledData[index++];
int g = ledData[index++];
int b = ledData[index++];
int interval = ledData[index++] * 100;
float dimmingFactor;
if (interval == 0) {
  dimmingFactor = 1.0;
} else {
  int phase = currentTime % interval;
  int halfInterval = interval / 2;
  if (phase < halfInterval) {
```

```
          dimmingFactor = (sin((float)phase / halfInterval * 2 * PI) + 1) / 2;
        } else {
          dimmingFactor = (sin(((float)phase - halfInterval) / halfInterval *
2 * PI) + 1) / 2;
        }
        if (dimmingFactor > 1.0) {
          dimmingFactor = 1.0;
        }
      }

      // Apply the dimming factor to the color components
      r = r * dimmingFactor;
      g = g * dimmingFactor;
      b = b * dimmingFactor;
```

The above function runs for every LED. A dimming factor is created based on the current time in milliseconds. Then it is put through a sine function to ensure that the animation is smooth. It was observed that the blinking animation without sine function was quite rigid. In addition to this Daniel et al.[17] found that in their physicalization which made use of animating rings a non-constant speed was slightly preferred. The sine made sure that the changing lights were not as rigid and constant as the approach without using the sine. The calculation was mainly done using integer operations, since the float operations took more time and decreased the real time feeling of the system. The rest of the code that controls the LEDs can be found in **appendix F**.

7. EVALUATION

7.1. EXPERIMENTAL DESIGN

A user study was conducted with different input and output methods for the globe. This evaluation focused on three potential outputs for showcasing climate change data (temperature anomaly): Colour scale, blinking scale and combining colour and blinking.

7.1.1. GOAL OF EXPERIMENT

The goal of the study is to compare the effectiveness, efficiency, user experience, mental load, physical load and user satisfaction of the three outputs: Colour, blinking, colour and blinking. Furthermore, the study tries to evaluate user engagement and assess how well the globe conveys climate change.

7.1.2. STUDY DESIGN

The study's design allowed both researchers to leverage the results for their individual analyses. Each input method was evaluated in conjunction with three distinct output modalities, creating nine possible input-output combinations. To reduce learning effects, a between-subject design was implemented, with participants assigned to only one specific combination of modalities. The goal was to have an equal number of participants testing each input method.

7.1.3. VARIABLES

The dependent variables are measured during the experiment to determine the effect of the independent variables. The table below shows the different dependent variables and how they were measured.

Dependent variables	Measuring Technique
Efficiency	The amount of time users take to answer the questions in the questionnaire.
Effectiveness	Number of accurate answers to tasks
Mental demand	NASA-TLX using a seven point likert scale
Physical demand	NASA-TLX (use seven point likert scale)
UX/usability	UEQ S
Satisfaction	3-question After Scenario Questionnaire (ASQ) for questions 1 and 2
Subjective feedback	Analysing comments left by the participants in a free input text

TABLE 3: DEPENDENT VARIABLES AND MEASURING TECHNIQUES

The variable of efficiency was chosen to compare the speeds of different users and see if their speed changes amongst different output modes. The mental demand uses a NASA-TLX standard scale so it can be compared to studies using similar scales. So do physical demand, usability and satisfaction conform to standardized tests. UEQ-S was chosen over SUS since the system was focused around a more task-based approach and not about usability over time or recommendation. Hence SUS was more redundant than UEQ-S.

During the experiment the output modes and input modes are the independent variables. Since one user only interacts with 1 input and 1 output mode a classification system was set up to ensure that the survey results were usable for analysis. See the table below for how the different participants were coded.

	Output 1 (color)	Output 2 (animation)	Output 3 (color + animation)
Input 1 (discrete knob)	P1.1	P2.1	P3.1
	P1.2	P2.2	P3.2
	P1.3	P2.3	P3.3
	P1.4	P2.4	P3.4
Input 2 (continuous knob)	P4.1	P5.1	P6.1
	P4.2	P5.2	P6.2
	P4.3	P5.3	P6.3
	P4.4	P5.4	P6.4
Input 3 (slider)	P7.1	P8.1	P9.1
	P7.2	P8.2	P9.2
	P7.3	P8.3	P9.3
	P7.4	P8.4	P9.4

TABLE 4: CODES FOR PARTICIPANT DISTRIBUTION

7.1.4. PARTICIPANTS

Participants were recruited using word of mouth and social media around the University of Twente. There are no requirements regarding age or experience. Participants do need to have sufficient English skills and sight to complete the tasks. Participants are however required to state experience with geographic data, climate change, data physicalizations and if they are colour blind. These are all factors that might influence the dependent variables of the study.

7.1.5. PROCEDURE AND TASKS

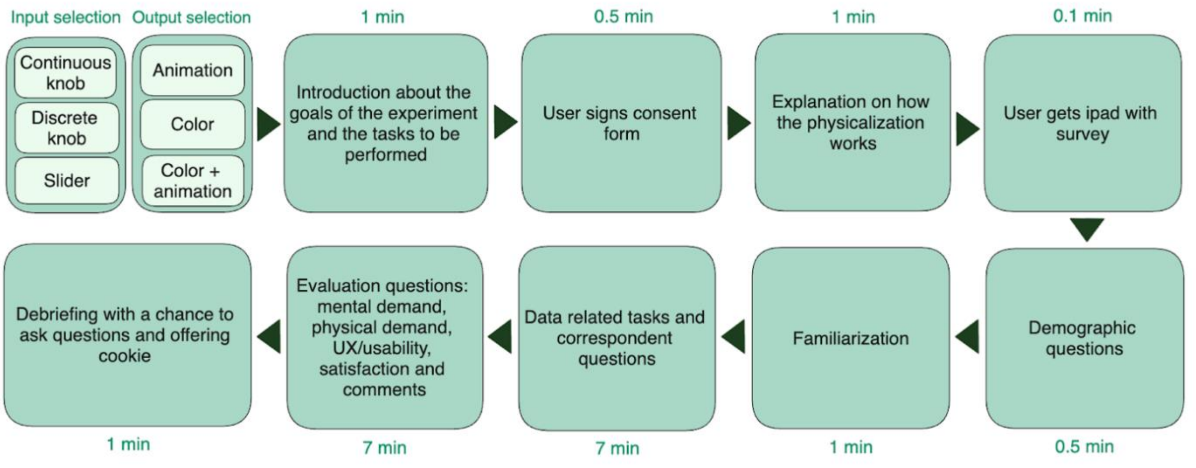


FIGURE 31: FLOW CHART OF USER EVALUATION

During the study one participant and one or two researchers were present. The estimated time for completion of the study is 15 minutes. The participant will be introduced to the concept of the physicalization and the meaning of the data that is being represented. The participant is asked if they have any questions, then they will be presented with a briefing letter and consent form that can be found in **appendix G and H.** The letter will contain the information that also has been explained verbally for clarification. A researcher will than proceed with explaining how the physicalization works and how they need to handle it. Next the participant will be prompted to start the survey on an laptop that will guide them through the experience. The full survey is found in **appendix I.** The laptop starts with a section of demographic questions, then the user will have some time to explore and familiarize themselves with the physicalization. Once they feel comfortable using it they will go to the next part of the survey where they will be prompted to perform tasks related to the data that is being displayed on the globe. The participant will perform a total of 4 tasks and after continuing the survey there is questions about the usability, mental demand, physical demand and some free text input for comments. At the end of the procedure the participant is thanked and offered a cookie.

7.1.6. APPARATUS

Participants sit or stand in front of the physicalization, they are facing their pre determined input mode. A laptop is placed next to the installation which the participant can use to answer questions related to the physicalization. The survey is hosted on LimeSurvey and measures time automatically. The following figure shows the experimental setup.

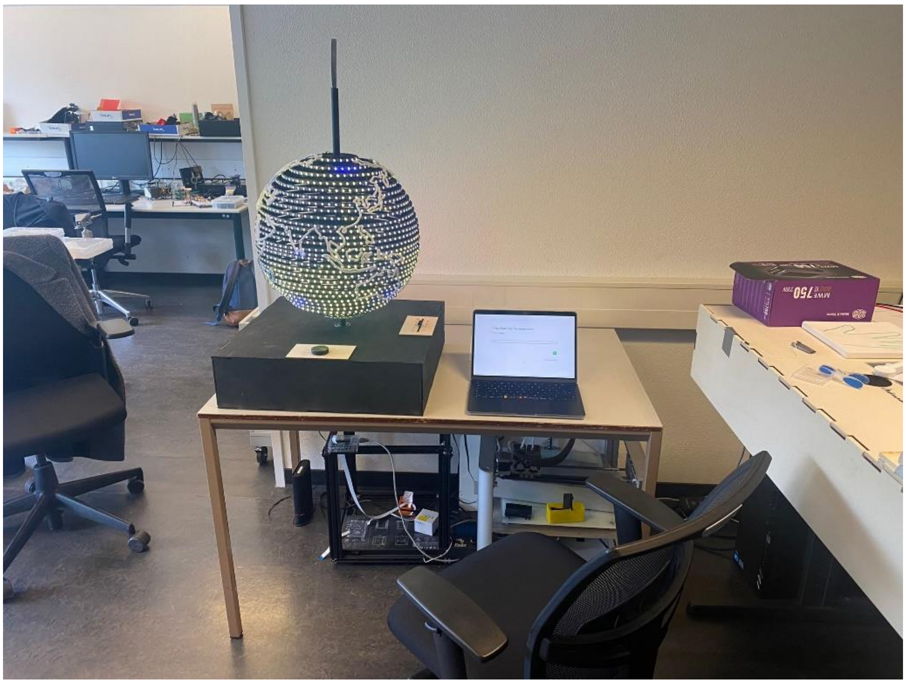


FIGURE 32: EXPIREMENTAL SETUP

7.1.7. QUESTIONS

During the ‘tasks’ the user is prompted to answer the following four questions:

- In the last decade (2010-2020) has Africa become more warm or more cool in general?
- In 1985, what hemisphere has the biggest temperature change, southern or northern? Keep in mind that change includes warmer as well as colder temperatures.
- What area has the highest temperature anomaly in 2000? (Warmer)
- Is the equator warmer in 1990 compared to 2020?

The questions were extracted from a potential identification and comparison questions (see table). The questions have various combinations of ways that the user must obtain the data. So for example a user has to look at a certain year, or compare a certain year to another year. Or comparing between regions. The wording was challenging to choose, since it influences how the users perceive and thus answer the question. In addition to this briefing well is of utmost importance to prevent users from becoming confused because of the questions. To help the user additional information was provided for question 2 and 3. The questions start of with an easy task, to give the user some confidence and set a baseline for the answers. Then the users are faced with some more challenging questions.

Type	Question type	n. years	n. regions	Template	Example	Answer
Y, R vs R	attribute in space + identify	1	2	In YEAR, what region has the biggest temperature change compared to the mean (anomaly), REGION or REGION?	In 1980, what hemisphere has the biggest temperature change compared to the mean (anomaly), southern or northern?	Southern/ northern
Y, max	attribute in space + identify	1	all	In YEAR, which REGION experiences the most increasing temperature/lower temperatures than usual?	In 2015, which continent experiences the most increasing temperature? Highest temperatures?	Continent
R, Y vs Y	Space in time + identify	2	1	Is REGION warmer/colder than usual in YEAR compared to YEAR?	Is the equator warmer in 1990 compared to 2020?	Yes/No
R, range YY	Space in time + identify	range	1	Over the past RANGE, has REGION generally become warmer or cooler?	Over the past decade, has Africa generally become warmer or cooler?	Warmer/ cooler
R, total Y	Space in time + identify	all	all	In what REGION does the mean temperature change the most?	In what area does the mean temperature change the most?	Poles/ equator

TABLE 5: CLASSIFICATION AND EXAMPLES OF CLIMATE CHANGE QUESTIONS

7.2. EVALUATION RESULTS

In the next section the quantitative results will be presented and analysed. Furthermore the qualitative results from the open question and the observations will be discussed.

7.2.1. QUANTITATIVE RESULTS AND SIGNIFICANCE

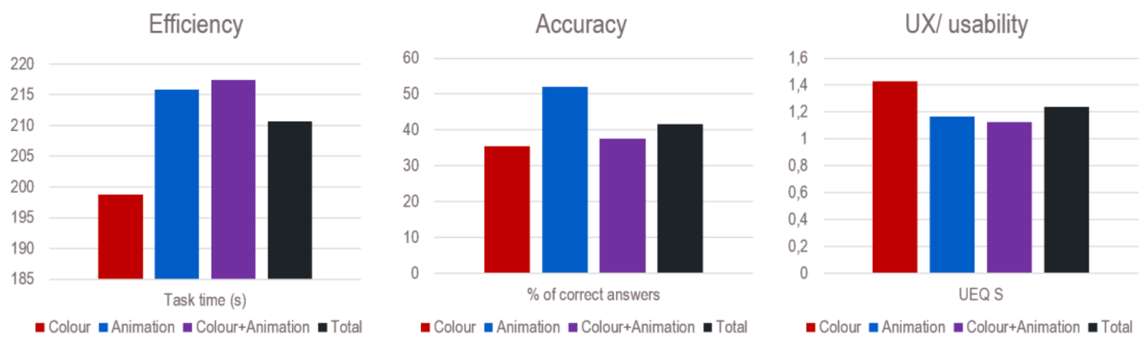


FIGURE 33: GRAPHED MEANS OF EFFICIENCY, ACCURACY AND USABILITY

In the above figure we can see the means for efficiency accuracy and usability. The efficiency is in time, accuracy in percentage and the usability using the UEQ-S, which results in individual scores from -3 to 3. The first observation is that the average usability has a positive mean amongst all groups. This could indicate that overall the physicalization is perceived as usable and user friendly. The accurateness seems to experience a higher mean for the group that just has the animation. We can also see that the task time for animation and colour+animation is higher in comparison to the colour group. However the accuracy does not get higher than 52%, which quite bad for multiple choice questions with at most 3 options.

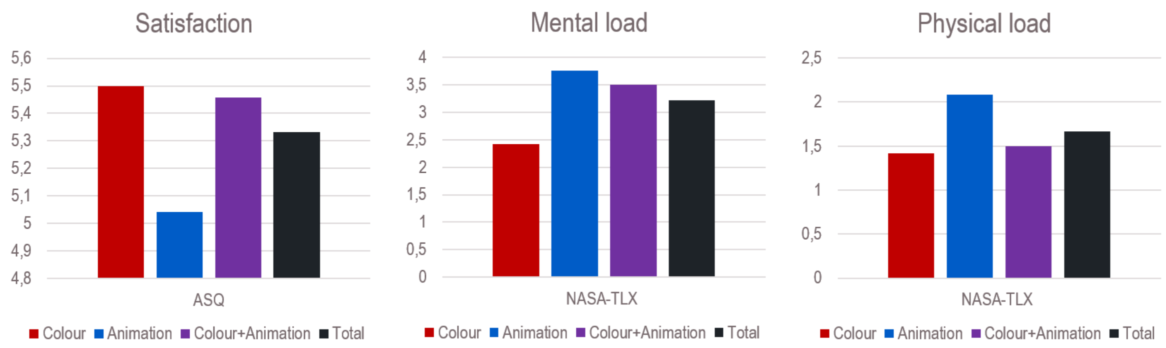


FIGURE 34: GRAPHED MEANS OF SATISFACTION, MENTAL LOAD AND PHYSICAL LOAD

The above figure shows the mean amongst the different groups for satisfaction, mental load and physical load. For satisfaction the average of the 2 questions about satisfaction were calculated, resulting in a single value. One can observe that the satisfaction is slightly higher for the animation, but so is the mental and physical load.

The results above show the mean, but this might not be statistaclly significant. To see if the means are equal an Anova test was done with the following hypotheses:

H0: There is no difference among group means

H1: There is a difference among group means

Since the study is an explorative study with no huge numbers of participants an alpha value of **0.1** was chosen. Anova requires the assumption of normality. QQ plots were generated amongst all participant data.

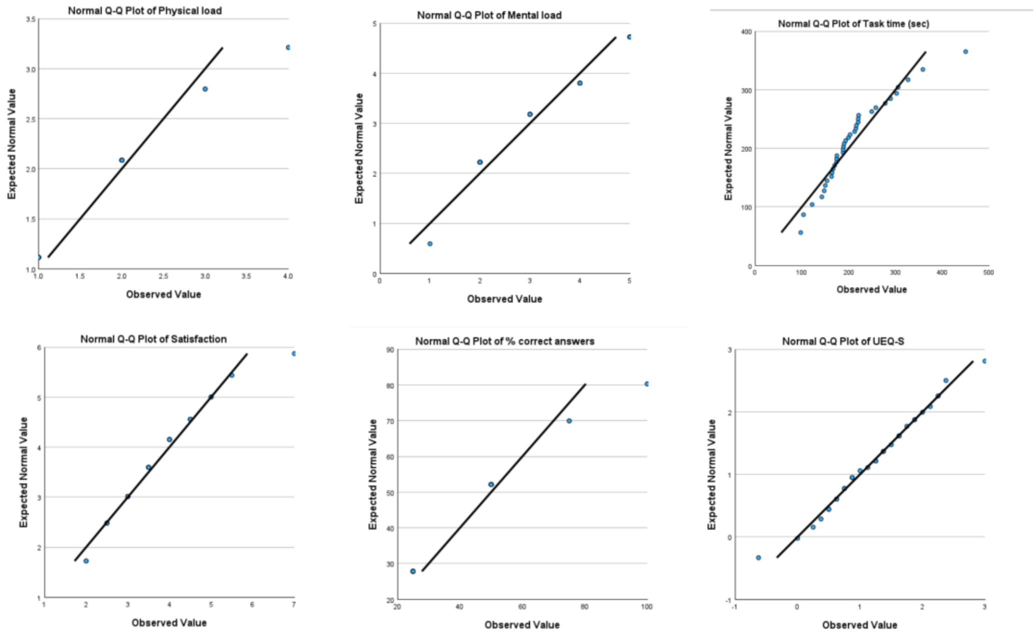


FIGURE 35: QQ PLOTS FOR VARIABLES

Since the values of the survey are close to the normality line on the QQ plots we assume that the data is normally distributed.

ANOVA

		Sig.
UEQ-S	Between Groups	.573
Task time (sec)	Between Groups	.796
Mental load	Between Groups	.016
Physical load	Between Groups	.097
Satisfaction	Between Groups	.505
% correct answers	Between Groups	.097

FIGURE 36: ANOVA RESULTS

In the above table one can see that the null hypotheses get rejected for mental load, physical load and percentage of correct answers. This would suggest that the mean difference for the above stated categories are statistically significant. A LSD Post-Hoc test was done to investigate potential relationships.

Multiple Comparisons

LSD

Dependent Variable	(I) Output	(J) Output	Mean Difference (I-J)	Std. Error	Sig.	90% Confidence Interval Lower Bound	90% Confidence Interval Upper Bound
UEQ-S	Colour	Animation	.260417	.307419	.403	-.25985	.78068
		Colour + animation	.302083	.307419	.333	-.21818	.82235
	Animation	Colour	-.260417	.307419	.403	-.78068	.25985
		Colour + animation	.041667	.307419	.893	-.47860	.56193
	Colour + animation	Colour	-.302083	.307419	.333	-.82235	.21818
		Animation	-.041667	.307419	.893	-.56193	.47860
Task time (sec)	Colour	Animation	-17.07583	30.50598	.579	-68.7029	34.5513
		Colour + animation	-18.60583	30.50598	.546	-70.2329	33.0213
	Animation	Colour	17.07583	30.50598	.579	-34.5513	68.7029
		Colour + animation	-1.53000	30.50598	.960	-53.1571	50.0971
	Colour + animation	Colour	18.60583	30.50598	.546	-33.0213	70.2329
		Animation	1.53000	30.50598	.960	-50.0971	53.1571
Mental load	Colour	Animation	-1.333*	.461	.007	-2.11	-.55
		Colour + animation	-1.083*	.461	.025	-1.86	-.30
	Animation	Colour	1.333*	.461	.007	.55	2.11
		Colour + animation	.250	.461	.592	-.53	1.03
	Colour + animation	Colour	1.083*	.461	.025	.30	1.86
		Animation	-.250	.461	.592	-1.03	.53
Physical load	Colour	Animation	-.667*	.324	.048	-1.22	-.12
		Colour + animation	-.083	.324	.799	-.63	.47
	Animation	Colour	.667*	.324	.048	.12	1.22
		Colour + animation	.583*	.324	.081	.03	1.13
	Colour + animation	Colour	.083	.324	.799	-.47	.63
		Animation	-.583*	.324	.081	-1.13	-.03
Satisfaction	Colour	Animation	-.4583	.4291	.293	-1.185	.268
		Colour + animation	-.0417	.4291	.923	-.768	.685
	Animation	Colour	.4583	.4291	.293	-.268	1.185
		Colour + animation	.4167	.4291	.339	-.310	1.143
	Colour + animation	Colour	.0417	.4291	.923	-.685	.768
		Animation	-.4167	.4291	.339	-1.143	.310
% correct answers	Colour	Animation	-16.667*	8.109	.048	-30.39	-2.94
		Colour + animation	-2.083	8.109	.799	-15.81	11.64
	Animation	Colour	16.667*	8.109	.048	2.94	30.39
		Colour + animation	14.583*	8.109	.081	.86	28.31
	Colour + animation	Colour	2.083	8.109	.799	-11.64	15.81
		Animation	-14.583*	8.109	.081	-28.31	-.86

*. The mean difference is significant at the 0.1 level.

FIGURE 37: POST HOC LSD TEST

The above LSD shows the relationship between the outputs and the different categories. Looking at the mental load all combinations except **animation** vs **colour + animation** are significant. Suggesting that the mental and physical load is higher for users that have animation or colour and animation. In addition to this the physical load experiences the same effect as the mental load, suggesting that the physical load is perceived as higher when the physicalization is in blinking mode. In the post hoc test the percentage of correct answers also seems to be significant, however this was not significant in the ANOVA.

7.2.2. Qualitative results

During the survey the open question was analysed to see if there are any comments that participants share. Furthermore, the observations were noted down. Analysing what has been said or suggested can be put into the following categories: Blinking, appearance, stability and enjoyment. The next section will explore each of these topics in detail.

Usability

Participants generally found the physical aspect of the globe to be intuitive and engaging, enjoying the ability to physically turn the globe rather than having to walk around it. However, some users found turning the globe by hand to be unstable, mentioning that the globe felt fragile and that the bearings seemed likely to fall. There were also issues with the LED indicators, as users found it confusing to know when a new year was selected due to the blinking lights. The blinking feature, while, made it hard for participants to quickly compare data between different years. 10 out of 24 participants that had either blinking or blinking and colour mentioned that the blinking was either confusing or hard to interpret.

Some participants found the colour aspect helpful and focused more on the colour of the lights than on the blinking animation. The blinking made it difficult to quickly compare data between different years, and some participants found it redundant since the colour changes conveyed similar information.

Perceiving

One fourth of the participants commented that it was hard to distinguish between land mass and sea. Some comments were made verbally, whilst others put it as feedback in the survey. Some participants gave suggestions, one of the suggestions was the use of small name tags for better recognition.

Participants also found the concept of anomalies hard to grasp. Throughout the experiment the briefing improved, because the researchers knew what to say to make the concept clearer. However, some participants needed some extra guidance for perceiving the anomaly data. One participant also mentioned that they do not know the scale of the data, the physicalization also did not consist of a legend which they mentioned they missed.

The users were taking a long time to answer question 3, the questions phrasing was discombobulating some users. After explanation, they got the question. About 10 users amongst different groups expressed that they had difficulty answering the questions and that they are not very confident in their answers.

Enjoyment

Many users found the globe visually striking and innovative, enjoying the smooth transition between years and the overall design. A user also mentioned that in class he used to have to read boring maps but suggested that this would be an insightful and fun way to use this in a classroom setting to learn about geographical concepts.

7.2.3. Analysis and Implications

Overall, the globe is perceived very well. This shows through positive comments during the observations and the survey. All participants told us they were intrigued, found it nice or had fun whilst testing. This would suggest that the globe is an engaging way to show data. This is also reflected in the survey, amongst all participants the UEQ-s gave a positive score. A positive UEQ-s score would mean that the globe is intuitive and supportive to use. The usability score mean was higher for the group that used just colour, but the ANOVA test showed that this fact is not statistically significant. Thus we can conclude that in terms of usability there was no major difference amongst groups.

The mean task time for colour was also very much lower than for the other groups. In the observations and comments, it comes forth that the users think they spend a lot of time concentrating at the blinking when they are presented with such a mode. This is reflected in the task time as well and suggests that just using colour causes users to finish their tasks the fastest. However, the ANOVA test says the finding is not statistically significant.

The mean for the mental and physical load is higher for animation, in comparison to colour or colour and animation. This is also reflected back in the quantitative results, where 12 out of 24 participants that were given the animation or colour and animation mode had a comment about the blinking being obtrusive. It is however interesting that the satisfaction of the animation group is also higher in comparison to the other groups. The mental and physical load are statistically significant according to the ANOVA test.

The accuracy of the animation group is statistically significant and larger than the other groups. This is not something that is reflected in the qualitative results, participants of the animation group do not talk about being confident in their answers. However, it would be logical if they are at least slightly more confident, given that the satisfaction of the animation group also has a higher mean, albeit that it is not statistically significant. The results somehow imply that the blinking mode is not preferred, but it is the most satisfying and accurate. However, the measure of accuracy might not be a great one in this user test, because none of groups performed well. In addition to this question 3 was not done right by any participants, showing that the questions and display might have negatively impacted results.

8. DISCUSSION

This research was focused on encoding LEDs on a spherical display and investigating different ways of encoding this data on the perception and usability. This chapter also explores the limitations and general conclusions about usability from the physicalization.

8.1. ACCURACY OF DISPLAYING DATA

During the project the accuracy of displaying data has been a major limitation for the globe. Gathering location-based data and putting this into a digital 2d map already has challenges. The modifiable areal unit problem presents itself whilst getting geographic data from unevenly distributed locations [31]. Transforming this accurately modelled data onto a map is also a big problem, spheres offer a solution for this. But where it is easy to make data accurate using spherical projections on traditional spherical displays, when building an LED sphere, the positions of the LEDs play a major role in how accurate the data you are presenting is (location wise).

A ring-based approach was chosen; however, this is definitely not the most accurate way to present data. The LEDs all face the same direction, and because of the way that the longitude and latitude bins are created it is possible that certain LEDs represent a larger surface than other LEDs. Going for different approaches increases the manual labour of the sphere significantly, since these approaches were based on soldering individual LEDs instead of using a LED strip-based approach.

In addition to the ring based approach, the 3d printed continents were placed, however not very accurately. The printed parts sometimes cover LEDs which does clutter visibility of certain data points. The regions are not displayed very accurately, which could have an influence on the user study and the questions asked during the tasks. The accurate representation of earth has most likely had an impact on the results of the study.

It is interesting for future research to investigate what the best way of placing LEDs in space is so an accurately modelled sphere is created. This not only provides users with accurate data, it also provides researchers with accurate results and a platform to rigidly measure the influence of output modalities.

8.2. LEDS FOR ENCODING VARIABLES

The LEDs used in the project had the ability to be controlled using RGB. This led to the conclusion that the type of encodings used in LEDs in relation to physicalization are colour, brightness, and animation. Since every LED communicates the temperature anomaly of a certain point on earth it is important that the animation should be possible with one LED, this is done through blinking. One algorithm for blinking was used, but in future research this might be implemented in a different way. This algorithm consists of using time and a sine function, but exploring linear vs sine vs other functions might be an interesting research opportunity. It might be interesting to see if different ways of blinking influence the perceivability or interaction.

Furthermore, colour bins were used to implement the colour scale of LEDs. Although some participants found some colours hard to distinguish, it was still the preferred mode for participants. This also reflected back in the statistics, the mental and physical load was lower, and the satisfaction was higher. In the future more high-quality contrasting LEDs might be of use to implement more contrasting colours. In addition to this the LEDs were directly displayed to the user, not having any type of cover nor diffusion. Adding a plastic layer of some sort to diffuse the LEDs would enable the LEDs to be brighter. In the current implementation the brightness was limited, due to strain on the eyes which is user unfriendly. Diffusing the LEDs might provide more contrast possibilities and also entice users to interact with the sphere more. Since a diffused sphere will look cleaner and more robust than the current design.

8.3. Blinking

The blinking was found difficult or hard to grasp, is what about half of the participants that got presented blinking animations mentioned. The blinking was carefully designed, and a lot of options were tried during the realization stage of the project. However during the realization stage, it was already noticed that it is quite hard to find perceivable blinking patters. It is easier to perceive which areas are going off at the same time than to perceive the blink speed at a certain area. What might help with this is a legend, in particular a legend on which one can see which areas are on or off live. This could enable users to directly compare the area they are looking at with the legend, to improve readability and perceivability.

During the user study the statistics show significant results at accuracy amongst groups, surprisingly enough the animation group scored highest in the accuracy measure. This could be because the user study was imperfect, this will be discussed in the next paragraph. Although the number of participants is quite low, one can still speculate about why this phenomenon happened. One of the potential reasons is that the users in the animation group spent, on average, more time with the installation. The blinking could have forced them to pay closer attention and thus more accurately answer questions.

The current implementation of colour and animation made sure that the animation and colour used the same set of bins, thus the same data was displayed using 2 different modalities. Further research could investigate using separate bins or separate datasets for the different modalities.

8.4. User study

The user study was conducted with 36 participants with 6 independent variables resulting in 9 groups of 4. The user study was conducted in a public space and all participants had about the same exernal effects. However, the study participant number still is a limitation for statistically significant data one can get out of the results.

Furthermore during the user study it was noticed that the concept of anomalies was hard to explain. Throughout the user test the participants asked less and less questions about it, since the researchers got better at explaining the concept of anomalies. Future research could benefit from displaying dummy data, or data that is easier to grasp, so the participants comprehension level of the data does not influence the results.

The user study consisted of 4 tasks that the participants had to perform. Question one was done correctly by all participants, whilst question three was done incorrectly by all participants. This shows that the questions chosen were not the best questions to ask. The questions should not be too easy, but also not be too hard. Future research could use dummy data to ensure that the questions and the answers are correctly displayed and relatively easy to grasp.

The user study did test for usability/user experience, however only in private setting. It could also be an interesting opportunity to display the globe in public and see how people then interact with it and how this influences usability or user experience.

8.5. Colour

Colour is traditionally used a lot in visualizations of geographic data. In this physicalization it was implemented using colour bins to show the value of temperature anomalies on the globe. Choosing the correct colours on these LEDs posed a challenge. The LEDs had a limited brightness because they would otherwise strain the eyes. Making them contrast was hard because of the limited output.

The colour was preferred by the users. One could see this through the comments, where participants mentioned the blinking felt redundant whilst using the animation + colour mode. Since the participants all had some familiarity with geographic maps and data it could mean that they opted for colour because of familiarity. An interesting research opportunity would be to test with a group that has never interacted with maps before, since the familiarization bias would be reduced.

9. CONCLUSION

The goal of this research was to investigate the effects of output modalities on the user experience and perceivability with the research question being:

How do different combination of output modes: colour and blink speed, and each modality separately, compare in data perception and user experience of a data physicalization conveying climate change data?

In order to answer this question Globey was built; a data physicalization showing climate change data using a ring based globe design. This physicalization was user tested with 36 participants by giving them tasks and letting them answer questions related to user experience, mental load, physical load and satisfaction. Statistical analysis shows that there is some type of statistically significant relationship between output modes and mental load, physical load and accuracy. However, because of the small size of the user test and limitations of the globe these results are not stable. Overall, the physicalization was perceived as very positive and attracted the participants of the study, showing that using an interactive globe is an effective way to attract users and spark interest.

There is preference for the colour modality, however this might be caused because of familiarization bias with geographic data. Users do not prefer blinking when it comes to showing data, however the group in this category performed best. No definitive statements can be made about which output modality worked best.

Future research could redo the study with a larger group and better selected questions, potentially using dummy data, to repeat and see if the results in the current study are accurate. Furthermore, future work could entail investigating what the most accurate and achievable way of placing LEDs on a sphere is to ensure an accurate representation of data. Also more modalities could be incorporated to see how this influences the perception of climate change data.

10. REFERENCES

[1] European Environment Agenct, 'Climate change impacts, risks and adaptation', Climate change impacts, risks and adaptation. Accessed: Jun. 12, 2024. [Online]. Available: https://www.eea.europa.eu/en/topics/in-depth/climate-change-impacts-risks-and-adaptation

[2] Y. Yang, B. Jenny, T. Dwyer, K. Marriott, H. Chen, and M. Cordeil, 'Maps and Globes in Virtual Reality', *Computer Graphics Forum*, vol. 37, no. 3, pp. 427–438, Jun. 2018, doi: 10.1111/cgf.13431.

[3] Intergovernmental Panel On Climate Change (Ipcc), *Climate Change 2021 – The Physical Science Basis: Working Group I Contribution to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*, 1st ed. Cambridge University Press, 2023. doi: 10.1017/9781009157896.

[4] Kadek Ananta Satriadi *et al.*, 'Tangible Globes for Data Visualisation in Augmented Reality', *International Conference on Human Factors in Computing Systems*, Apr. 2022, doi: 10.1145/3491102.3517715.

[5] E. Van Den Hoven, J. Frens, D. Aliakseyeu, J.-B. Martens, K. Overbeeke, and P. Peters, 'Design research & tangible interaction', in *Proceedings of the 1st international conference on Tangible and embedded interaction*, Baton Rouge Louisiana: ACM, Feb. 2007, pp. 109–115. doi: 10.1145/1226969.1226993.

[6] Y. Jansen *et al.*, 'Opportunities and Challenges for Data Physicalization', in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, Seoul Republic of Korea: ACM, Apr. 2015, pp. 3227–3236. doi: 10.1145/2702123.2702180.

[7] S. S. Bae, C. Zheng, M. E. West, E. Y.-L. Do, S. Huron, and D. A. Szafir, 'Making Data Tangible: A Cross-disciplinary Design Space for Data Physicalization', in *CHI Conference on Human Factors in Computing Systems*, New Orleans LA USA: ACM, Apr. 2022, pp. 1–18. doi: 10.1145/3491102.3501939.

[8] Nikita Soni *et al.*, 'Towards Understanding Interactions with Multi-Touch Spherical Displays', *CHI Extended Abstracts*, May 2019, doi: 10.1145/3290607.3313063.

[9] Julie R. Williamson *et al.*, 'GlobalFestival: evaluating real world interaction on a spherical display', *Ubiquitous Computing*, pp. 1251–1261, Sep. 2015, doi: 10.1145/2750858.2807518.

[10] Hrvoje Benko, H. Benko, Andrew Wilson, A. D. Wilson, Ravin Balakrishnan, and R. Balakrishnan, 'Sphere: multi-touch interactions on a spherical display', *ACM Symposium on User Interface Software and Technology*, pp. 77–86, Oct. 2008, doi: 10.1145/1449715.1449729.

[11] Julie R. Williamson, J. R. Williamson, Daniel Sundén, D. Sundén, Keith Hamilton, and K. Hamilton, 'The lay of the land: techniques for displaying discrete and continuous content on a spherical display', *International Symposium on Pervasive Displays*, pp. 38–44, Jun. 2016, doi: 10.1145/2914920.2915005.

[12] David Englmeier *et al.*, 'TangibleSphere – Interaction Techniques for Physical and Virtual Spherical Displays', *Nordic Conference on Human-Computer Interaction*, 2020, doi: 10.1145/3419249.3420101.

[13] Harry Brignull, H. Brignull, Yvonne Rogers, and Y. Rogers, 'Enticing People to Interact with Large Public Displays in Public Spaces.', Jan. 2003.

[14] Julie R. Williamson, J. R. Williamson, Daniel Sundén, and D. Sundén, 'Enter the Circle: Blending Spherical Displays and Playful Embedded Interaction in Public Spaces', *International Symposium on Pervasive Displays*, pp. 195–200, Jun. 2015, doi: 10.1145/2757710.2757731.

[15] Eva Hornecker, E. Hornecker, Matthias Stifter, and M. Stifter, 'Learning from interactive museum installations about interaction design for public settings', *Australasian Computer-Human Interaction Conference*, pp. 135–142, Nov. 2006, doi: 10.1145/1228175.1228201.

[16] Y. Jansen, P. Dragicevic, and J.-D. Fekete, 'Evaluating the efficiency of physical visualizations', in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Paris France: ACM, Apr. 2013, pp. 2593–2602. doi: 10.1145/2470654.2481359.

[17] M. Daniel, G. Rivière, and N. Couture, 'CairnFORM: a Shape-Changing Ring Chart Notifying Renewable Energy Availability in Peripheral Locations', in *Proceedings of the Thirteenth International Conference on Tangible, Embedded, and Embodied Interaction*, Tempe Arizona USA: ACM, Mar. 2019, pp. 275–286. doi: 10.1145/3294109.3295634.

[18] D. Peeters, C. Ranasinghe, A. Degbelo, and F. Ahmed, 'EmoClock: Communicating Real-Time Emotional States Through Data Physicalizations', in *Human-Computer Interaction – INTERACT 2023*, vol. 14142, J. Abdelnour Nocera, M. Kristín Lárusdóttir, H. Petrie, A. Piccinno, and M. Winckler, Eds., in Lecture Notes in Computer Science, vol. 14142. , Cham: Springer Nature Switzerland, 2023, pp. 416–425. doi: 10.1007/978-3-031-42280-5_26.

[19] C. Ranasinghe and A. Degbelo, ‘Encoding Variables, Evaluation Criteria, and Evaluation Methods for Data Physicalisations: A Review’, *MTI*, vol. 7, no. 7, p. 73, Jul. 2023, doi: 10.3390/mti7070073.
[20] W. Ju and L. Leifer, ‘The Design of Implicit Interactions: Making Interactive Systems Less Obnoxious’, *Design Issues*, vol. 24, no. 3, pp. 72–84, Jul. 2008, doi: 10.1162/desi.2008.24.3.72.
[21] S. Dullaert, C. Ranasinghe, A. Degbelo, and N. Bouali, ‘Data Physicalization with Haptic Variables: Exploring Resistance and Friction’, in *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, Honolulu HI USA: ACM, May 2024, pp. 1–8. doi: 10.1145/3613905.3651011.
[22] R. Van Loenhout, C. Ranasinghe, A. Degbelo, and N. Bouali, ‘Physicalizing Sustainable Development Goals Data: An Example with SDG 7 (Affordable and Clean Energy)’, in *CHI Conference on Human Factors in Computing Systems Extended Abstracts*, New Orleans LA USA: ACM, Apr. 2022, pp. 1–7. doi: 10.1145/3491101.3519638.
[23] S. De Kreij, C. Ranasinghe, and A. Degbelo, ‘Data Physicalization and Tangible Manipulation for Engaging Children with Data: An Example with Air Quality Data’, in *Proceedings of the 23rd Annual ACM Interaction Design and Children Conference*, Delft Netherlands: ACM, Jun. 2024, pp. 507–516. doi: 10.1145/3628516.3655788.
[24] S. Houben *et al.*, ‘Physikit: Data Engagement Through Physical Ambient Visualizations in the Home’, in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, San Jose California USA: ACM, May 2016, pp. 1608–1619. doi: 10.1145/2858036.2858059.
[25] X. Du *et al.*, ‘That’s Rough! Encoding Data into Roughness for Physicalization’, in *Proceedings of the CHI Conference on Human Factors in Computing Systems*, Honolulu HI USA: ACM, May 2024, pp. 1–16. doi: 10.1145/3613904.3641900.
[26] A. Mader and W. Eggink, ‘A design process for creative technology’, *INTERNATIONAL CONFERENCE ON ENGINEERING AND PRODUCT DESIGN EDUCATION*, Sep. 2014.
[27] *LED sphere with 194 LEDs and how to build it*, (Dec. 16, 2019). [Online Video]. Available: https://www.youtube.com/watch?v=Q5d8gTppuYo
[28] *An LED Sphere YOU Can Build*, (Jan. 18, 2024). [Online Video]. Available: https://www.youtube.com/watch?v=l0WGbmc9_4Q&t=1s
[29] *Spherical LED Matrix - Sphere Shaped LED Screen From LED Pixel Strip*, (Jun. 12, 2015). [Online Video]. Available: https://www.youtube.com/watch?v=oP9oEa2yVMQ
[30] *Giant Animated RGB LED Pixel Mega Sphere Ball Kit*, (Aug. 14, 2018). [Online Video]. Available: https://www.youtube.com/watch?v=QivlngKW6wg
[31] D. E. Jelinski and J. Wu, ‘The modifiable areal unit problem and implications for landscape ecology’, *Landscape Ecol*, vol. 11, no. 3, pp. 129–140, Jun. 1996, doi: 10.1007/BF02447512.

11. APPENDIX

A. Server code

App

```
from flask import Flask, render_template
from flask_sock import Sock
import threading
from functions import *
import numpy as np
import time
import re

app = Flask(__name__)

sock = Sock(app)

global last_access_time, min_temp, max_temp, bins_and_frequencies, color_bins,
colors, max_brightness, slider_active, knob_1_active, knob_2_active,
sphere_animation, sphere_color, animation_color
min_temp = -3
max_temp = 3
bins_and_frequencies: List[Tuple[int, int, int]] = [(2, 3, 20),(0.5, 2,
60),(0, 0.5, 100)]
color_bins = [(-3,-2),(-2, -0.5),(-0.5, 0.5),(0.5, 2),(2,3)]
colors = [
    (0, 0, 255),
    (50, 50, 150),  # Cyan
    (50, 50, 40),  # Orange
    (75, 75, 0),
    (255, 0, 0),  # Orange
    ]
max_brightness = 25
slider_active = False
knob_1_active = False
knob_2_active = False
sphere_animation = True
sphere_color = True
animation_color = (50,50,50)

RATE_LIMIT = 0.050
last_access_time = 0
# Store active WebSocket connections
connections = {}
sphere_data = get_detail()

@app.route(“/”)
def hello_world():
    return render_template(‘index.html’)
```

```
@sock.route('/north')
def send(wsSend):
    connections['north'] = wsSend
    try:
        while True:
            data = wsSend.receive()
            print(data)
    except Exception as e:
        print(f"Connection error: {e}")

@sock.route('/south')
def sendSouth(wsSend):
    connections['south'] = wsSend
    try:
        while True:
            data = wsSend.receive()
            print(data)
    except Exception as e:
        print(f"Connection error: {e}")

@sock.route('/rgb')
def sendRGB(wsSend):
    connections['rgb'] = wsSend
    try:
        while True:
            data = wsSend.receive()
            if data:
                print(f"Received data: {data}")
                try:
                    rgb_values = re.findall(r'\d+', data)
                    r, g, b = map(int, rgb_values)

                    detail = sphere_data['2006']
                    limited = limit_data(detail, -3, 3)
                    sphere_colors = make_one_color(limited, 25, r, g, b)
                    concat_array = get_concatenated_array(sphere_colors)
                    north, south = return_hemispheres(concat_array)
                    print(north[:10])
                    print(south[:10])
                    north_bytes = bytes(north)
                    south_bytes = bytes(south)
                    threading.Thread(target=broadcast, args=(south_bytes,
'south')).start()
                    threading.Thread(target=broadcast, args=(north_bytes,
'north')).start()
                except ValueError as e:
                    print(f"Error parsing RGB values: {e}")
```

```
            else:
                break

            # print(data['r'], data['g'], data['b'])
            # 59eceived = wsSend.receive()
    except Exception as e:
        print(f"Connection error: {e}")

@sock.route('/settings')
def setSettings(wsSend):
    global min_temp, max_temp, bins_and_frequencies, color_bins, colors,
max_brightness, slider_active, knob_1_active, knob_2_active, sphere_animation,
sphere_color, animation_color
    connections['settings'] = wsSend
    try:
        while True:
            received = wsSend.receive()
            if received:
                try:
                    settings = json.loads(received)

                    min_temp = settings['min']
                    max_temp = settings['max']
                    bins_and_frequencies = settings['bins_and_frequencies']
                    colors = settings['colors']
                    max_brightness = settings['max_brightness']
                    color_bins = settings['color_bins']
                    slider_active = settings['slider_active']
                    knob_1_active = settings['knob_1_active']
                    knob_2_active = settings['knob_2_active']
                    sphere_animation = settings['sphere_animation']
                    sphere_color = settings['sphere_color']
                    animation_color = tuple(settings['animation_color'][0])
                    # animation_color = (animation_color, animation_color[1],
animation_color[2])

                    colors = [tuple(color) for color in settings['colors']]
                    color_bins = [tuple(bin) for bin in
settings['color_bins']]
                    bins_and_frequencies = [tuple(bins) for bins in
settings['bins_and_frequencies']]

                except ValueError as e:
                    print(f"Error parsing settings: {e}")
            else:
                break
    except Exception as e:
        print(f"Connection error: {e}")
```

```
@sock.route(‘/input1’)
def sendInput(wsSend):
    connections[‘input1’] = wsSend
    try:
        while True:
            60eceived = wsSend.receive()
            if 60eceived:
                global last_access_time
                current_time = time.time()  # Convert to milliseconds
                try:
                    if current_time – last_access_time < RATE_LIMIT:
                        # print(‘skip’)
                        continue  # Skip the rest of the loop
                    # print(‘not skippng’)

                    last_access_time = current_time
                    sendSphereData(60eceived)


                    # threading.Thread(target=broadcast, args=(south_bytes,
‘south’)).start()
                    # threading.Thread(target=broadcast, args=(north_bytes,
‘north’)).start()
                except ValueError as e:
                    print(f”Error parsing RGB values: {e}”)
            else:
                break
    except Exception as e:
        print(f”Connection error: {e}”)

@sock.route(‘/controller’)
def sendYear(wsSend):
    connections[‘controller’] = wsSend
    try:
        while True:
            60eceived = wsSend.receive()
            if 60eceived:
                global last_access_time, knob_1_active, knob_2_active,
slider_active
                current_time = time.time()  # Convert to milliseconds
                try:


                    last_access_time = current_time
                    parsed = json.loads(60eceived)
                    slider = parsed[‘slider’]
                    knob1 = parsed[‘knob1’]
```

```
                    knob2 = parsed[‘knob2’]
                    if knob_1_active:
                        sendSphereData(knob1)
                    elif knob_2_active:
                        sendSphereData(knob2)
                    elif slider_active:
                        sendSphereData(slider)

                except ValueError as e:
                    print(f”Error parsing RGB values: {e}”)

            else:
                break
    except Exception as e:
        print(f”Connection error: {e}”)

def broadcast(message, client):
    try:
        connections[client].send(message)
    except Exception as e:
        print(f”Error sending message: {e}”)

def sendSphereData(61eceived):
    detail = sphere_data[str(61eceived)]
    normalize = limit_data(detail, min_temp, max_temp)
    sphere_colors = []
    if sphere_animation and sphere_color:
        sphere_colors = map_data_to_colors_and_frequency(normalize,colors,
color_bins, max_brightness, min_temp, max_temp, bins_and_frequencies)
    elif sphere_animation and not sphere_color:
        sphere_colors = map_data_to_frequency(normalize,
max_brightness,bins_and_frequencies, animation_color, min_temp, max_temp)
    elif not sphere_animation and sphere_color:
        sphere_colors = map_data_to_colors(normalize, color_bins, colors,
max_brightness)
    elif not sphere_animation and not sphere_color:
        sphere_colors = get_geography(25)

    north, south = get_hemispheres(sphere_colors)
    # print(north)
    # print(“North”,north[:100], “South”,south[:100])
    # print(“North”,north[3864:], “South”,south[3864:])
    # print(“leng”, len(north), len(south))

    north_bytes = bytes(north)
    south_bytes = bytes(south)
    broadcast(south_bytes, ‘south’)
    broadcast(north_bytes, ‘north’)
```

Functions

```
import json
import numpy as np
import os
import matplotlib.pyplot as plt
from matplotlib.widgets import Slider
from matplotlib.colors import LinearSegmentedColormap
import itertools
from typing import List, Dict, Tuple
import matplotlib.colors as mcolors

def get_detail():
    all_detail = {}
    # Get the list of all JSON files in the sphere_detail_output folder
    json_files = [f for f in os.listdir('sphere_detail_output') if
f.endswith('.json')]

    # Append the contents of each JSON file to the all_detail dictionary
    for file in json_files:
        year = file.split('_')[2].split('.')[0]
        with open(os.path.join('sphere_detail_output', file), 'r') as f:
            detail = json.load(f)
            all_detail[year] = detail
    return all_detail

def get_geography(max_brightness):
    detail = {}
    with open(os.path.join('mcont', 'continents.json'), 'r') as f:
        detail = json.load(f)
    rgb_data = {}
    for key in detail:
        rgb_data[key] = []
        for value in detail[key]:
            # Convert normalized value to RGB color
            rgba = tuple(value)
            rgb = [int(max_brightness * c) for c in rgba]
            rgb.append(int(0))
            rgb_data[key].extend(rgb)
    return rgb_data


# Update function for slider
def limit_data(detail_from_year, min, max):
    for key in detail_from_year:
        for i in range(len(detail_from_year[key])):
            if detail_from_year[key][i] > max:
                detail_from_year[key][i] = max
            elif detail_from_year[key][i] < min:
                detail_from_year[key][i] = min
```

```
            # print(detail_from_year[key][i], "Normalized:",
(detail_from_year[key][i] - min) / (max - min))
            # detail_from_year[key][i] = (detail_from_year[key][i] - min) /
(max - min)
        # detail_from_year[key] = np.clip(detail_from_year, 0, 1)
    return detail_from_year

def get_frequency(value, min, max, bins_and_frequency: List[Tuple[int, int,
int]]):
    distance_from_middle = abs(value)
    if distance_from_middle > max:
        distance_from_middle = max
    if distance_from_middle < 0:
        distance_from_middle = 0
    bins = [(x[0], x[1]) for x in bins_and_frequency]
    frequency_per_bin = [x[2] for x in bins_and_frequency] #Divide by 100,
since we are multiplying with 100 in the end
    for index, (bin_start, bin_end) in enumerate(bins):
        if bin_start <= distance_from_middle <= bin_end:
            return frequency_per_bin[index]
    return frequency_per_bin[len(frequency_per_bin)-1]

def get_color(value, bins, colors):
    for index, (bin_start, bin_end) in enumerate(bins):
        if bin_start <= value < bin_end:
            return colors[index]
    return colors[-1]

def map_data_to_colors_and_frequency(clipped_data, colors, color_bins,
max_brightness, min, max, bins_and_frequency: List[Tuple[int, int, int]]):
    rgb_data = {}
    for key in clipped_data:
        rgb_data[key] = []
        for value in clipped_data[key]:
            # Convert normalized value to RGB color
            rgba = get_color(value, color_bins, colors)
            rgb = [int(max_brightness * c/255) for c in rgba]  # Take the RGB
values and scale to 0-255
            rgb.append(int(get_frequency(value, min, max,
bins_and_frequency)))  # Add the alpha value to the list (for frequency of the
color)
            rgb_data[key].extend(rgb)
    return rgb_data

def map_data_to_colors(clipped_data, color_bins, color, max_brightness):
    rgb_data = {}
    for key in clipped_data:
        rgb_data[key] = []
```

```
        for value in clipped_data[key]:
            # Convert normalized value to RGB color
            rgba = get_color(value, color_bins, color)
            rgb = [int(max_brightness * c/255) for c in rgba]
            rgb.append(int(0))
            rgb_data[key].extend(rgb)
    return rgb_data

def map_data_to_frequency(clipped_data, max_brightness, bins_and_frequency:
List[Tuple[int, int, int]], animation_color: Tuple[int, int, int], min_temp,
max_temp):
    rgb_data = {}
    for key in clipped_data:
        rgb_data[key] = []
        for value in clipped_data[key]:
            # Convert normalized value to RGB color
            rgba = animation_color
            if value < 0:
                rgba = (0, 0, 255)
            elif value >= 0:
                rgba = (255, 0, 0)
            rgb = [int(max_brightness * c/255) for c in rgba]
            rgb.append(int(get_frequency(value,min_temp, max_temp,
bins_and_frequency)))
            rgb_data[key].extend(rgb)
    return rgb_data

def make_one_color(clipped_data, max_brightness, r, g, b):
    rgb_data = {}

    for key in clipped_data:
        rgb_data[key] = []
        for value in clipped_data[key]:
            # Convert normalized value to RGB color
            rgba = (r/255, g/255, b/255)
            rgb = [int(max_brightness * c) for c in rgba]  # Take the RGB
values and scale to 0-255
            distance_from_middle = abs(value - 0.5)
            # rgb.append(int(distance_from_middle * 255))  # Add the alpha
value to the list (for frequencty of the color)
            rgb.append(int(0))
            rgb_data[key].extend(rgb)

    return rgb_data



def create_red_to_blue_cmap(colors):
```

```
    colors1 = [
    (0, (5/255, 0, 213/255)),
    (0.08, (0, 40/255, 214/255)),   # Dark Blue
    (0.16, (0, 80/255, 214/255)),  # Blue
    (0.24, (0, 130/255, 214/255)),  # Cyan
    (0.32, (62/255, 142/255, 190/255)),  # White
    (0.42, (89/255, 143/255, 177/255)),  # Yellow
    (0.5, (107/255, 107/255, 107/255)),  # Orange
    (0.58, (255/255, 85/255, 13/255)),      # Red
    (0.68, (255/255, 80/255, 0)),  # Blue
    (0.76, (255/255, 50/255, 0)),  # Cyan
    (0.84, (255/255, 30/255, 0/255)),  # White
    (0.92, (255/255, 20/255, 0)),  # Yellow
    (1, (255/255, 0, 0)),  # Orange
    ]

    # colors = [
    # (0, (5/255, 0, 213/255)),
    # (0.24, (0, 130/255, 214/255)),  # Cyan
    # (0.5, (107/255, 107/255, 107/255)),  # Orange
    # (0.75, (255/255, 50/255, 0)),
    # (1, (255/255, 0, 0)),  # Orange
    # ]

# Create the colormap
    cmap = mcolors.LinearSegmentedColormap.from_list('discrete_color', colors,
N=len(colors))


    return cmap

def get_concatenated_array(colordata):
    concatenated_array = sum(colordata.values(), [])
    return concatenated_array

def get_hemispheres(colordata):
    north_hemisphere = []
    south_hemisphere = []

    # Ensure that keys are sorted as integers to maintain order
    sorted_keys = sorted(colordata.keys(), key=lambda x: int(x))

    for key in sorted_keys:
        if 1 <= int(key) <= 15:
            north_hemisphere.extend(colordata[key])

    for key in sorted_keys[::-1]:  # Iterate over sorted keys in reverse order
        if 16 <= int(key) <= 30:
            south_hemisphere.extend(colordata[key])
```

```python
    return north_hemisphere, south_hemisphere

def return_hemispheres(concat_list):
    total_rows = len(concat_list)
    mid_index = total_rows // 2  # Integer division for midpoint

    # Handle odd number of rows by adding one extra to the northern hemisphere
    if total_rows % 2 != 0:
        mid_index += 1

    south = concat_list[:mid_index]
    north = concat_list[mid_index:]

    return north, south


# detail = get_detail()['2005']
# normalize = normalize_data(detail, -3, 3)
# colors = map_data_to_colors(normalize, create_red_to_blue_cmap(), 50)
# concat_array = get_concatenated_array(colors)
# north, south = return_hemispheres(concat_array)
```

Html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Globey</title>
    <link rel="stylesheet" href="{{url_for('static', filename='global.css')}}"
type="text/css">
    <script src="{{url_for('static', filename='script.js')}}"
type="text/javascript"></script>
  </head>
  <body>
    <h1>Globey</h1>
    <span id="sliderValue">1979</span>
    <input style="display: inline-block; width: 70%;" type="range"
id="messageInput" min="1979" max="2023" step="4" value="1979">
    <input type="color" id="colorPicker" name="head" value="#e66465" />
    <div>
      <form id="dataForm">
        <label for="min">Min:</label>
        <input type="number" id="min" name="min" value="-3"><br><br>

        <label for="max">Max:</label>
```

```
        <input type=”number” id=”max” name=”max” value=”3”><br><br>

        <label for=”bins_and_frequencies”>Bins and Frequencies (format:
[0,1,10],[1,2,20],[2,3,30]):</label>
        <input type=”text” id=”bins_and_frequencies”
name=”bins_and_frequencies” value=”[0,0.5,100],[0.5,2,60],[2,3,20]”><br><br>

        <label for=”colors”>Colors (format:
[255,255,255],[255,0,0],[0,0,255]):</label>
        <input type=”text” id=”colors” name=”colors” value=”[0, 0, 255],[50,
50, 150],[50, 50, 40],[75, 75, 0],[255, 0, 0]”>
        <br><br>

        <label for=”bins”>Bins (format: [-3,-2],[-2,-1]):</label>
        <input type=”text” id=”bins” name=”bins” value=”[-3,-2],[-2, -0.5],[-
0.5, 0.5],[0.5, 2],[2,3]”><br><br>

        <label for=”max_brightness”>Max Brightness:</label>
        <input type=”number” id=”max_brightness” name=”max_brightness”
value=”25”><br><br>

        <label for=”slider_active”>Slider Active:</label>
        <input type=”checkbox” id=”slider_active”
name=”slider_active”><br><br>

        <label for=”knob1_active”>Knob 1 Active:</label>
        <input type=”checkbox” id=”knob1_active” name=”knob1_active”
checked><br><br>

        <label for=”knob2_active”>Knob 2 Active:</label>
        <input type=”checkbox” id=”knob2_active” name=”knob2_active”><br><br>

        <label for=”animation”>Animation:</label>
        <input type=”checkbox” id=”animation” name=”animation”
checked><br><br>

        <label for=”animation_color”>Animation color (format:
[255,255,255]):</label>
        <input type=”text” id=”animation_color” name=”animation_color”
value=”[50,50,50]”><br><br>

        <label for=”color”>Color:</label>
        <input type=”checkbox” id=”color” name=”color” checked><br><br>

        <button type=”button” id=”sendForm”>Send</button>
    </form>
    </div>
</body>
```

```
</html>
```

Javascript

```
const sendSocket = new WebSocket("ws://" + window.location.host + "/input1");
const settingsSocket = new WebSocket("ws://" + window.location.host +
"/settings");
const rgbSocket = new WebSocket("ws://" + window.location.host + "/rgb");
document.addEventListener("DOMContentLoaded", function () {
    // const sendRGB = new WebSocket("ws://" + window.location.host + "/rgb");

    sendSocket.onmessage = function(event) {
        const messageDisplay = document.getElementById("messages");
        const messageElement = document.createElement("div");
        messageElement.textContent = "Received: " + event.data;
        messageDisplay.appendChild(messageElement);
    };

    document.getElementById("colorPicker").oninput = function() {
        const hexValue = document.getElementById("colorPicker").value;
        const r = parseInt(hexValue.substring(1, 3), 16);
        const g = parseInt(hexValue.substring(3, 5), 16);
        const b = parseInt(hexValue.substring(5, 7), 16);
        const colorData = `r${r}g${g}b${b}`;
        console.log(colorData);
        rgbSocket.send(colorData);
    };


    document.getElementById("messageInput").oninput = function() {
        const message = document.getElementById("messageInput").value;
        document.getElementById("sliderValue").textContent = message;
        console.log("Slider changed to: " + message);
        sendSocket.send(message);
    };

    function combineColorsAndBins(colors, bins) {
        return colors.map((color, index) => {
            return { color: color, bin: bins[index] };
        });
    }

    document.getElementById("sendForm").onclick = function() {
        const data = {
            min: parseFloat(document.getElementById('min').value),
            max: parseFloat(document.getElementById('max').value),
            bins_and_frequencies:
parseArrayInput(document.getElementById('bins_and_frequencies').value),
            colors: parseArrayInput(document.getElementById('colors').value),
            color_bins:
parseArrayInput(document.getElementById('bins').value),
```

```
            max_brightness: 
parseFloat(document.getElementById('max_brightness').value),
            slider_active: document.getElementById('slider_active').checked,
            knob_1_active: document.getElementById('knob1_active').checked,
            knob_2_active: document.getElementById('knob2_active').checked,
            animation_color: 
parseArrayInput(document.getElementById('animation_color').value),
            sphere_animation: document.getElementById('animation').checked,
            sphere_color: document.getElementById('color').checked
        };

        settingsSocket.send(JSON.stringify(data));
    };

    function parseArrayInput(input) {
        return input.split('],[').map(item => item.replace(/\[|\]/g, 
'').split(',').map(Number));
    }
});
```

B. Json dump code

```
import numpy as np
import xarray as xr
import matplotlib.pyplot as plt
import cartopy.feature as cfeature
import cartopy.io.shapereader as shpreader
from shapely.geometry import Point
from shapely.prepared import prep
import geopandas as gpd
import os
import json



variable_name = 't2m'  # Example: 2 meter temperature
file_path = 
'C:\\Users\\lukav\\Downloads\\yearTemperature\\12month_anomaly_Global_ea_2t_20
2001-202012_1981-2010_v02.grib'
ds = xr.open_dataset(file_path, engine='cfgrib')[variable_name]

sphere_detail_amount = [21, 36, 46, 54, 60, 65, 69, 73, 76, 78, 80, 82, 83, 
84, 84, 84, 84, 83, 82, 80, 78, 76, 73, 69, 65, 60, 54, 46, 36, 21]
sphere_detail = {i: [(0, 0, 0) for _ in range(sphere_detail_amount[i-1])] for 
i in range(1, len(sphere_detail_amount)+1)}

def calculate_equal_area_latitude_bins(num_rings):
```

```
    latitudes = np.linspace(90, -90, num_rings + 1)
    return [(latitudes[i], latitudes[i+1]) for i in range(num_rings)]

def calculate_longitude_bins(num_leds):
    longitudes = np.linspace(180, -180, num_leds + 1)
    return [(longitudes[i], longitudes[i+1]) for i in range(num_leds)]



def map_data(ds):
    sphere_detail = {i: [0 for _ in range(sphere_detail_amount[i-1])] for i in
range(1, len(sphere_detail_amount)+1)}
    latitude_bins =
calculate_equal_area_latitude_bins(len(sphere_detail_amount))

    shapefile = shpreader.natural_earth(resolution='50m', category='physical',
name='land')
    land_gdf = gpd.read_file(shapefile)

    def is_land(lat, lon):
        point = Point(lon, lat)
        return any(land_gdf.contains(point))

    for lat_idx, (lat_start, lat_end) in enumerate(latitude_bins):
        lat_slice = ds.sel(latitude=slice(lat_start, lat_end))
        num_leds = sphere_detail_amount[lat_idx]
        longitude_bins = calculate_longitude_bins(num_leds)
        longitude_bins = list(reversed(longitude_bins))
        # Iterate over each longitude bin for the current latitude
        for lon_idx, (lon_start, lon_end) in enumerate(longitude_bins):
            lon_slice = lat_slice.sel(longitude=slice(lon_start, lon_end))

            # interpolated_slice =
lon_slice.interp(latitude=lat_slice.latitude, longitude=lon_slice.longitude,
method='cubic')

            # Check if the center of the bin is land or ocean
            center_lat = (lat_start + lat_end) / 2
            center_lon = (lon_start + lon_end) / 2
            if is_land(center_lat, center_lon):
                sphere_detail[lat_idx + 1][lon_idx] = (1, 1, 0)  # Yellow for
land
            else:
                sphere_detail[lat_idx + 1][lon_idx] = (0, 0, 1)  # Blue for
ocean
    return sphere_detail

max_length = max(sphere_detail_amount)
```

```
# Initialize an empty array to store the colors for the plot
output_folder =
'c:\\Users\\lukav\\Documents\\graduationScripts\\sphere_detail_output'

ds = xr.open_dataset(file_path, engine='cfgrib')[variable_name]

# Calculate the sphere_detail for the dataset
sphere_detail = map_data(ds)

# Create the output file path
output_file_path = os.path.join(output_folder, f'continents.json')

# Dump the sphere_detail to a JSON file
with open(output_file_path, 'w') as f:
    json.dump(sphere_detail, f)
```

C. Data conversion code

```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from scipy.interpolate import griddata
import numpy as np
import xarray as xr

variable_name = 't2m'  # Example: 2 meter temperature
file_path =
'C:\\Users\\lukav\\Downloads\\yearTemperature\\12month_anomaly_Global_ea_2t_19
8101-198112_1981-2010_v02.grib'
ds = xr.open_dataset(file_path, engine='cfgrib')[variable_name]

def calculate_equal_area_latitude_bins(num_rings):
    sin_latitudes = np.linspace(1, -1, num_rings + 1)
    latitudes = np.degrees(np.arcsin(sin_latitudes))
    return [(latitudes[i], latitudes[i+1]) for i in range(num_rings)]

def calculate_longitude_bins(num_leds):
    longitudes = np.linspace(0, 360, num_leds + 1)
    return [(longitudes[i], longitudes[i+1]) for i in range(num_leds)]

def interpolate_data(xarray_data, target_latitudes, target_longitudes):
    lat_vals = xarray_data.latitude.values
    lon_vals = xarray_data.longitude.values
    data_vals = xarray_data.values

    lon_grid, lat_grid = np.meshgrid(target_longitudes, target_latitudes)
    points = np.array([(lat, lon) for lat in lat_vals for lon in lon_vals])
    values = data_vals.flatten()
```

```
    grid_z = griddata(points, values, (lat_grid, lon_grid), method='linear')
    return grid_z

def map_data_to_leds(xarray_data, ring_led_counts):
    num_rings = len(ring_led_counts)
    latitude_bins = calculate_equal_area_latitude_bins(num_rings)

    target_latitudes = [(lat_start + lat_end) / 2 for lat_start, lat_end in
latitude_bins]
    target_longitudes = np.concatenate([np.linspace(0, 360, count,
endpoint=False) for count in ring_led_counts])

    interpolated_data = interpolate_data(xarray_data, target_latitudes,
np.unique(target_longitudes))

    led_mapping = {}

    for i, (lat_start, lat_end) in enumerate(latitude_bins):
        ring_led_count = ring_led_counts[i]
        longitude_bins = calculate_longitude_bins(ring_led_count)

        for j, (lon_start, lon_end) in enumerate(longitude_bins):
            # Get indices of longitudes within the current bin
            lon_indices = np.where((target_longitudes >= lon_start) &
(target_longitudes < lon_end))[0]

            # Ensure indices are within bounds
            lon_indices = lon_indices[lon_indices <
interpolated_data.shape[1]]

            if len(lon_indices) > 0:
                avg_value = interpolated_data[i, lon_indices].mean()
            else:
                avg_value = np.nan

            led_mapping[(i, j)] = avg_value

    return led_mapping


def plot_led_sphere(led_mapping, ring_led_counts):
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')

    for (i, j), value in led_mapping.items():
        lat_bin = calculate_equal_area_latitude_bins(len(ring_led_counts))[i]
        lon_bin = calculate_longitude_bins(ring_led_counts[i])[j]

        lat = (lat_bin[0] + lat_bin[1]) / 2
```

```
        lon = (lon_bin[0] + lon_bin[1]) / 2

        lat_rad = np.radians(lat)
        lon_rad = np.radians(lon)

        x = np.cos(lat_rad) * np.cos(lon_rad)
        y = np.cos(lat_rad) * np.sin(lon_rad)
        z = np.sin(lat_rad)

        ax.scatter(x, y, z, color=plt.cm.viridis(value))

    plt.show()

# Example usage with an xarray dataset `ds` and given ring LED counts:
ring_led_counts = [21, 36, 46, 54, 60, 65, 69, 73, 76, 78, 80, 82, 83, 84, 84]
led_mapping = map_data_to_leds(ds, ring_led_counts)
plot_led_sphere(led_mapping, ring_led_counts)
```

D. Plotting the raw data

```
import os
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
import cartopy.feature as cfeature
import xarray as xr
from matplotlib.widgets import Slider
import matplotlib.colors as mcolors

# Path to the directory containing your GRIB files
directory_path = ‘C:\\Users\\lukav\\Downloads\\yearTemperature’


# Variable you want to plot
variable_name = ‘t2m’  # Example: 2 meter temperature

# Get a sorted list of GRIB files
grib_files = sorted([f for f in os.listdir(directory_path) if
f.endswith(‘.grib’)])

# Extract years from filenames and create a dictionary mapping
file_years = {int(f.split(‘_’)[5][:4]): f for f in grib_files if
f.endswith(‘.grib’)}

# Create a figure and axis for the plot
fig, ax = plt.subplots(figsize=(10, 5), subplot_kw={‘projection’:
ccrs.Mollweide()})
```

```
# Dictionary to store preloaded data
preloaded_data = {}

# Preload data for all years
all_data = []
for year, file_name in file_years.items():
    file_path = os.path.join(directory_path, file_name)
    ds = xr.open_dataset(file_path, engine='cfgrib')
    preloaded_data[year] = ds[variable_name]
    all_data.append(ds[variable_name])
    ds.close()


# Concatenate all data along the time dimension
all_data = xr.concat(all_data, dim='time')
vmin = all_data.min()
vmax = all_data.max()
colors = [(0, 'red'),(0.5, 'white'), (1, 'blue')]  # Positions and colors
cmap = mcolors.LinearSegmentedColormap.from_list('blue_white_red', colors)

# Create initial plot
data = preloaded_data[min(file_years.keys())]
lats = data.latitude.values
lons = data.longitude.values
plot = ax.imshow(data, origin='lower', extent=(lons.min(), lons.max(),
lats.min(), lats.max()),
                 transform=ccrs.PlateCarree(), cmap=cmap, vmin=-3, vmax=3)
ax.coastlines()
ax.add_feature(cfeature.BORDERS)

cbar = plt.colorbar(plot, orientation='horizontal', pad=0.05, ax=ax)
cbar.set_label(variable_name)
ax.set_title(f'{variable_name} for the year {min(file_years.keys())}')

# Function to update the plot for a given year
def update_plot(val):
    year = int(slider.val)

    # Get the data for the selected year from preloaded data
    data = preloaded_data[year]
    # Update contour levels with new data
    plot.set_data(data.values)
    # Update title
    ax.set_title(f'{variable_name} for the year {year}')

    # Redraw the figure
    fig.canvas.draw()

# Create a slider
```

```python
ax_slider = plt.axes([0.1, 0.02, 0.8, 0.03])
slider = Slider(ax_slider, 'Year', min(file_years.keys()),
max(file_years.keys()), valinit=min(file_years.keys()), valstep=1)

# Attach the update function to the slider
slider.on_changed(update_plot)

plt.show()
```

E. Dashboard code

```cpp
#include <ArduinoHttpClient.h>
#include <WiFi.h>
#include <ArduinoJson.h>

// Wifi settings
char* ssid = "Interaction Lab HMI-435 2.4GHz";
char* pass = "h!9N8Jlwo2";

char serverAddress[] = "192.168.50.168";  // server address will be given to
you
int port = 5000;
char path[] = "/controller";  // if you are working on the brain component,
put /brain for instance

WiFiClient wifi;
WebSocketClient client = WebSocketClient(wifi, serverAddress, port);

int status = WL_IDLE_STATUS;

int discreteKnob = 33;
int continuousKnob = 34;
int slider = 32;

int years[] = { 1980, 1985, 1990, 1995, 2000, 2005, 2010, 2015, 2020 };

int allDiscreteKnobRanges[] = { 100, 300, 650, 1000, 1600, 1800, 2200, 2500,
2900, 3300, 3980 };
int discreteKnobRanges[] = { 2900, 2500, 2200, 1800, 1600, 1000, 650, 300 };
int continuousKnobRanges[] = { 120, 650, 1200, 1700, 2350, 2850, 3350, 3900 };
int sliderRanges[] = { 70, 600, 1150, 1650, 2260, 2750, 3400, 4030 };
int year = 0;
int year1 = 0;
int year2 = 0;
int previousYear = -1;  // Initialize to a value that won't match
int previousYear2 = -1;
int previousYear1 = -1;

void connectSocketClientIfNotConnected() {
  if (!client.connected()) {
```

```
    client.begin(path);
    Serial.println("Connected");
    delay(500);
  }
}

void sendToServer(int sliderValueThing, int knob1, int knob2) {
  // Make a JSON document so the webserver can easily do something with this.
  StaticJsonDocument<JSON_OBJECT_SIZE(100)> doc;
  doc["slider"] = sliderValueThing;
  doc["knob1"] = knob1;
  doc["knob2"] = knob2;

  String jsonMessage = "";
  serializeJson(doc, jsonMessage);  // Put the message in the string
jsonMessage

  client.beginMessage(TYPE_TEXT);  // Begin sending the message in Json using
websockets
  client.print(jsonMessage);
  client.endMessage();
}

void setup() {
  // Put your setup code here, to run once:
  Serial.begin(115200);
  WiFi.begin(ssid, pass);
  Serial.print("Connecting to WiFi ..");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(WiFi.status());
    delay(1000);
  }
  // Print your WiFi IP address:
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);
  connectSocketClientIfNotConnected();
}

unsigned long previousTime = 0;
void loop() {
  unsigned long currentTime = millis();
  if (currentTime - previousTime > 50) {
    previousTime = currentTime;
    int discreteKnobValue = analogRead(discreteKnob);
    for (int i = 0; i < 8; i++) {
      if (discreteKnobValue >= discreteKnobRanges[i]) {
        year = years[i];
```

```
        break;
      } else {
        year = years[8];
      }
    }

    //------------------------------continuous knob--------------------------
-------
    int continuousKnobValue = analogRead(continuousKnob);
    for (int i = 0; i < 8; i++) {
      if (continuousKnobValue < continuousKnobRanges[i]) {
        year1 = years[i];
        break;
      } else {
        year1 = years[8];
      }
    }

    //------------------------------slider--------------------------------
    int sliderValue = analogRead(slider);
    for (int i = 0; i < 8; i++) {
      if (sliderValue < sliderRanges[i]) {
        year2 = years[i];
        break;
      } else {
        year2 = years[8];
      }
    }
  }

  if (previousYear != year || previousYear2 != year2 || previousYear1 !=
year1) {
    // sendToServer(year2, year, year1);
    Serial.print("Discrete Year: ");
    Serial.println(year);
    Serial.print("Slider Year: ");
    Serial.println(year2);
    Serial.print("Continuous Year: ");
    Serial.println(year1);
    sendToServer(year2, year, year1);  //knob 1 is the discrete knob

    // Update previous values
    previousYear = year;
    previousYear2 = year2;
    previousYear1 = year1;
  }

  // delay(50);  // Wait for half a second before the next loop
```

```
}
```

F. Hemisphere code

```
#include <ArduinoHttpClient.h>
#include <WiFi.h>
#include <ArduinoJson.h>
#include <Adafruit_NeoPixel.h>

// Define the number of strips and their respective lengths
#define NUM_STRIPS 7
#define WS_TX_BUFFER_SIZE 7000

// Lengths and pins for each strip
int stripLengths[NUM_STRIPS] = { 103, 179, 142, 154, 162, 167, 84 };
int stripPins[NUM_STRIPS] = { 22, 21, 19, 5, 16, 0, 15 };
//Don't forget to install these libraries if you don't have them locally
uint8_t ledData[3964];

long previousTime = 0;
long deltaTime = 0;

int maxFrequency = 500;

// Create separate instances of Adafruit_NeoPixel for each strip
Adafruit_NeoPixel strips[] = {
  Adafruit_NeoPixel(stripLengths[0], stripPins[0], NEO_GRB + NEO_KHZ800),
  Adafruit_NeoPixel(stripLengths[1], stripPins[1], NEO_GRB + NEO_KHZ800),
  Adafruit_NeoPixel(stripLengths[2], stripPins[2], NEO_GRB + NEO_KHZ800),
  Adafruit_NeoPixel(stripLengths[3], stripPins[3], NEO_GRB + NEO_KHZ800),
  Adafruit_NeoPixel(stripLengths[4], stripPins[4], NEO_GRB + NEO_KHZ800),
  Adafruit_NeoPixel(stripLengths[5], stripPins[5], NEO_GRB + NEO_KHZ800),
  Adafruit_NeoPixel(stripLengths[6], stripPins[6], NEO_GRB + NEO_KHZ800),
};



//Wifi settings
char* ssid = "Interaction Lab HMI-435 2.4GHz";
char* pass = "h!9N8Jlwo2";

char serverAddress[] = "192.168.50.168";  // server address will be given to
you
int port = 5000;
char path[] = "/north";  //if you are working on the brain component, put
/brain for instance
```

```
WiFiClient wifi;
WebSocketClient client = WebSocketClient(wifi, serverAddress, port);

int status = WL_IDLE_STATUS;
int count = 0;

unsigned long currentTime;

void connectSocketClientIfNotConnected() {
  if (!client.connected()) {
    client.begin(path);
  }
}

void setup() {
  //setup serial connection and wifi
  // Serial.begin(115200);
  //WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, pass);
  // Serial.print("Connecting to WiFi ..");
  while (WiFi.status() != WL_CONNECTED) {
    // Serial.print(WiFi.status());
    delay(1000);
  }
  // print your WiFi IP address:
  IPAddress ip = WiFi.localIP();
  // Serial.print("IP Address: ");
  // Serial.println(ip);



  ///////////////////////////
  //You can start coding setup here:
  /////////////////////////////
  for (int i = 0; i < NUM_STRIPS; i++) {
    strips[i].begin();
    strips[i].show();
    delay(50);
  }
  //////////////////////////
}

void setStrip() {
  int index = 0;  // Starting index in ledData array
  unsigned long currentTime = millis();
  for (int i = 0; i < NUM_STRIPS; i++) {
    int stripLength = stripLengths[i];
    for (int j = 0; j < stripLengths[i]; j++) {
      int r = ledData[index++];
```

```
      int g = ledData[index++];
      int b = ledData[index++];
      int interval = ledData[index++] * 100;  // Blink rate from 0 (always on)
to 255 (fastest blink)

      float dimmingFactor;
      if (interval == 0) {
        dimmingFactor = 1.0;
      } else {
        // Map blinkRate to a time interval for blinking
        int phase = currentTime % interval;  //j is the offset calculated
based on led
        int halfInterval = interval / 2;
        if (phase < halfInterval) {
          dimmingFactor = (sin((float)phase / halfInterval * 2 * PI) + 1) / 2;
          // dimmingFactor = 1.0 - (float)phase / halfInterval;
        } else {
          // dimmingFactor = (float)(phase - halfInterval) / halfInterval;
          dimmingFactor = (sin(((float)phase - halfInterval) / halfInterval *
2 * PI) + 1) / 2;
        }  // Calculate the phase of the blink using modulus to prevent
overflow issues
        if (dimmingFactor > 1.0) {
          dimmingFactor = 1.0;
        }  // dimmingFactor = (sin(phase * 2 * PI) + 1) / 2;  // Creates a
smooth sinusoidal dimming
      }

      // Apply the dimming factor to the color components
      r = r * dimmingFactor;
      g = g * dimmingFactor;
      b = b * dimmingFactor;
      strips[i].setPixelColor(j, r, g, b);
    }
  }
  for (int i = 0; i < NUM_STRIPS; i++) {
    strips[i].show();
  }
}

void loop() {
  // long currentTime = millis();
  // currentTime = millis();


  connectSocketClientIfNotConnected();
  // Serial.println(client.available());
  //checking if message has been 80eceived
  int messageSize = client.parseMessage();
```

```
  if (messageSize > 0) {

    if (messageSize != 3964) {
      return;
    }
    int bytesRead = client.read((uint8_t*)ledData, messageSize);

    if (bytesRead != messageSize) {
      return;
    }
  }
  setStrip();
  // delay(10);
  //END OF MESSAGE BLOCK
}
```

G. Briefing letter

What are the possible benefits or risks of participation?
Participation in this research project does not provide any direct benefits for you. However, your contribution may lead to very valuable data to improve the installation. As for risks, this research project does not have risks for you, physically or psychologically. This research project has been reviewed by the Ethics Committee Information and Computer Science.

What happens if you do not wish to participate in this research?
You may voluntarily decide whether you wish to participate in this research project. If you decide not to participate, you do not need to take any action. You also do not have to provide a reason for not wanting to participate in the research project. If you do decide to participate, you still may, at any moment of the research project, decide to stop your participation. Any data gathered before the withdrawal of participation will be deleted.

What will be done with your data?
If you participate in the study, your data will be gathered. Your data may only be used for the study and will never be publicly released. You will not be identifiable in any publication or presentation about the study. Personal data, including age, gender, and profession, will only serve for demographical purposes and will not be used to identify you. You will get a copy of your answers once we have processed the information, so you can correct any interpretation mistakes. Your data and audio recordings will be deleted once the project is finished.

Where can you ask any other questions?
Before you participate in this study, you have the opportunity to ask any questions you may have.
If you have any questions about or issues with the study, please do not hesitate to contact the researchers (see contact information below). Please also find, accompanied to this study brochure, the informed consent that you will sign before the start of the experiment.

Thank you in advance for your cooperation in this study.
Yours sincerely,
María Cobo Muñoz and Luka van Hoeven

Creative Technology
Faculty of EEMCS
University of Twente

UNIVERSITY OF TWENTE.

H. Consent form

Consent Form for Data physicalization of climate change data using an interactive globe

YOU WILL BE GIVEN A COPY OF THIS INFORMED CONSENT FORM

Please tick the appropriate boxes	Yes	No
Taking part in the study		
I have read and understood the study information dated [DD/MM/YYYY], or it has been read to me. I have been able to ask questions about the study and my questions have been answered to my satisfaction.	☐	☐
I consent voluntarily to be a participant in this study and understand that I can refuse to answer questions and I can withdraw from the study at any time, without having to give a reason.	☐	☐
I understand that taking part in the study involves that the interview will be audio recorded and saved. It will be deleted at the completion of the research or when requested. We will not transcribe the interview.	☐	☐
Use of the information in the study		
I understand that information I provide will be used for the user evaluation of the installation.	☐	☐
I understand that personal information collected about me that can identify me, such as [e.g. my name or where I live], will not be shared beyond the study team.	☐	☐
I agree that my information can be quoted in research outputs.	☐	☐
Future use and reuse of the information by others		
I give permission for the corresponding report to be archived in Mobility online so it can be used for future research and learning.	☐	☐
I give the researchers permission to keep my contact information and to contact me for future research projects.	☐	☐

Signatures

____________________ ____________________ ________

Name of participant Signature Date

I have accurately read out the information sheet to the potential participant and, to the best of my ability, ensured that the participant understands to what they are freely consenting.

______________________ _________________ ________

Researcher name Signature Date

UNIVERSITY OF TWENTE.

Study contact details for further information:
Luka van Hoeven (l.k.vanhoeven@student.utwente.nl),
María Cobo Muñoz (m.cobomunoz@student.utwente.nl)

Contact Information for Questions about Your Rights as a Research Participant

If you have questions about your rights as a research participant, or wish to obtain information, ask questions, or discuss any concerns about this study with someone other than the researcher(s), please contact the Secretary of the Ethics Committee Information & Computer Science: ethicscommittee-CIS@utwente.nl

UNIVERSITY OF TWENTE.

I. Survey

LimeSurvey

Section A: To be filled in by the researchers

A1. **Participant number**

Section B: Demographics

B1. **What is your age?**

B2. **What is your gender?**

Female ☐
Male ☐
Non-binary ☐

B3. **Are you colorblind?**

Yes ☐
No ☐

B4. **How would you rate your knowledge about climate change?**

No knowledge	Beginner	Somewhat knowledgable	Knowledgable	Expert
☐	☐	☐	☐	☐

B5. **How familiar are you with data physicalizations?**

Not at all	A little bit	Somewhat	Very
☐	☐	☐	☐

B6. **How familiar are you with reading geographical data? (How good you are reading maps, how often, etc.)**

Not at all	A little bit	Somewhat	Very
☐	☐	☐	☐

LimeSurvey

Section C: Exploration

Feel free to expore the functionality of the globe, once you feel comfortable go to the next part of the survey where you complete tasks related to the data being shown. Use this time to familiarize yourself with the globe. Once you feel comfortable press next.

Section D: Tasks

You will be asked questions about the data that is being shown

D1. In the last decade (2010-2020) has Africa become more warm or more cool in general?

Warmer ☐

Cooler ☐

D2. In 1985, what hemisphere has the biggest temperature change, southern or northern? Keep in mind that change includes warmer as well as colder temperatures.

Northern hemisphere ☐

Southern hemisphere ☐

D3. What area has the highest temperature anomaly in 2000? (Warmer)

North pole ☐

North America ☐

Africa ☐

D4. Is the equator warmer in 1990 compared to 2020?

Yes ☐

No ☐

Section E: Evaluation

E1. Answer the following questions about satisfaction

	Strongly disagree						Strongly agree
Overall, I am satisfied with the ease of completing these tasks.	☐	☐	☐	☐	☐	☐	☐
Overall, I am satisfied with the amount of time it took to complete these tasks.	☐	☐	☐	☐	☐	☐	☐

E2. Answer the following questions about mental and physical demand.

	Very low						Very high
How mentally demanding was these tasks?	☐	☐	☐	☐	☐	☐	☐
How physically demanding was these tasks?	☐	☐	☐	☐	☐	☐	☐

asd

LimeSurvey

E3. **Fill in the following in regards to your experience with the physicalization**

	1	2	3	4	5	6	7
obstructive I supportive	☐	☐	☐	☐	☐	☐	☐
complicated I easy	☐	☐	☐	☐	☐	☐	☐
inefficient I efficient	☐	☐	☐	☐	☐	☐	☐
confusing I clear	☐	☐	☐	☐	☐	☐	☐
boring I exciting	☐	☐	☐	☐	☐	☐	☐
not interesting I interesting	☐	☐	☐	☐	☐	☐	☐
conventional I inventive	☐	☐	☐	☐	☐	☐	☐
usual I leading edge	☐	☐	☐	☐	☐	☐	☐
	☐	☐	☐	☐	☐	☐	☐

E4. **Please fill in any questions, remarks or feedback about the physicalization and your experience.**

J

```
import numpy as np
TOTALLEDSAVAILABLE = 35 * 60
LEDSPERMETER = 60

def heightFromTop(row, diameter, rowAmount):
    return ((diameter) / rowAmount) * (row) + (diameter/rowAmount)/2

def radiusBasedOnRow(row, diameter, rowAmount):
    heightFromCentre = (diameter/2) - heightFromTop(row, diameter, rowAmount)
    radius = np.sqrt((diameter/2)**2 - heightFromCentre**2)
    return radius

def detailBasedOnRowAndRadiusOfRow(rowRadius):
    circumfrence = rowRadius* 2 * np.pi
    return circumfrence * LEDSPERMETER

def roundUpDetailAndGetRadius(details):
    ceiling = np.ceil(details)
    return (ceiling/LEDSPERMETER) / (2 * np.pi)

def roundDownDetailAndGetRadius(details):
    ceiling = np.floor(details)
    return (ceiling/LEDSPERMETER) / (2 * np.pi)

def heightBasedOnRadius(radius, maxDiameter):
    maxRadius = maxDiameter / 2
    heightFromCentre = np.sqrt(maxRadius**2 - radius**2)
    return maxDiameter/2 - heightFromCentre

def radiusBasedOnHeight(height, maxDiameter):
    maxRadius = maxDiameter / 2
    heightFromCentre = maxDiameter - height
    return np.sqrt(maxRadius**2 - heightFromCentre**2)

def calcBasedOnDiameter(maxDiameter):
    details = 0
    rowAmount = int((maxDiameter) / 0.015)
    # print("rowAmount", rowAmount)
    for row in range(rowAmount):
        radiusOfRow = radiusBasedOnRow(row, maxDiameter, rowAmount)
        detailsBasedOnRow = detailBasedOnRowAndRadiusOfRow(radiusOfRow)
        nearestRadiusBasedOnDetail =
roundDownDetailAndGetRadius(detailsBasedOnRow)
        detailToHeight = heightBasedOnRadius(nearestRadiusBasedOnDetail,
maxDiameter)
        print("height", heightFromTop(row, maxDiameter, rowAmount) * 100 ,
"Diameter:",nearestRadiusBasedOnDetail * 200)
```

```
        details = details + np.ceil(detailsBasedOnRow)
    print(details)


calcBasedOnDiameter(0.45)
```

J. J

K.