Process book

1. Data collection

To gather the necessary data for our project, we relied on the WorldPop database. However, the dataset was organized by country and year, which meant we needed to perform web scraping to extract the required information. Our goal was to create individual CSV files for each country, spanning the years 2000 to 2020. These files contained population density data represented as a grid per square kilometer.

During the data collection process, we encountered a few challenges that required careful consideration. One of the main hurdles was dealing with the large volume of data points. The precision of population density per square kilometer was valuable, but it posed a challenge for the 3D visualization, particularly for loading the view. Therefore, we decided to decrease the precision of the dataset by averaging every 9 points into 1 single datapoint. This reduction in data points significantly reduced the computational load and improved the loading time, without sacrificing the overall visual representation of population density. We also decided to only focus on European countries as the amount of storage required for all the countries was too big and not realistic for a student project web server.

2. 2D Map

This part was mostly done in the 2nd milestone, but we still made some improvements.We changed the heatLayer to display a gradient of green color for better understanding of the population ratio. We add a legend and radius control on the right of the map. The radius value controls the size of the radius for merging the population density. We first wanted to remove this interaction because it was in the form of a slider and was ambiguous. This new approach is more intuitive and having the possibility to change the radius adds some interactivity to the map and a better understanding of the data. As we can affine the radius we can choose more or less regions to be merged.

We also add some animation for passing from the 2D map to the 3D one. As the 3D map takes some time to load and parse the data, we add a loading screen once all the data are ready to be visualized in 3D the three.js 3D scene is shown and the loading div is hidden. To go back to the 2D map there is a small cross on top right of the 3D scene view, clicking on it will hide the 3D scene and show the 2D map.

3. 3D Map

After building on the 2D map, we wanted to enhance the visualization by adding a 3D map with interactive elements. The goal of this map was to make the visualization more interactive and more entertaining for users. We used Three.js, a popular JavaScript library, to create the 3D environment and render the population data as cubes on the map. We also implemented camera controls for user navigation within the 3D space. Here, we can compare the visualization that we wanted at the beginning of the project and the visualization that we managed to do. We feel we made some really pleasant maps that are really intuitive to analyze and to interact with.

Implementing a fixed view of one year of one country was made easily by following common methods of Three.js. However, it was more challenging when we tried to move around or to change the year. The goal was that one user could easily navigate but also change the year without any loading process. To do so, we had to further improve the optimization methods of the three.js elements. We managed it by using some merged mesh for each year and only making visible the merged mesh of the chosen year.

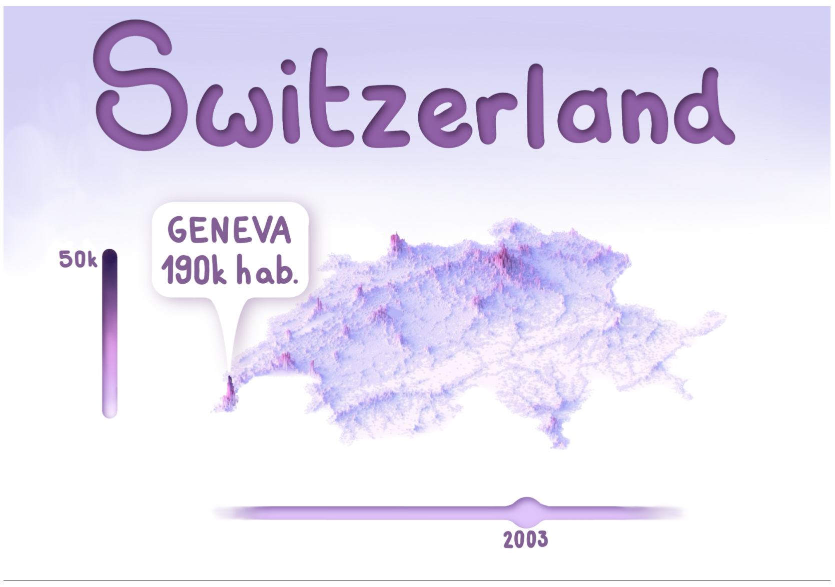
One issue we have nonetheless is that to switch from one country to another, you need to come back to the overall map, as it is not possible to load everything at the same time.

The camera controls were another issue that we had as there are various ways to move around. We tried several versions, with the first version being a fixed x,y movement with WASD and no camera rotation to a directional movement based on the direction of the camera with WASD and the scroll wheel paired with the mouse for camera rotations. This was challenging to implement as the camera controls in three.js are not that intuitive.

One limitation that we still have is that the loading of the csv files sometimes crashes with no reason. This could be linked to memory constraints or network issues during the loading process. We have been actively investigating this issue and exploring potential solutions to improve the stability and reliability of the loading mechanism without results.

4. Visual Identity and Interactions

The visual identity we wanted to give our data visualization was something simple, we didn't want the website to be overcrowded with information. We chose a color to keep a coherence in the website, we chose a blueish green we found pretty. We tried to replicate a soft gradient like we put in our original sketches.



We chose to have the 3D representation kept in blue and red because the contrast between the two colors is huge and so we were easily able to tell the different values apart. Lighter colors were not that visible because of the dark shadows, and softening the shadows required too much GPU usage.

The interactions on the 3D map, while possible, were complicated. We thought that only showing data for one square of the map wasn't very interesting and we couldn't find an intuitive way to use a brushing cursor on the 3D map.
Therefore, we only added a zone selection interaction on the 2D map only. It was hard to do a brushing zone on a map, as the cursor goes to grabbing to move the map, so we used the Leaflet Select Area module which permits us to use the selection tool while pressing control. We decided to allow the selection mode only when a country has already been chosen, to be able to easily plot the graphs for the whole country. The problem we still face for the graph was to plot it with the dates going from the year 2000. We also had trouble with making it appear and disappear.

5. Peer Assessment

We divided the work as follows:

- Data collection: everyone
- 2D Map: Antonio Pisanello
- 3D Map: Benjamin Krieger
- Visual Identity and Interactions: Romane Clerc