Data Visualization

Process Book

Social interactions on the /r/place

Damien Martin
Marc Jollès
Yassin Kammoun

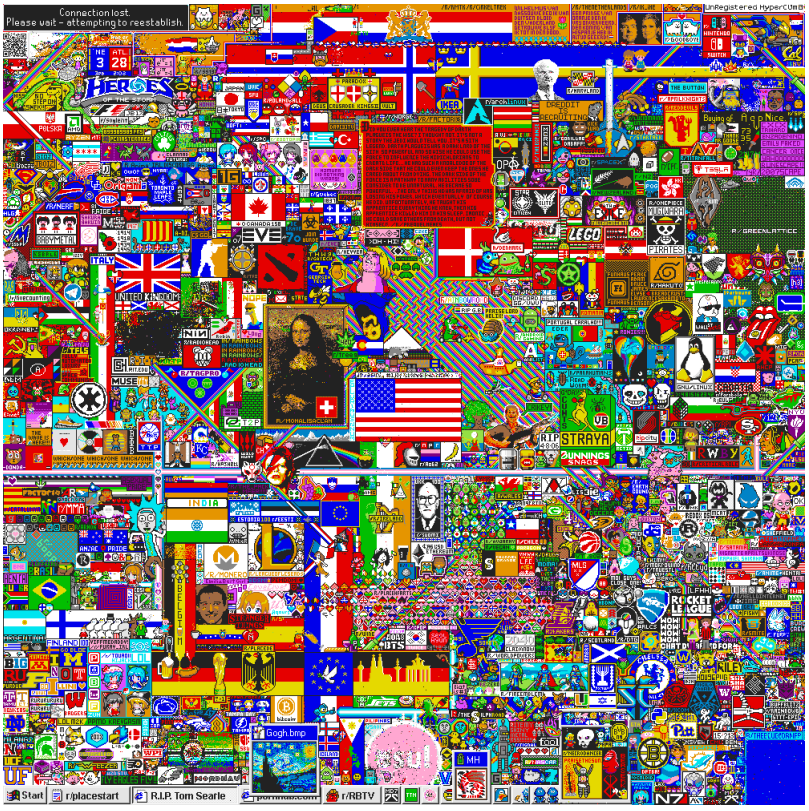
December 21, 2018

EPFL

Introduction

A good way to understand the human behavior is to figure out how people interact with each other. The /r/place event is a perfect case study of online interactions between individuals, with cooperation, competition and pure destruction of other people's work.

This event took place on April 2017, corresponding to one of the special events organized every year by the social media Reddit. Each person could modify one pixel, in the 1000 by 1000 pixels canvas, every 5 to 20 minutes, picking a color from a specific color palette. Some people could think that such a drawing framework would lead to chaos, with random colors placed at random locations.
However, one must also take into account communication means offered by Reddit: subreddits. Thanks to these Reddit communication channels, a person can be part of multiple groups, organizing and sharing some work with other people, to create some shapes in specific locations. And obviously, since we are dealing with internet communities, pop culture references, flags and memes were the main characters of this decentralized play.



Here is the final result of the whole event, 72 hours after its beginning.

Motivation

The /r/place was an amazing event, showing a lot of memes and references that internet users can recognize. It can be linked to other events such as TwitchPlaysPokemon, in which some order emerges from planned chaos and randomness. Also, even if we didn't participate to this event, it is really interesting and funny for us to see what people can achieve in such a small period of time, in this tiny picture of internet history.

Target audience

Any Reddit member, or curious person about internet and online communities could enjoy such a visualization, so we would be very happy to share it on Reddit.

Related work

Given the popularity of the /r/place event, there were many works on the subject. Those works relied on various forms of the original dataset. We propose to retain three of these projects.

The /r/place Atlas is a project that aimed to chart all the artworks created during the /r/place April's fools event on Reddit in 2017. It comes in the form of an an interactive map of Reddit's /r/place, with information to each artwork of the canvas. The sidebar of the /r/place Atlas gave inspiration to our own sidebar. Furthermore, we extensively used the dataset on which the /r/place Atlas was built on.

There are two other interesting projects for which there are were interactive visualization, but only videos were made available. First, the /r/Place 3D Render Timelapse *project* that created a 3D "visualization" with Blender of the /r/place event, produced a video out of it, and supposedly highlights the number of edits made for a given drawing. Then, the Reddit Place (/r/place) - FULL 72h (90fps) TIMELAPSE *project* that created a 2D "visualization" showing the evolution of the /r/place event's map during the 72 hours of the event.

The results of those three projects significantly inspired us in developing our own visualization. We basically managed to create a visualization that exactly proposes the contents of the two aforementioned videos. **We distinguish ourselves from these projects by the fact that our visualization is provided with additional features. Our visualization is, above all, interactive, real time, and defines multiple levels of abstractions, among which the detection and the highlighting of communities**.

Conception

With this visualization, we want to show the story of the /r/place, which is a collective artistic performance, with its evolution through time, interactions between communities and the strongest references which emerged from it.

Since our data represents user edits in a canvas, the first idea we had was to show them in an interactive map. We wanted to display statistics about canvas pixels, including the number of edits, colors, and the number of unique users who clicked on this pixel.

To achieve this, we wanted a 3D map with one bar per pixel, evolving through time, and a way to filter edits by some particular color. Also, we wanted to achieve a

graph representing the different communities, with a way to filter only their edits, through a sidebar, containing a search bar and a list of communities.

To work with 3D, we chose to work with Three.js, which can ease the usage of OpenGL in the browser (called WebGL). This library allows us to render 3D objects as well as interacting with the 3D space, through panning, rotation and zoom capabilities.

After some investigation, we have seen that it's not possible to obtain a visualization with one bar per pixel, for performance and memory reasons : the canvas has 1000*1000 pixels, which would mean 1000*1000*8 vertices (a bit less with some optimizations) to create cubes with different heights. This would need too much processing power for WebGL, and it would need a lot of bandwidth to obtain the whole dataset, even if we partitioned it. The solution for that was to subdivide our 1000x1000 in 200*200 zones of 5*5 pixels each.

Since we want to see the evolution of the canvas with respect to time, we have integrated a timeline in the visualization. The corresponding /r/place time can be seen on the top left hand corner of the web page. In the timeline, a line chart shows the total number of edits made in each time partition. If some communities are selected in the sidebar, their edits are shown in a stacked way, so that one may compare their influence easily.

Time partitions have been introduced to load progressively data to the web browser, and is useful for some visualization choices also : we create background images for each 15-minute period, so we have less data to take into account.

For each zone, we set as its height the max of the edits made by users in this zone, in a 30 minutes period. We then introduce some interpolation between each time partition, so that when played, peaks can vary smoothly. The color used on peaks corresponds to the corresponding color of the canvas during that time partition, projected in the surface.

Given the amount of data we fetch for the canvas when loading the page, there is some delay before the visualization can start. Our visualization does not need all the data in the beginning to show the 3D map, so we allowed it to begin before having obtained all images (when half of the images were downloaded). However, after having tested it, it appears that it was not a good idea, as sometimes the network is capricious, leading to a black canvas instead of the correct image. As we have been able to improve the loading time, by using the APNG (Animated PNG) format, which takes way less time for loading the page, we assume that it is not a priority anymore.

In any case, the user needs to wait for a few seconds before the web app shows

anything relevant. To this end, we chose to display some kind of welcome screen. In such a screen, one can see the animated /r/place logo and some facts about the event. There is also a short tutorial on controls, with animated mouse, keyboard and touchpad images.

In Figure 1, it is possible to see a sketch of our first ideas on the visualization.

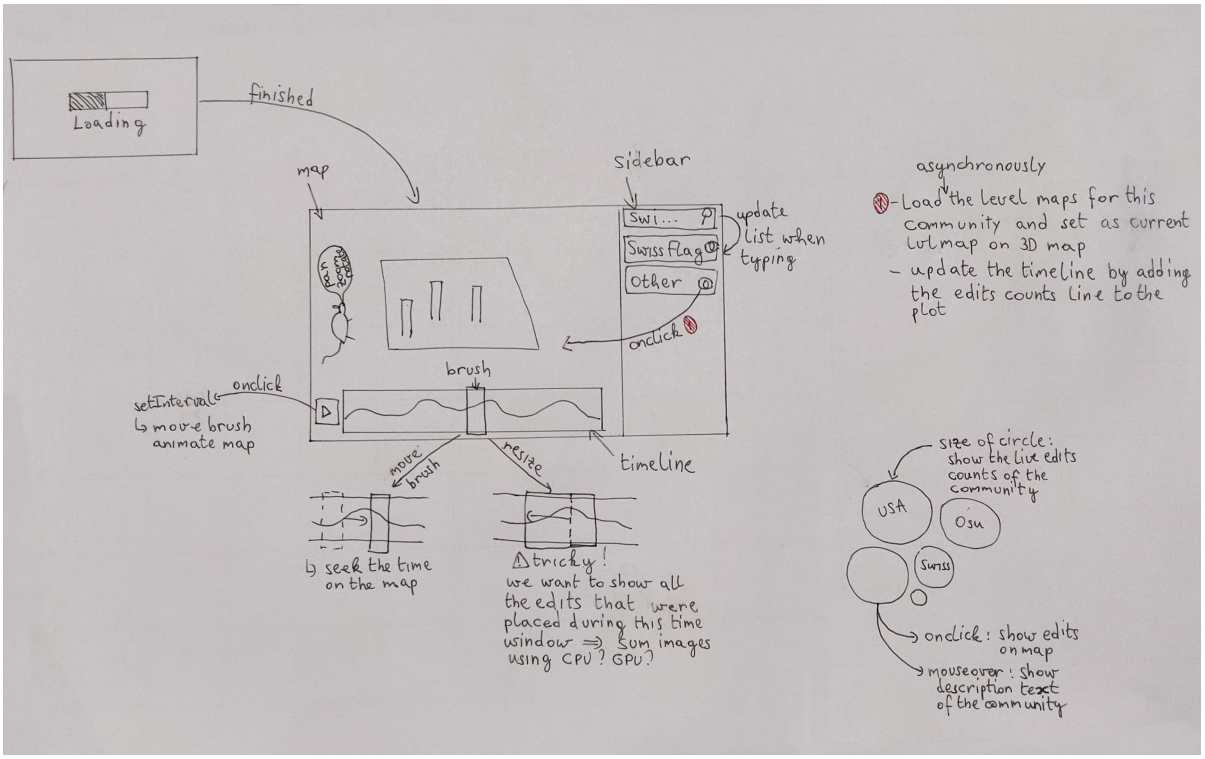


Figure 1: First version sketch

Dataset and Preprocessing

The format of the r/place dataset is very simple: a list of 16 millions edits, done by approximately 1.16 millions (unique) users, on a canvas of 1000x1000 pixels. For each edit we have: the timestamp, the hash of the user name, the x and y coordinates and the color of the pixel that is placed.

There are already plenty of visualizations existing on the web, showing the evolution of the canvas during the 72h, but we haven't found any which allows to get some

interesting insights about how all of those users were organized.

Our first steps were to get to know this fairly large dataset (892MB CSV file). The Figure 2 show the number of edits done by the users throughout the r/place event. We first noticed that the dataset contains samples from time periods lying outside the offical event duration. Figure 3 shows the cumulative activity for each pixel.

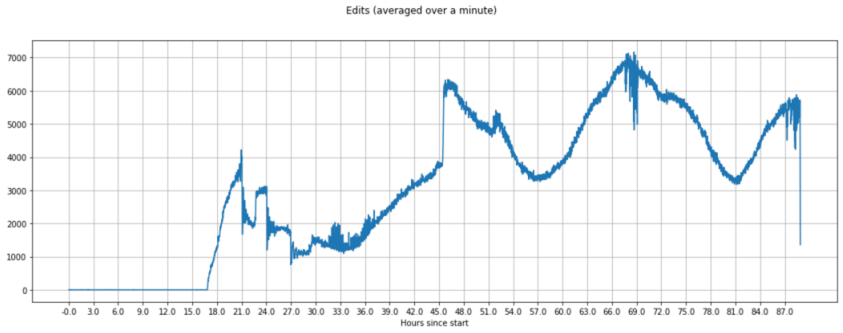


Figure 2: Number of edits through time (1 minute windows)

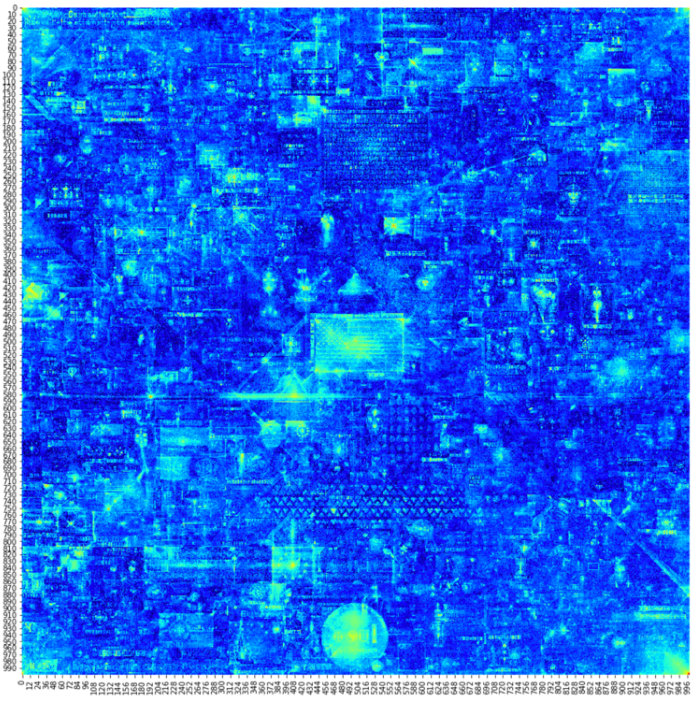
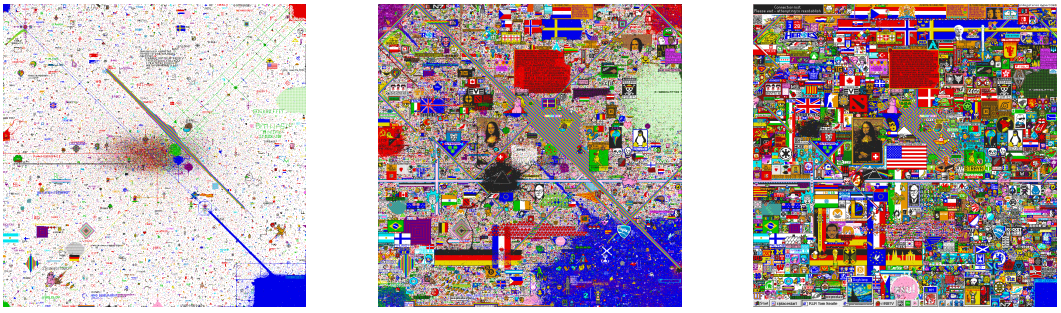


Figure 3: Heatmap of the user edits

Snapshots of background images

As mentioned, we choose to generate a background for each time period of 15 minutes. This choice was done to have a good trade-off between bandwidth usage versus animation smoothness. To generate each background, we simply read each line of the CSV dataset sequentially and draw pixels one after the other on a PIL image (python library). Then, we save these images to PNG format when we reach the end of the current time window, since we sample background images each 15 minutes.

In Figure 4, it's possible to see examples of such background images.



(a) frame 12 (b) frame 100 (c) last frame

Figure 4: Some generated backgrounds

All of the 290 generated backgrounds weight 153 MB, which is possible to download with a good internet connection, but still blocks the user for a long time. When reloading the page, web caching then comes into play and reduces significantly this delay, but the waiting time could be long for the first page loading. During the last week of the project, we found a way to drastically reduce this size to approximately 13 MB by combining all these backgrounds in an animated PNG file (APNG).

Communities edits

To delve into community analysis, some community detection preprocessing was necessary. Our first idea (which seems pretty natural in a graph setup) was to use the Louvain community detection algorithm, because it works relatively well in practice to detect communities and it runs in $O(n*lg(n))$. To use this algorithm, we tried to build a graph where nodes represent users and edges represent links between them. Building this graph is a difficult task for several reasons: it is not easy to define weights for the edges, and how to create a graph with that many nodes. After

some tests, we found that due to time and computational resources, this approach was not feasible.

So we changed the way to detect communities and decided to use the semantic segmentation (from the Reddit community) created by Reddit users[1], as shown in Figure 6. This segmentation is done on the final result of the /r/place canvas, and contains data on communities, such as their name ("Mona Lisa", "France Flag", ...), their description, and the path delimiting their contour. We used this path to obtain, for each community, the coordinates of all of its pixels, and then assigned users to the communities with this rule : a user is in a community if he put a pixel of the same color as the final color, in some fixed time window. We define the final color of a pixel as the most frequent color drawn on it within the five last edits. We then used those lists of users to filter the /r/place dataset in order to only have edits made by some specific communities.

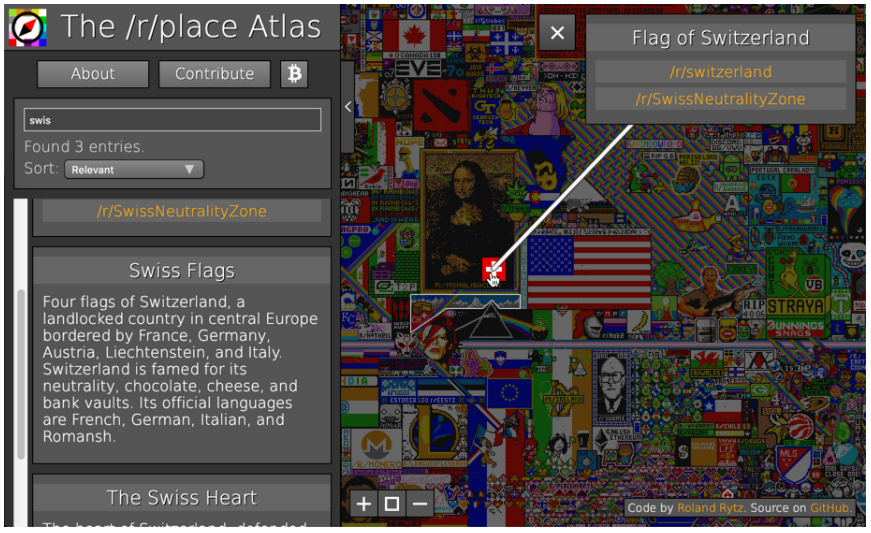


Figure 5: Place-atlas semantic segmentation

With these lists of users, as we wanted to display the number of edits in the visualization, we chose to show it in the third dimension. These edits would need to be linked to the time, so that we can see the progression of the canvas and of edits as the /r/place event evolves. We tackled this by first discretizing time in 30 minutes intervals. Then we computed a matrix containing for each pixel how many times it was modified, and then reduced its size by doing the max of each 5x5 pixels partitions. This matrix is then converted to a PNG file that represent a level map of the edits. These steps allowed us to reduce the size of the 145 level maps to approximately 1.5MB for each community.

[1]https://github.com/RolandR/place-atlas

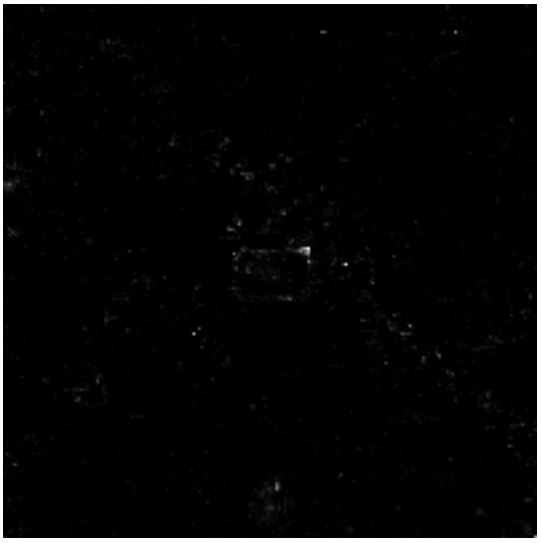


Figure 6: One of the generated level map (Rainbow Road)

The semantic segmentation contains almost 2000 communities, so we choose to generate level maps for the 150 biggest ones (by their number of users), as it would take a lot of time (and a lot of memory) to process all of them. Of course, it would not be hard to generate all of them, and our preprocessing notebook can do it by modifying one line of code.

During the preprocessing step, we also generated an image mask for each community. We show the mask of the Mona Lisa community in Figure 7. These masks were important to show to the user where the community is located on the /r/place, when he hovers over a community in the selection list.

Implementation

In this section we will discuss about implementation details of the visualization. We will describe the intent and features of the visualization. To this end, we will provide clear and well-referenced images showing the key design and interaction elements.

Tools

The implementation was done using JavaScript as the core programming language of the visualization. More specifically, the development followed as best as possible

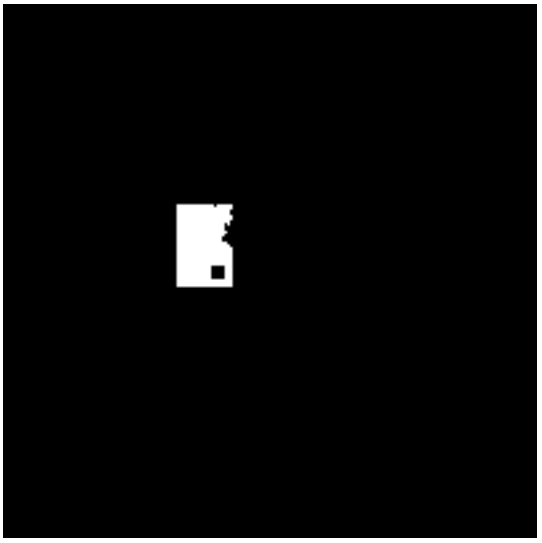


Figure 7: The mask of the Mona Lisa community

ECMAScript 6 best practices and used its features as much as possible when necessary. Furthermore, the visualization involved a certain number of tools, of which the main ones are the following:

- D3.js that we used for displaying the timeline, as well as for scaling and interpolating images represented as arrays.
- Vue.js that we used to enforce the MVVM pattern as the architecture of the visualization.
- three.js that we use for displaying the map and for manipulating the camera that comes along with it.

In addition to these well-known libraries in the JavaScript community, the visualization implementation relied on a few ad-hoc libraries for particular needs. These libraries will be introduced when we will talk about some functionalities of the visualization.

Functionalities

Loading screen

The visualization is preceded by a loading screen, whose purpose is to make users wait until the needed assets are downloaded and the map is built. Even though we managed to drastically speed up this non-obvious processing, there are still some

delays before the visualization is ready. For that reason, we provide the user with a few tips on how to interact with the visualization. Moreover, we display a few messages about the /r/place event to give some context (fun facts and important statistics) before discovering the visualization and playing with it.

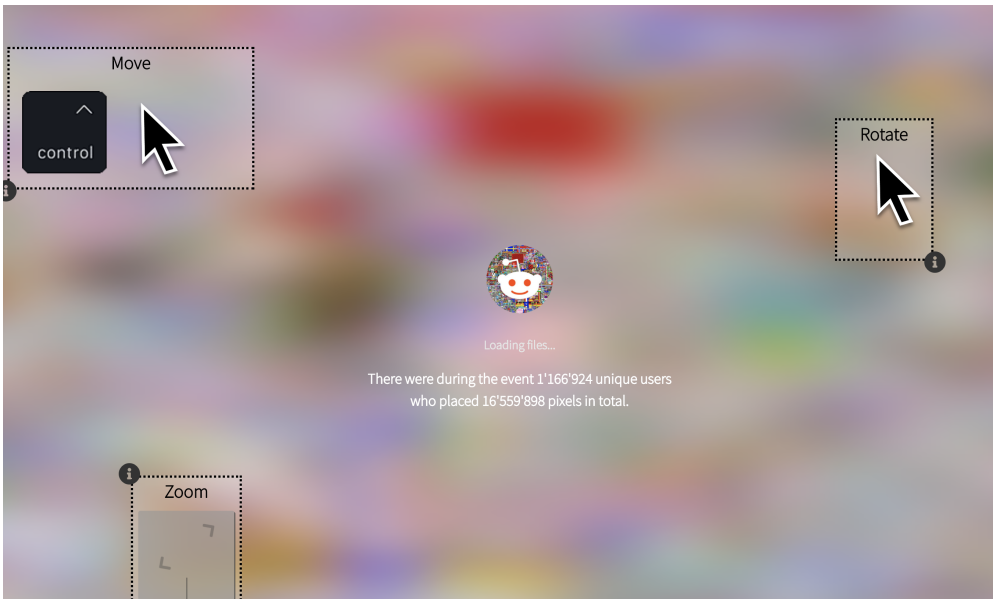


Figure 8: Loading screen

Tutorial

We have chosen to add a tutorial to the visualization. We already have some reminders about interaction keys in the welcome screen, however it is clearly not enough to show how to use the visualization, since we also have the timeline and the community panels, which deserve to be introduced.

For this tutorial, we have chosen the martini glass approach to introduce different features. First of all, we show the final image and explain very quickly the subject of the visualization. We completely disable all interactions with the map, the timeline and the sidebar at that moment. Then, we introduce map controls (pan, rotation and scale), so we only allow interactions with the map. Then, we show how the timeline works, and then how the sidebar works, each time giving a bit more interactivity to the user.

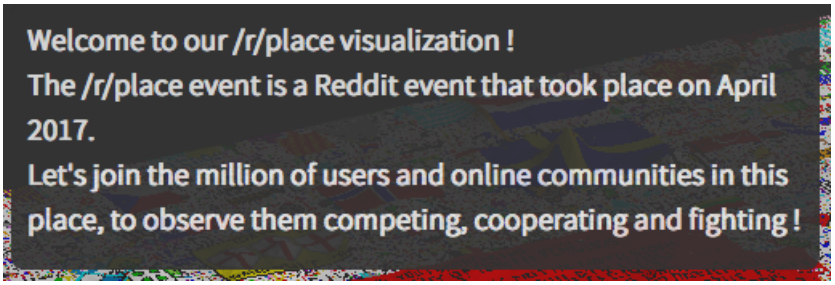


Figure 9: Tutorial message

For the sidebar, as there is a lot of content to present, and to avoid holding users hostage, we choose to have an embedded video in autoplay and loop mode, showing main characteristics of the panel. It appeared to be more easy and stimulating to learn about visualization features with such a video, so an open improvement could be to add videos also for the timeline and for map interactions.

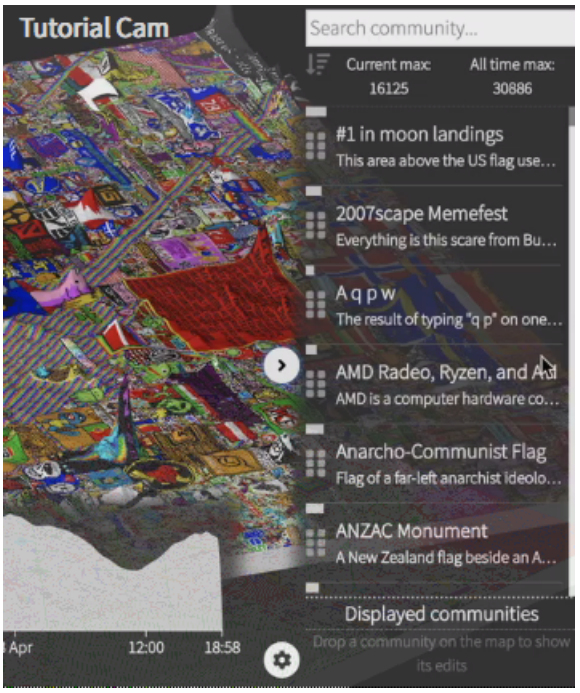


Figure 10: Sidebar tutorial video

Then, we introduce a case study : we want to show how two communities can fight each other, and then cooperate. The France flag and German flag communities were perfect for that, as at the beginning they were fighting each other, and in the end, the EU flag is drawn between these two flags, with cooperation between these two communities.

Right after, we still have a message indicating other great communities to have a look at, and we then release the user, so that he is able to use the visualization without any limitation.

Map

In the 3D map, we can see peaks, corresponding to user edits made during a small time period (depending on the selection in the timeline).

We use the three.js library to create the different 3D elements and to organize them. We have global scene to which we can add children objects, which will then be shown. The main scene children are a plane, with enough subdivisions (vertices) to have a peak per 200x200 level map pixel, and a scaled cube below (as a pedestal). We also add the camera to it, which is linked to the OrbitalControl, to allow panning, rotation and scaling interactions. We also have three lights added to the scene : two directional lights and an ambient light. The aim is to have shadows created by the peaks, but to avoid black shadows, so that when we rotate we can still see colors when peaks are big.

The plane background image has nearest pixel filters, so that we attenuate some blurring effects on this image. Such filters are the best in our case, since we are dealing with pixel art images, where each pixel is important. Of course, it suffers from aliasing, but the final render quality seemed pretty good with this approach.

Sidebar

On the right hand side of the visualization comes a sidebar, accessible by clicking on a *swipe* button as suggested by a teaching assistant.

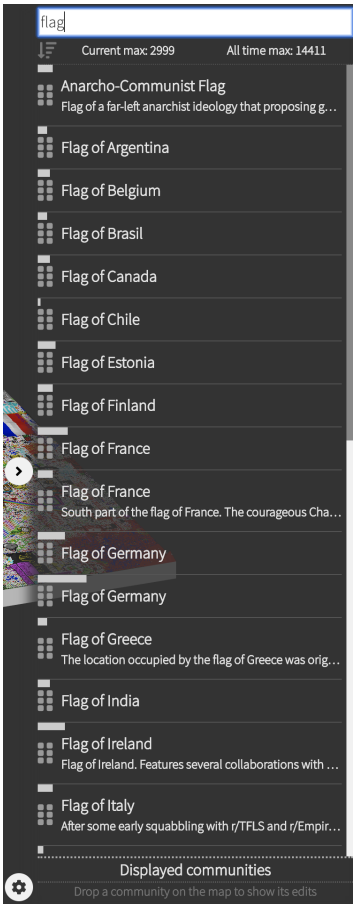


Figure 11: Sidebar

Swiping the sidebar and keeping it hidden by default made definitely sense. The map which focus the main attraction of the visualization takes quite some space. Therefore, minimizing the room occupied by the other components of the visualization was necessary. For that reason, it was decided to make visible the sidebar only on request.

This sidebar provides with a search bar that allows anyone to search for communities by keywords. The results of a search are sorted alphabetically. Those results are named and these are the names of the communities with respect to the dataset that was produced by the /r/place Atlas project.

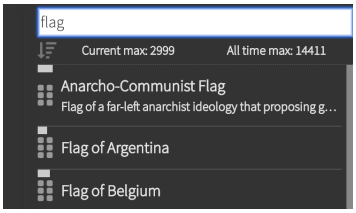


Figure 12: Sidebar's search bar

As listed in the search results, a community entry is essentially defined by a name and a description, the latter being truncated because of some characters limit that is applied by default. Nevertheless, clicking on the community shows the whole description of the community if necessary. Below the community entry is a progress indicator which indicates the cumulative number of edits made by the said community at a time t.

Figure 13: Sidebar's search entry

Right after the search bar, one can find two stats; the left most stats indicates the number of edits by the the most active community at time t while the right most stats indicates the number of instant editions made by the most active community during the whole event.

Whenever one hovers the mouse over a community, the community activity is highlighted in white on the map. More specifically, the highlighting of a community represents where lies the final result - the final drawing - of a community. This upper part of the sidebar also provides with a button that allows for sorting communities by the number of edits made at time t, i.e. ordering the communities by real time edits.

The sidebar hides another interesting feature that is the drag & drop of community entries from the sidebar to the map. The idea of such a feature is to focus the content of both the map and the timeline on the *drag & dropped* communities in order to

compare the activity of those communities. One can compare ten communities at most.

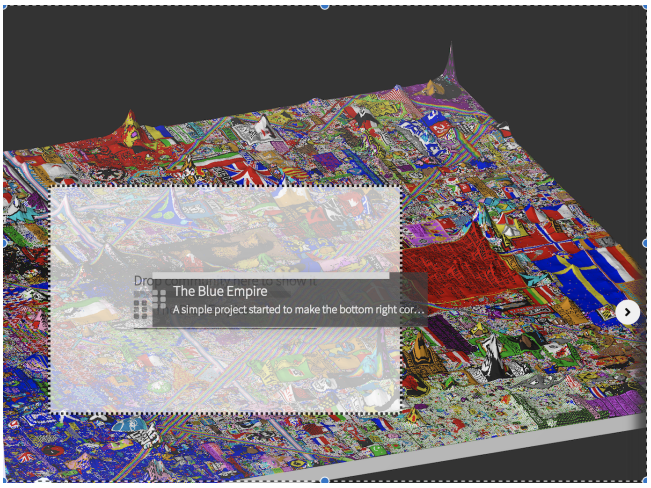


Figure 14: Sidebar's drag & drop

Communities that are dragged and dropped in the map are highlighted. What basically happens is that only those communities edits are considered and displayed on the map with respect to their activity during the /r/place event. The remaining communities are *flat*, i.e. their edits are not displayed at all on the map.

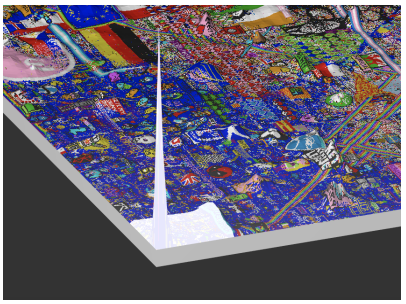


Figure 15: Sidebar's highlight when hovering a community entry

Dragged and dropped communities are not only highlighted on the map, but also on the sidebar. One can indeed see at the lower part of the sidebar a list of displayed

communities. The entries of this list are colored. They can also be reordered at will within the sidebar and the timeline, or even be removed from them if necessary by using a *trash* button.

Figure 16: Sidebar with displayed communities

Finally, at the very bottom of the sidebar are a few settings, here also accessible via a dedicated button. The *Smoothing* settings basically applies a Gaussian filter to the level maps to smooth the peaks in the third dimension that represent the number of edits for each pixel coordinate. This smoothing is applied independently on the level maps, i.e. a spatial smoothing rather than a temporal smoothing. The *Auto-rotate* setting makes the map rotate automatically.

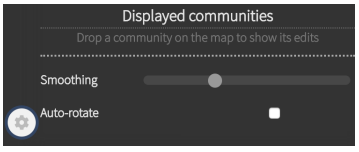


Figure 17: Sidebar settings

Timeline

The visualization comes with a timeline that controls the current state of the visualization in terms of time. The value of the time is bounded to the time period of the /r/place event. In addition to being the *clock* of the underlying application under the hood, the timeline is an interactive component that shows the progress of the /r/place event over time. To this end, the timeline provides a slider - a brush in D3.js 's jargon - that can be moved forward or backward to display a recent or an older state of the /r/place event, i.e. the drawings of Reddit communities on the map at a particular time.

Figure 18: Timeline

The slider of the timeline can progress in an automated way. This is made possible by the provision of a timeline control - a *play* button - that starts or stops the time.

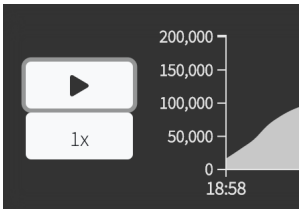


Figure 19: Timeline controls

Starting the time animates the map of the visualization while stopping the time freezes the map of the visualization. When the time progresses, the button turns into a *pause* button; when the time is stopped, the button turns into a *play* button. Also, when progressing, the time (and the slider) can be speed up or slow down by means of a dedicated button. Once the slider reaches the end of the timeline, the slider loops and goes back to the beginning. The time starts again.

Besides controlling the time, the timeline component displays a time series graph that shows the number of edits made at a given time in the form of line charts. By default, the timeline shows the number of edits made over time by the whole Reddit community that participated in the /r/place event. Depending on the interactions between the communities sidebar, that very same timeline can also show the number of edits made by multiple communities. The line charts of those communities are stacked and colored appropriately to distinguish them.

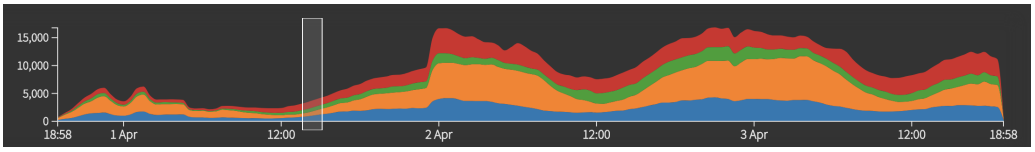


Figure 20: Timeline with stacked line charts

The timeline is characterized by a last feature which is the timeline window. One can indeed increase or decrease the width of the slider, the window in other words. Tinkering with this window does not have any impact on what the timeline displays strictly speaking. It is actually used by other components that compose the visualization.

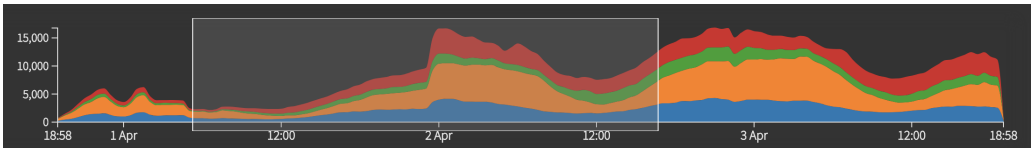


Figure 21: Timeline with time window

Evaluation

Data insights

The main question in the beginning of the project was the feasibility of detecting communities. A supposedly useful dataset was available and known algorithms could fit the problem, but there were uncertainties about the results we would get, the relevance of them, and whether those results would have been worth showing something. In such a situation, we had no choice but to give it a try. Even with strong assumptions (pixels with same color as the last color means that the user is in the community), we have been able to obtain pretty good communities, showing interesting results.

While preparing the visualization tutorial, we have been able to see interesting examples :

- the french-german fight, as shown in the tutorial,

- the american flag, becoming completely black a bit before the end of the visualization, but returning to normal after an hour, showing a coordinate attack toward a community.

- a community which remains during the whole event, such as the Switzerland flag.

Future work

The visualization works pretty well and is not subject to known bugs. Even though there is a pretty heavy preliminary processing phase before the visualization is ready, the observed delays are acceptable. A user would not have to wait long before being able to interact with the visualization. However, even though it is smooth, the visualization performs really resource-intensive tasks which implies a high usage of CPU and memory.

A first area of improvement of the visualization would definitely be reducing the usage of the CPU and the memory because some hardware may not be able to run the visualization. That's actually the case for a few Apple devices like the iPhones and the iPads. Strangely, it works perfectly fine on Android devices.

Even though significant efforts were made to produce a result as accurate as possible on the map, it can still be improved. This would require working further on the algorithms that were developed and others that were used for the preprocessing phase.

The visualization tries to be as user-friendly as possible. To this end, an interactive tutorial was implemented to help users learn how to interact with the visualization. This being so, it could be improved further. Right now, each step of the tutorial requires users to explicitly click on the *Next* button to proceed further. The state machine that is used under the hood could automatically trigger a transition whenever a user achieves a certain task. Besides that, more videos could be produced and made available. Last but not least, the tutorial systematically starts when accessing the visualization. One could think of some mechanism to trigger the tutorial only on the first access and then skip it. Using cookies should be more than enough for to achieve this.

Bots detection was imagined as an interesting feature in the very beginning of the project. Priority was put on other features, though. None the less, such a feature could give more data insights. Integrating this on the visualization would consist in highlighting communities whose members are bots or even better, highlighting

individuals that are actually bots.

Finally, we could emphasize further collaboration during the /r/place event. The visualizazion focuses on collaboration intra-communities for now. One extension would be showing collaboration inter-communities.

Peer assessment

All of the team members agreed on the following answers :

Were they prepared during team meetings? We were almost always present during meetings, with something new to offer, and to be shown either to Benjamin Ricaud or to Prof. Benzi.

Did they contribute productively to the team discussion and work? We all took part in the decision process, for choosing the libraries as well as for preprocessing choices. We all participated in different parts of the project, with each team member being more specialized : - Damien being specialized in Vue.js, preprocessing and the timeline, - Marc being specialized in 3D components, preprocessing and the tutorial, - Yassin being specialized in the timeline, preprocessing and visualization setup.

Did they encourage others to contribute their ideas? We were pretty much open to ideas, of course within the team, but most of all to the teaching team, to Benjamin Ricaud and to friends, as most of the time they were able to give good ideas.

Were they flexible when disagreements occurred? We did disagree on preprocessing choices, as we sometimes did strong assumptions. We did also disagree on the 3D rendering model of peaks. But with an external point of view, we were always able to make a choice, and never complained again later.

Acknowledgment

We would like to pay special tribute to Benjamin Ricaud for his invaluable help during the project. The precious and pertinent ideas that he gave us contributed significantly to make the visualization better and better over the weeks, which allowed us to reach such a result.

We would like to thank the teaching team, Prof. Benzi, for this great project, great feedbacks and hints, which were very valuable.