Undergraduate Thesis, November, 2018

ICEG- Instituto de Ciências Exatas e Geociências
Ciência da Computação

ORIGINAL PAPER

Using Kepler.gl to visualize weather data

Paulo Ricardo Koch*, Carlos Amaral Hölbig† and Rafael Rieder‡[1]

[1]University of Passo Fundo, BSc in Computer Science

*Undergraduate student: 117699@upf.br
†Co-advisor: holbig@upf.br
‡Advisor: rieder@upf.br

Abstract

Observing the global climate is considered as the great challenge facing humankind in the 21st century. With the increase in the number of weather stations, a big data is being stored on a daily basis. This data can be applied for effective analysis of weather and catastrophe forecasting. Correlations of different variables about processes and phenomena can show interesting climate behaviors, helping predictions. We understand that the visualization is a key point on weather research, however, the works that have been done in the area demanded a big structure and team to analyze the data and build the visualizations. Our goal is to present a data visualization application using Kepler.gl library, adding other tools as Mapbox, R scripts, and GeoJson. A case study was developed to show how to use our solution considering the use of temperature, humidity, precipitation, and solar radiation.

Keywords: Analysis tool; INPE; Kepler.gl; Visualization; Weather forecasting.

1 Introduction

Accurate weather predictions are important for planning our day-to-day activities. Considering the latest technological updates, the capabilities of retrieving and storing have increased; resulting in the availability of massive meteorology data in different formats. This data is generated both from the surface observation stations and aerial study stations. With the increase in the number of weather stations, a huge amount of data is available and grow up exponentially. This data can be applied for effective analysis of weather and catastrophe forecasting, helping people in decision making (Krishna; 2015).

Considering the computing advances, the static and dynamic meteorological data generated by geoscience and climate researchers has been steadily increasing. These data are often in a multidimensional form and multivariate volumetric datasets; thus, visualizing of these data is a computational and data-intensive task. When underlying properties are to be identified from simulations and observational data, visualization is the principal method for evaluating the reliability of conclusions based on these datasets (Lui et al.; 2014).

It is important identifying correlations within variables, because of the changes to our climate system. For improved insight into these complex spatial datasets that are the results of the simulations, 3D visualization is an appropriate instrument. In contrast to 2D visualization, where just a few variables can be visualized combined, 3D visualizationS allow evaluation and analysis of big and heterogeneous datasets. For example, correlations between variables like wind vectors, humidity and cloud coverage can be made visible (Helbig et al.; 2014).

Web mapping has become a popular way of distributing online maps through the Internet. Multiple services, like the popular Google Maps or Microsoft Bing Maps, allow users to visualize cartography using a simple Web browser and an Internet connection. However, geographic information is an expensive resource, and for this reason standardization is needed to promote its availability and reuse. In order to standardize this kind of map services, the Open Geospatial Consortium (OGC) developed the Web Map Service (WMS) recommendation. This standard provides a simple HTTP interface for requesting geo-referenced

map images from one or more distributed geospatial databases. It was designed for custom maps rendering, enabling clients to request exactly the desired map image. In this way, clients can request arbitrary sized map images to the server, superposing multiple layers, covering an arbitrary geographic bounding box, in any supported coordinate reference system or even applying specific styles and background colors (García et al.; 2012).

In this paper, we present an application to visualize weather data using the Kepler.gl, an open source geospatial analysis tool for large-scale datasets. The data was provided from INPE (National Institute for Space Research) and consisted of the forecast for the next eleven days, 24 readings each day. It has temperature, precipitation, solar radiation, and humidity information. With Kepler.gl, we created an application for visualizations that allow to compare and correlate some variables of interest, for example, humidity X temperature X precipitation; and temperature X solar radiation.

This paper is structured in five sections. Section 2 presents a few related works on visualization of weather phenomena. Section 3, shows the materials and methods used in the development of the study. Section 4 describes a case study where we find the appropriate hours to pulverize an apple orchard showcasing the platform. Finally, we conclude and discuss future work in the Section 5.

2 Related work

Helbig et al. (2015) built a flexible workflow that combines different state-of-the-art visualization software components in order to hide the complexity of 3D data visualization tools from the end user. They developed a lightweight custom application (MEVA - multifaceted environmental data visualization application), in order to complete the workflow and to enable the domain scientists to easily analyze their data. This workflow combines a variety of different data abstraction methods provided a 3D visualization and interactive application.

With domain expert they specified a set of requirements (Figure 1), a basis for the development of the workflow and the application.

Figure 2 shows a set of filters, where the user can show or hide variables, change their opacity and add outlines do the 3D objects. Time interface (Figure 3) provides controls such as play, pause, one step forward/back, go to first/last time step. In addition, user can adjust the speed of the animation.

Figure 4 details the workflow leading form input data to visualization. To evaluate the final version of MEVA a user study was conducted at TESSIN (Terrestrial Environmental System Simulation and Integration Network) VisLab. Figure 5 shows the presentation of the visualization. Authors concluded that the participants from various scientific domains were able to learn using the application very quickly. Even participants with little know-how in meteorology and no experience with 3D environments were enabled to explore the data. Most of the participants could learn something new with the help of the visualization and showed interest to use MEVA.

Helbig et al. (2014) present a concept and workflow that describes the way from raw input data of simulations and observation to a meaningful visualization, easy-to-understand and well-arranged. Simulations were selected from two different projects focusing on different scientific questions for exploring the possibilities of scientific 3D visualization of atmospheric data. The major goal was to give better insight into complex heterogeneous datasets and the correlations between the included variables.

Authors concluded that with the help of visualization, processes and phenomena can be uncovered and correlations between variables can be detected by interacting with data in a 3D environment. Visualization in the basis for analyzing and presenting weather and climate modeling results and gives a better insight into complex heterogeneous data sets. Visualization of atmospheric data is a domain that needs a close cooperation between experts of visualization and meteorology (Figure 6).

Murata et al. (2016) have built a real time 3D visualization system for the PAWR (Phased Array Weather Radar Data) data in conjunction with ecosystems provided by a cloud service. They visualize the full-resolution data in time and space within one minute after the observation in every 30 seconds.

One of the most important objectives in this study was to visualize the full-resolution PAWR data without data reduction.

Authors developed a low cost system to combine multiple ecosystems on a cloud, including a high-speed data transfer technique, a real-time data processing technique, a light-weight 3D visualization technique and a large scale distributed file system (Figure 7). Combining these ecosystems provided by the NICT (National Institute of Information and Communications Technology) Science Cloud, they succeeded in developing a real-time 3D visualization from 54 seconds to 69 seconds after every observation period in the NICT Science Cloud.

Lui et al. (2014) proposed a systematic meteorological data visualization framework on a virtual globe, which extends the emerging virtual globe platform to include the simulation and visualization of climate data for studies.

They also described a volume rendering algorithm on the GPU, which is widely used in climate applications. Authors also presented a virtual globe application of a tropical cyclone to demonstrate that virtual globes can be an effective tool for meteorological data visualization.

The approach described can be extended to other climate applications not only for visualization but also as an analysis and decision-making support tool/system. Lui et al. (2014) think we can use their enhanced globe to address the problems facing humanity as a result of climate change and natural disasters.

Considering these related work, we could understand that visualization is one important key point on weather forecasting research. Using three-dimensional or n-dimensional visualization resources could improve insights into the climate data enabling to explore different variables at the

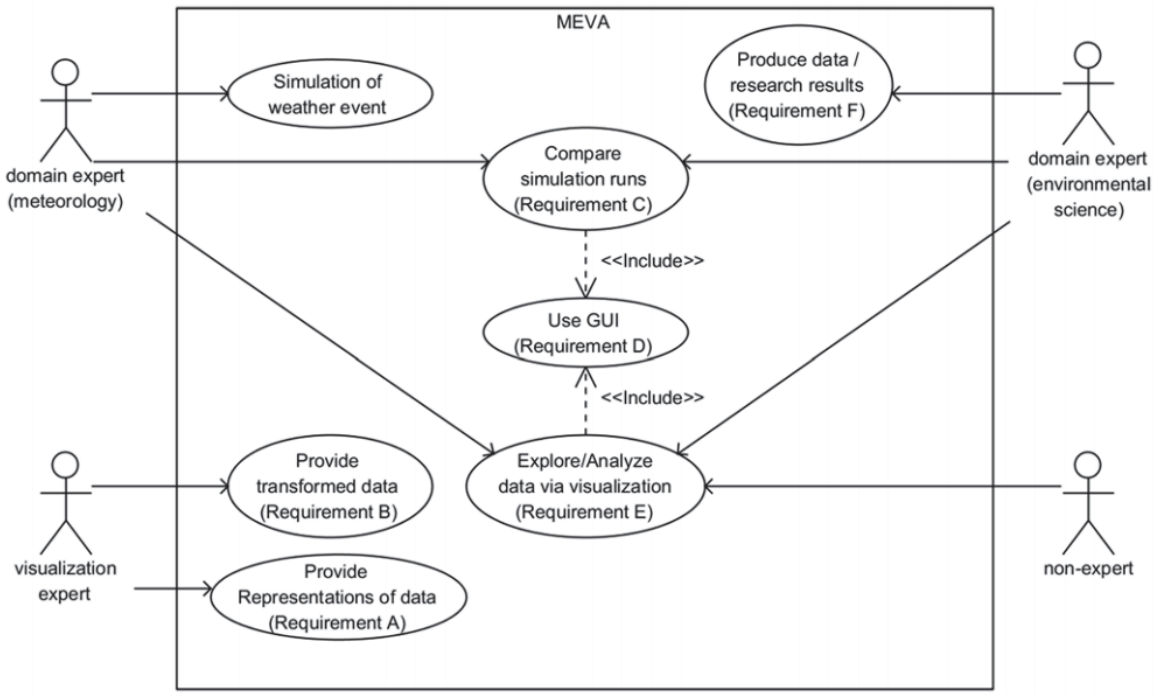

Figure 1: Basis requirements for the development of the workflow and the application (Helbig et al.; 2015).

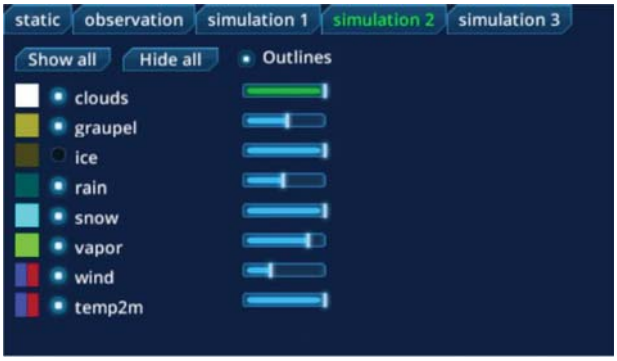

Figure 2: Set of filters, where the user can show or hide variables, change their opacity and add outlines do the 3D objects (Helbig et al.; 2015).

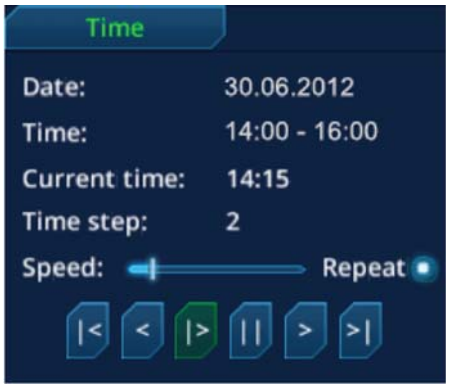

Figure 3: Time interface provides controls such as play, pause, one step forward/back, go to first/last time step (Helbig et al.; 2015).

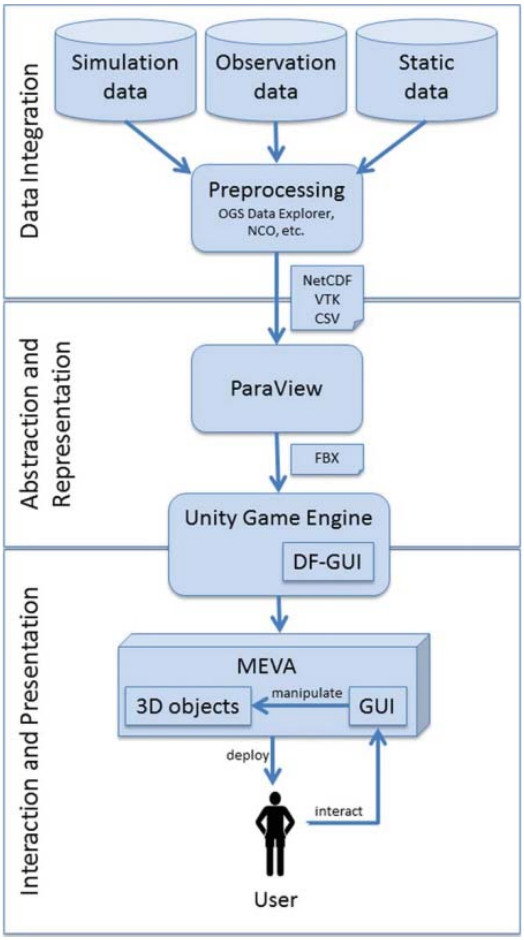

Figure 4: The workflow leading form input data to visualization (Helbig et al.; 2015).

Figure 5: To evaluate the final version of MEVA a user study was conducted at TESSIN (Terrestrial Environmental System Simulation and Integration Network) VisLab (Helbig et al.; 2015).

Figure 6: Visualization of atmospheric data is a domain that needs a close cooperation between experts of visualization and meteorology Helbig et al. (2015).

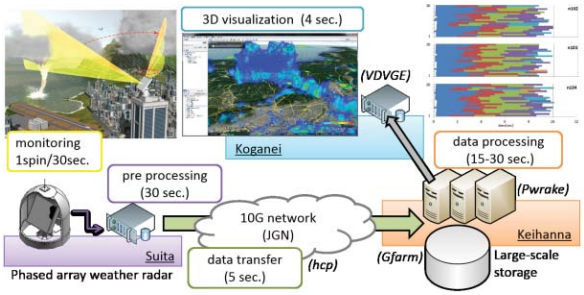

Figure 7: Authors developed a low cost system to combine multiple ecosystems on a cloud, including a high-speed data transfer technique, a real-time data processing technique, a light-weight 3D visualization technique and a large scale distributed file system (Murata et al.; 2016).

same time. With this in mind, our goal is to build a simple and user-friendly application offering this kind of resource to visualize weather data, using different web tools and supporting reuse.

3 Materials and methods

In order to develop our approach, we selected some tools to create an Kepler.gl application to visualize weather data. Next sections present this resources.

3.1 Mapbox

Mapbox is a large provider of custom online maps for websites and applications such as Foursquare, Lonely Planet, Facebook, the Financial Times, The Weather Channel and Snapchat. Since 2010, it has rapidly expanded the niche of custom maps, as a response to the limited choice offered by map providers such as Google Maps. Mapbox contributes significantly to some open source mapping libraries and applications, including the MBTiles specification, the TileMill cartography IDE, the Leaflet JavaScript library, and the CartoCSS map styling language and parser.

The data are taken both from open data sources, such as OpenStreetMap and NASA, and from proprietary data sources, such as DigitalGlobe. The technology is based on Node.js, Mapnik, GDAL, and Leaflet.

Mapbox uses data from tracks of its clients' users, such as Strava and RunKeeper, to identify missing data in OpenStreetMap using automatic methods (Mapbox; 2018).

3.2 React (JavaScript library)

React (also known as React.js or ReactJS) is a JavaScript library for building user interfaces. It is maintained by Facebook and a community of individual developers and companies. It can be used as the base in the development of single-page or mobile applications (React; 2018).

3.3 WebGL

WebGL (Web Graphics Library) is a Javascript API for rendering interactive 2D and 3D graphics within any compatible web browser without the use of plug-ins. WebGL is fully integrated with other web standards, allowing GPU-accelarated usage of physics and image processing and effects as part of the web page canvas (Khronos; 2018).

3.4 Kepler.gl

Kepler.gl is a open source geospatial analysis tool for large-scale datasets. Built with Deck.gl, it uses WebGL to render datasets quickly and efficiently.

This tool allows to drag and drop a dataset (this data is written with the geojson standardization), add filters, apply scales, and do aggregation on the fly. Because it is built on React and Redux, Kepler.gl can be embedded inside your own mapping applications (Uber; 2018).

The tool is one of the Uber's visualization frameworks, and it was built to make data-driven maps that allow us to get insights from location and deliver business/scientific outcomes.

Kepler.gl allows the data scientist to see multiple aspects of the weather in the same visualization. It enables him to get better results from the analysis and deliver better conclusions to the community.

3.5 R

R is a language and environment for statistical computing and graphics. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues. R can be considered as a different implementation of S. There are some important differences, but much code written for S runs unaltered under R.

R is available as Free Software under the terms of the Free Software Foundation's GNU General Public License in source code form. It compiles and runs on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux), Windows and MacOS (Foundation; 2018).

3.6 JSON

JSON is a text syntax that facilitates structured data interchange between all programming languages. It is a syntax of braces, brackets, colons, and commas that is useful in many contexts, profiles, and applications. JSON stands for JavaScript Object Notation and was inspired by the object literals of JavaScript aka ECMAScript as defined in the ECMAScript Language Specification, Third Edition.

It is an open-standard file format that uses human-readable text to transit data objects consisting of attribute-value pairs and array data types (or any serializable value). It is a very common data format used for asynchronous browser-server communication, including as a replacement for XML in some AJAX-style systems. The official Internet media type for JSON is application/json and uses the file extension (.json) (International; 2017).

3.7 GeoJSON

GeoJSON is a format for encoding a variety of geographic data structures using JavaScript Object Notation (JSON) [RFC7159]. It is widely used in JavaScript web-mapping libraries, JSON-based document databases, and web APIs (Force; 2016).

A GeoJSON object may represent a region of space (a Geometry), a spatially bounded entity (a Feature), or a list of Features (a FeatureCollection). It supports these geometry types: Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon, and GeometryCollection. Features in GeoJSON contain a Geometry object and additional properties, and a FeatureCollection contains a list of Features.

The format is concerned with geographic data in the broadest sense; anything with qualities that are bounded in geographical space might be a Feature whether or not it is a physical structure. GeoJSON concepts are derived from open geographic information system standards and have been streamlined to better suit web development.

3.8 Tool integration

Our approach use Kepler.gl as main tool to present and filter data resources. Datasets can be displayed in 3D using time-lapse visualizations, in order to explore more details from the information.

Figure 8 shows a sample of our application, presenting a distribution of some climate variables at a given time, in a given region. On the left, temperature and solar radiation; on the right, humidity and precipitation. At the bottom is the time-lapse controller, so we can see the data changes throughout the period of analysis. The legend is on the top right of each map.

The data provided was in a format (.Rda) designed to use with R. This file stores a complete R work-space or selected "objects" from a work-space in a form that can be loaded back by R. In order to use the data with Kepler.gl, an R script was created to extract the information into a geojson file (control flow represented by Figure 9). Each Rda file had one weather variable, the geojson had all information (precipitation, wind, humidity, solar radiation and temperature) grouped in a FeatureCollection class.

Considering 0.15 degrees latitudinal and longitudinal interval, observational weather data is taken each hour. The data provided had all the South America readings. A R script was used to filter Brazil data, by whole set, groups of States or a single State.

Our geojson file has a collection of features. Each feature has the following properties: temperature, humidity, precipitation, wind, solar radiation, latitude, longitude, reading time, index; and the geometry: (type: Point, coordinates: [latitude, longitude]).

To render the map, Kepler.gl makes use of the Mapbox mapping packages: mapbox/geo-viewport, mapbox/geojson-extenmapbox/geojson-normalize. So we had to create an account and set the MapboxAccessToken in our system.

4 Case study

The data used in this case study was provided by Brazilian National Institute for Space Research (INPE). This dataset has the South America weather information considering the period between November 14th and November 22nd of 2018. The data was stored in .Rda files as follows: OCIS.Rda had solar radiation readings, PREC.Rda precipitation,

Figure 8: On the left, temperature and solar radiation; on the right, humidity and precipitation. At the bottom is the time-lapse controller, so we can see the data changes throughout the period of analysis. The legend is on the top right of each map.

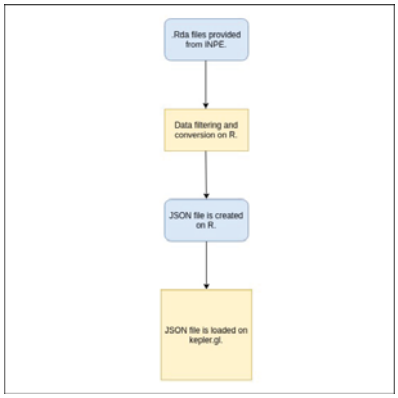

Figure 9: The data provided is filtered and exported on R to be loaded on Kepler.gl.

TP2M.Rda temperature at two meters height, and UR2M.Rda humidity at two meters from the ground.

We have developed a script in R to extract the data into a geojson file with the data Kepler.gl needs to plot the visualizations on the map. The file consists of a JSON object with the structure presented on Figure 10. For every 0.15 degrees of latitude and longitude there is a point to get and plot in a map. Figure 11, Figure 12 and Figure 13 show, respectively, a tabular view of reading points, the first loading of these points, and the region of interest after filtering.

Using R we can filter the data from Brazil or a specific state or city. After the data conversion, it can be imported on Kepler.gl. The solution allows to create layers, filters, and setup the map style on the platform, and choose between a few layer types (Figure 14). In the case study, we opted by grid because it has the same shape that the readings were collected. For each layer, we can choose the colors used in correlation with some attribute. It is possible to set the variable that will define the grid elevation.

The view can be split. For example, we create a layer for the temperature and solar radiation using

```
{
   "datasets": [{
      "version": "v1",
      "data": {
        "id": "idRS",
        "label": "RS.geojson",
        "color": [...],
        "allData": [
          [
            {
              "type": "Feature",
              "properties": {
                "temperature": 22.4458,
                "humidity": "88.86",
                ...
              },
              "geometry": {
                "type": "Point",
                "coordinates": [
                  -53.45,
                  -33.55
                ]
              }
            }
          ]
        ]
   }],
   "config": {
          "version": "v1",
          "config": {
            "visState": {
              "filters": [
                {
                  ...
                }
              ],
              "layers": [
                {
                  ...
                }
              ],
              "interactionConfig": {
                ...
              },
              "layerBlending": "normal",
              "splitMaps": []
            },
            "mapState": {
              "bearing": 24,
              "dragRotate": true,
              "latitude": -31.0931,
              "longitude": -54.9953,
              ...
            },
            "mapStyle": {
              ...
            }
          }
        },
        "info": {
          "app": "Kepler.gl",
          "created_at": "Sat Nov 17 19:37:09 2018"
        }
}
```

Figure 10: JSON object to load on Kepler.gl. It has datasets, layers, filters and configurations that will be loaded on the platform.

this resource (Figure 15). The temperature was the base color and solar radiation as elevation. The other

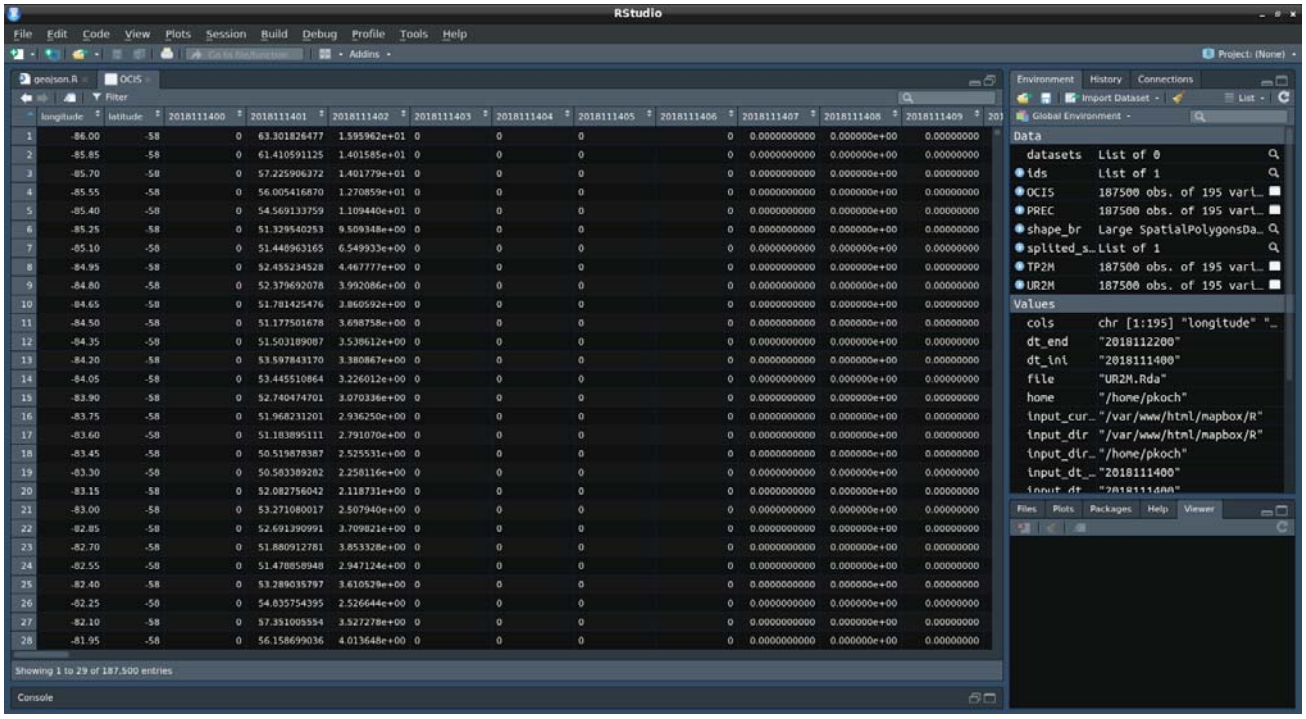

Figure 11: Original data on R. Each row contains latitude, longitude and every hour readings of solar radiation.

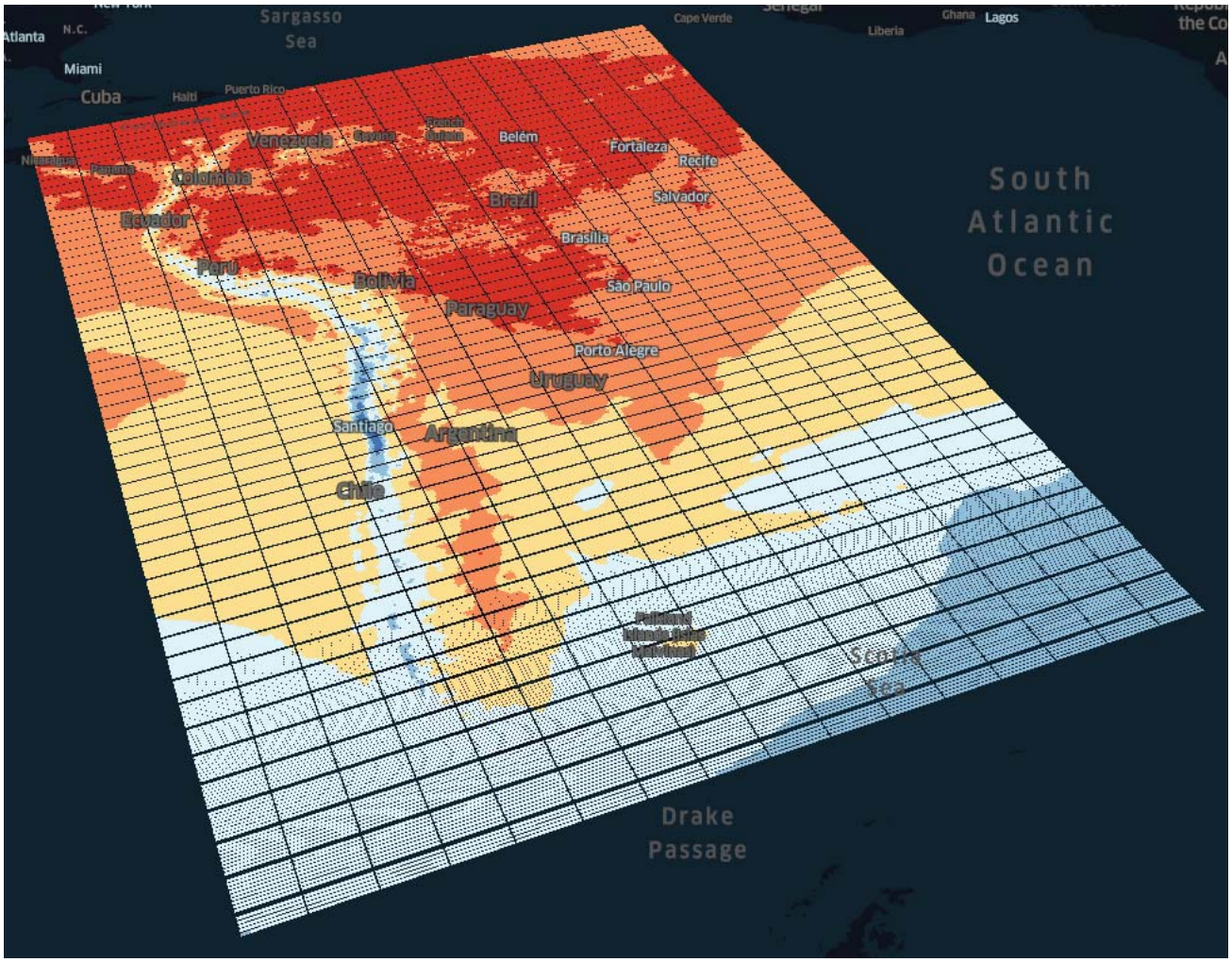

Figure 12: Temperature without data filtering.

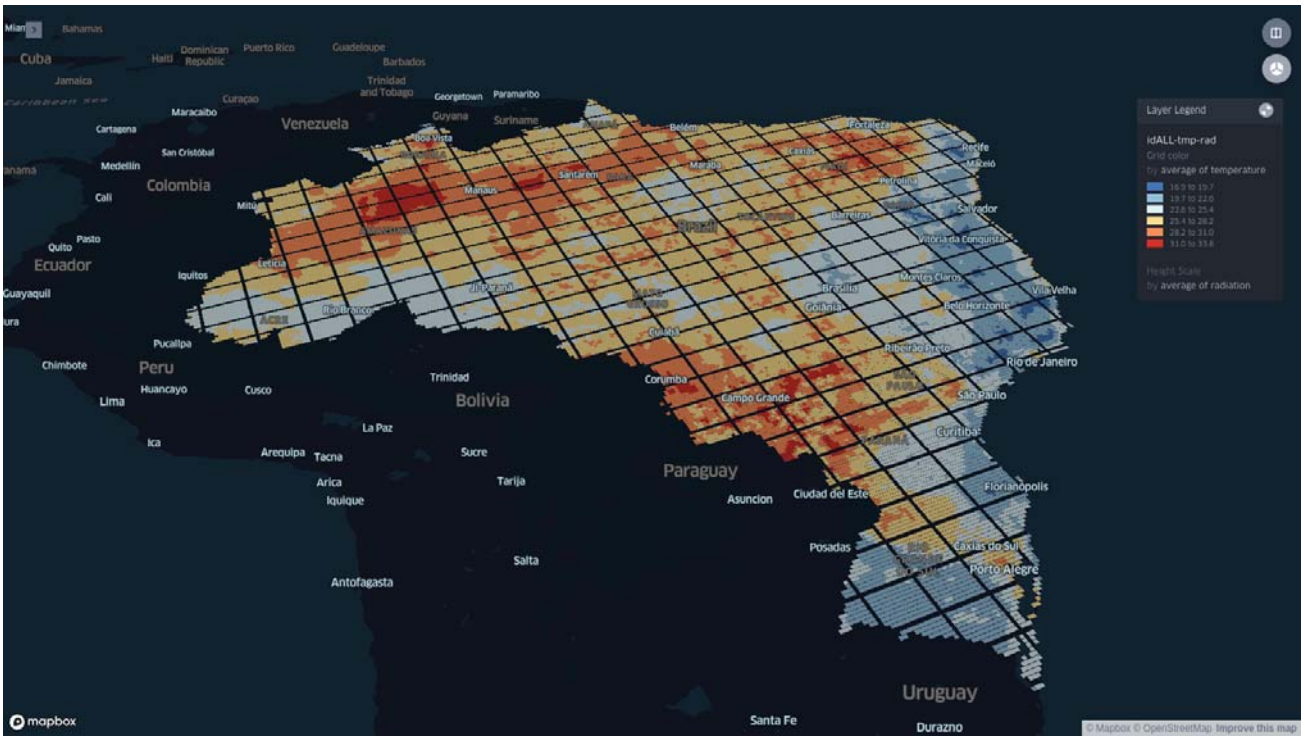

Figure 13: Temperature after data filtering. Kepler.gl performance is better as the volume of data decreases.

layer has the humidity information the base color and precipitation as elevation.

Every hour the information is read. On Kepler.gl we can filter the time of the readings and create a time-lapse for the visualization (Figure 16). Using filters it is possible to search for any others variables such as temperature and precipitation. As we filter the data the visualization is updated. We can also open a table to access the current information.

On the time-lapse controller, there is a button to play/pause the visualization and other to restart the animation. We can control the speed of the time between 0.5x, 1x, 2x and 4x, and choose the reference value to plot on the Y-axis. The Y value enables us to see the changes in the chosen variable throughout the period of the scene. We can configure the map style (base color, font color etc.) and the map attributes that will be seen such as Road, Land, Water, Building and Border (Figure 17).

In our case, we were looking for temperatures and humidity that were ideal to pulverize an apple orchard. The temperature had to be below 30 degrees Celsius, the humidity above 55% and it couldn't be raining (Figure 18). With this in mind, we extracted the data of a specific city considering a period between November 15th and November 18th. We create a file for each day because it is easier to see the temperature on the Y-axis of the time-lapse controller. Using our Kepler.gl application, we filtered the information as described before.

Running the time-lapse we could find the periods where it was ideal to apply the pesticides. We were able to show that the best time to apply the pesticides is very early in the mornings when it is not raining because in the afternoon the humidity starts to decrease as the solar radiation goes up (Figure 19).

5 Conclusions

In this paper, we develop a n-dimensional application to explore correlations between variables in the weather phenomena, seeing multiple information at the same time, offering a better insight of the data provided. Mapbox and Kepler.gl enabled us to develop visualizations to understand the climate data and weather forecasting. Importing JSON files is useful because it enable to work and extract data with our tool and others such as R, loading results into the web browser and present it.

The configuration needed to run the tool is little, and the interface is intuitive and easy to use. However as the volume of data increases the application starts to show some performance issues.

As a future work we could improve the data processing and filtering on the platform. It would be interesting to add the option to visualize the data with a VR headset, and integrate this solution in a weather forecasting application.

Acknowledgements

We thank Uber for developing and sharing their frameworks under the MIT License, so the community can use and improve it.

References

Force, I. E. T. (2016). The geojson format. Accessed: 2018-11-15, https://tools.ietf.org/html/rfc7946.

Foundation, T. R. (2018). Introduction to r. Accessed: 2018-11-15, https://www.r-project.org/about.html.

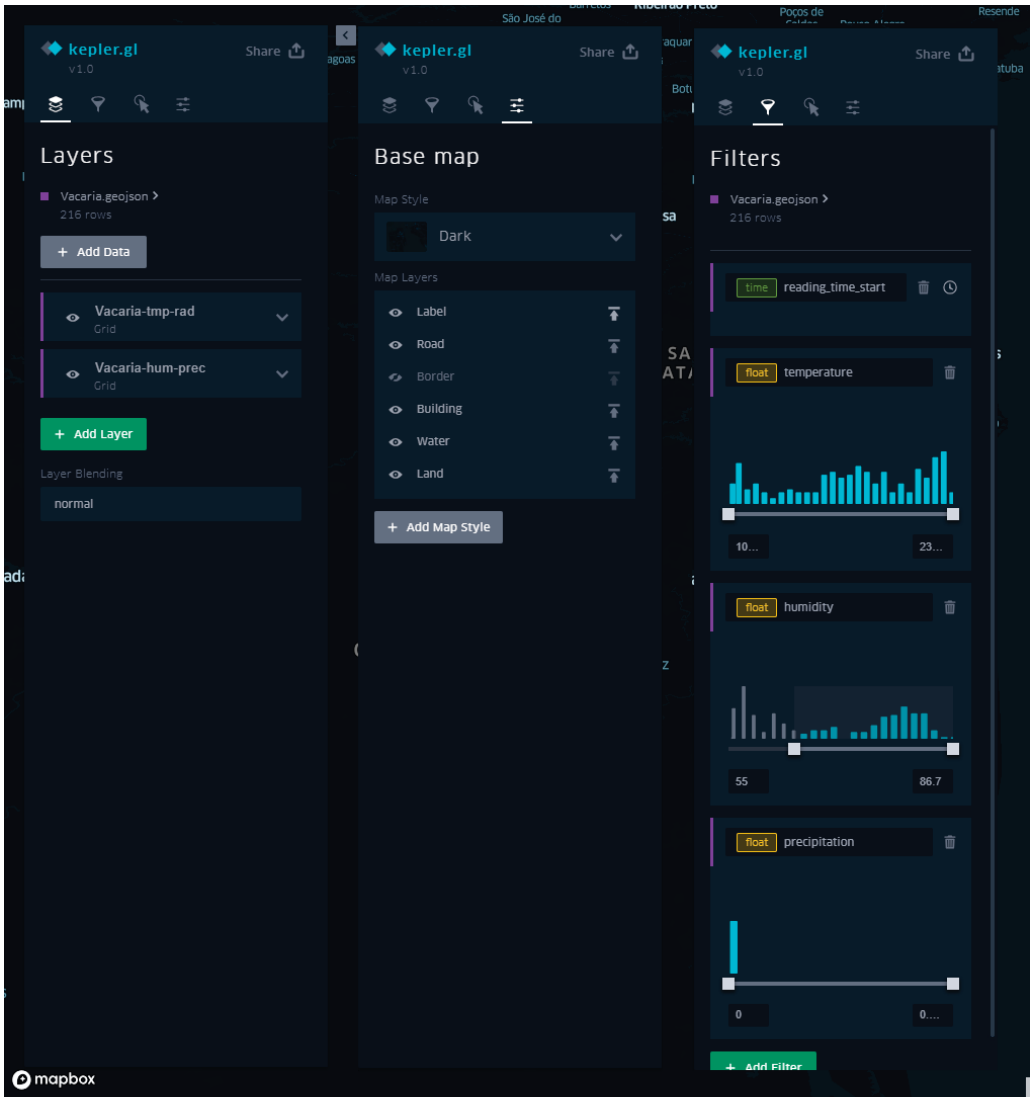

Figure 14: This picture shows the Layers, Filters and Base map panels.

García, R., de Castro, J. P., Verdú, E., Verdú, M. J. and Regueras, L. M. (2012). Web map tile services for spatial data infrastructures: Management and optimization, *IntechOpen* p. 1. http://dx.doi.org/10.5772/46129.

Helbig, C., Bauer, H.-S., Rink, K., Wulfmeyer, V., Frank, M. and Kolditz, O. (2014). Concept and workflow for 3d visualization of atmospheric data in a virtual reality environment for analytical approaches, *Environmental Earth Sciences* pp. 1–2.

Helbig, C., Bilke, L., Bauer, H. S., Bottinger, M. and Kolditz, O. (2015). Meva – an interactive visualization application for validation of multifaceted meteorological data with multiple 3d devices, *PLoS ONE* .

International, E. (2017). The json data interchange syntax. Accessed: 2018-11-15, http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf.

Khronos (2018). Webgl. Accessed: 2018-11-15, https://www.khronos.org/webgl/.

Krishna, G. (2015). An integrated approach for weather forecasting based on data mining and forecasting analysis, *International Journal of Computer Applications* **120**(11): 1.

Lui, P., Gong, J. and Yu, M. (2014). Visualizing and analyzing dynamic meteorological data with virtual globes: A case study of tropical cyclones, *Environmental Modelling and Software* p. 1. https://www.sciencedirect.com/science/article/pii/S1364815214003405.

Mapbox (2018). Mapbox. Accessed: 2018-11-15, https://mapbox.com/.

Murata, K. T., Muranaga, K., Yamamoto, K., Nagaya, Y., Pavarangkoon, P., Satoh, Mizuhara, T., Kimura, E., Tatebe, O., Tanaka, M. and Kawahara, S. (2016). Real-time 3d visualization of phased array weather radar data via concurrent processing in science cloud, *IEEE* .

React (2018). React. Accessed: 2018-11-15, https://reactjs.org/.

Uber (2018). Kepler.gl. Accessed: 2018-11-15, http://kepler.gl/.

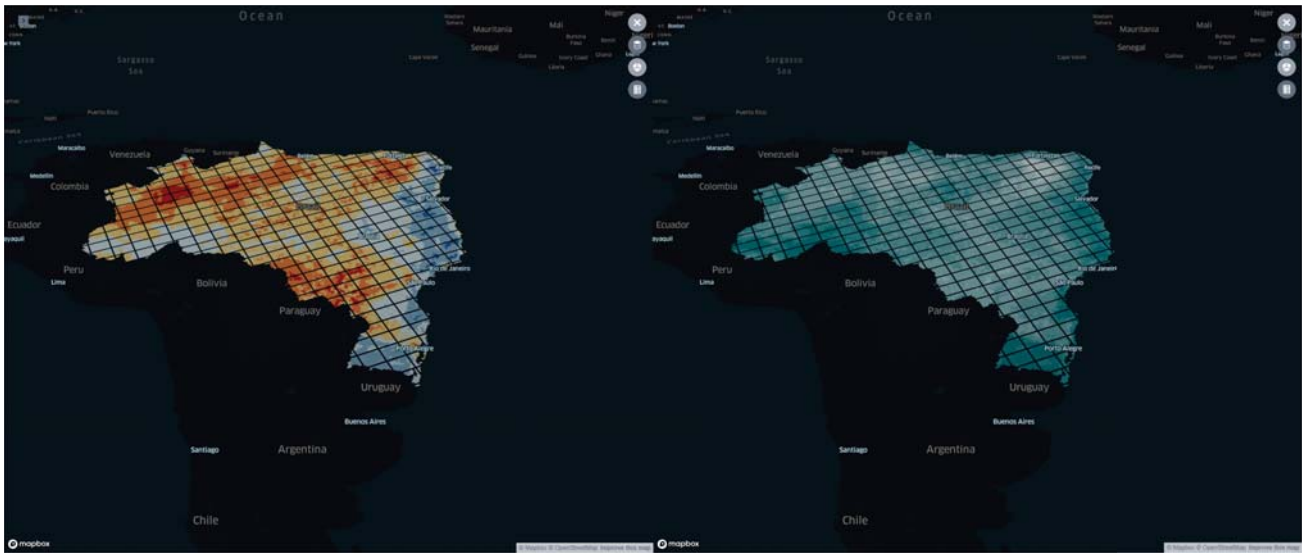

Figure 15: The temperature and solar radiation on the left, humidity and precipitation on the right.

Figure 16: The time-lapse controller with the humidity information on the Y-axis.

Figure 17: Panel to configure the layers.

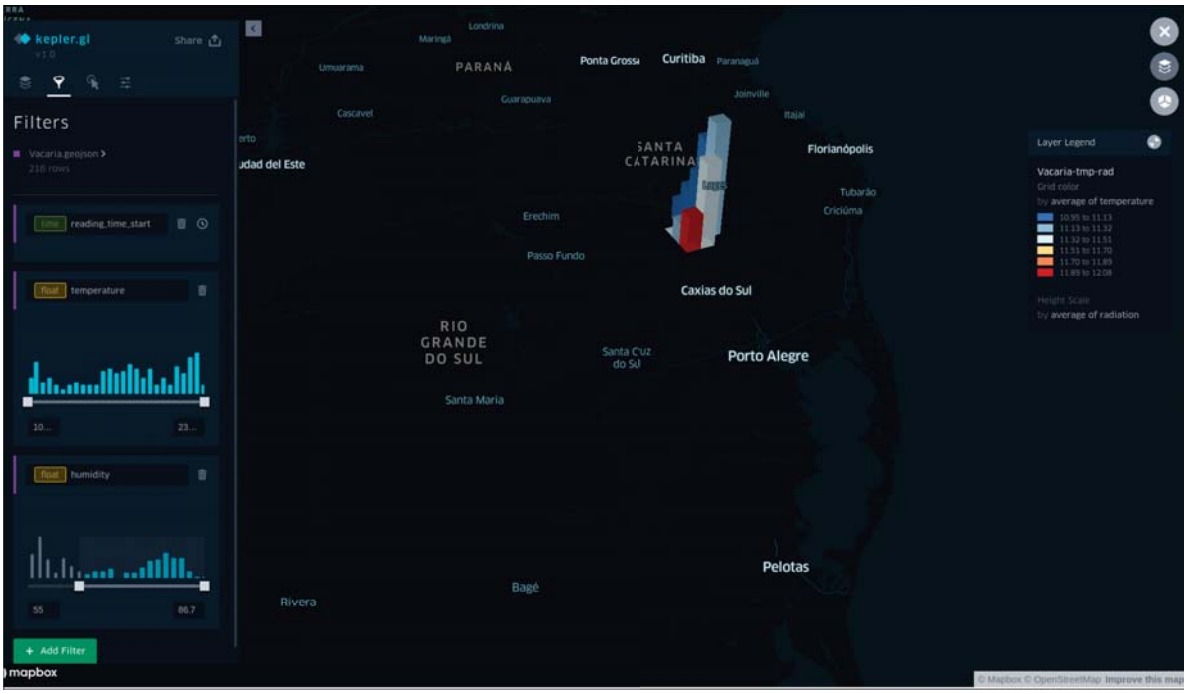

Figure 18: Temperature and humidity filters used on our case study. Temperature between 10 and 23 degrees Celsius and humidity above 55%.

Figure 19: The humidity goes down as solar radiation increases.