BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2021
http://excel.fit.vutbr.cz

Authoring System for Defining Geospatial Objects

Andrej Tlčina

Geovisto Leaflet TURF

Abstract
The goal of this project is to create an authoring system that will help users to create thematic maps with custom geospatial objects and custom datasets. This paper presents an authoring system that provides tools for creating simple and complex geospatial objects. The system allows users to import their datasets, containing custom data, like data about sales, wages, population, to mention few. Upon creating the needed geospatial object, the user then can use the mentioned imported data and map certain information onto the object. The project can help people like architects, cartographers, and academic users, who need thematic maps, so they can better communicate their ideas. For example, students can create thematic maps containing states that do not exist anymore. The project is an extension of the project Geovisto which uses predefined data for data visualization. This predefined data can be modified only by rewriting code. The extension will bring the possibility of creating custom graphical features or editing predefined ones, which can be used in other parts of the project. At the current stage of implementation, the system provides several tools for creating geospatial objects. It can import a dataset and apply an identifier which then maps the corresponding data onto the object. In addition, the user can export the state of the map and object and later import it.

Keywords: Authoring system — Geospatial objects — Web application

Supplementary Material:

*xtlcin00@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

According to [1] there is a lot of use for geospatial data in today's world. This type of data can be presented in the form of thematic maps which can emphasize the meaning of the data. Those data are related to geospatial objects, which represent natural or man-made features, such as lakes, roads, or buildings. They can become complex for visualization.

Nowadays, when individuals want to create thematic maps, there are two ways to go about it. The first one involves the use of an authoring system. Unfortunately, many of these applications have their limits, like the lack of complex shapes. The other way is to use the programming language and create the needed functionality themselves. Clearly, the second option has the requirement of having programming skills.

The solution that the paper introduces tries to find the so-called 'sweet spot'. It would offer a full-fledged application, so users do not have to create it by their selves, but also will offer functionality for thematic map creation. The geospatial objects creation would be done by using a prepared tools palette or by importing them in GeoJSON format, which is often used with geospatial data. Additionally, users can import a custom generic dataset, containing data in some relation with objects. This data can be expressed as a table with one of the columns containing an identifier. By the identifier, the system can map the data in the row of the selected identifier to the selected objects.

The solution contains a palette of multiple tools for the definition of geospatial objects, tools for their transformation, for the possibility of searching places, and finally, the tool for creating topologies.

2. Analysis

One of the main functionalities of the project is the definition of geospatial objects. Those can be used as components of various thematic maps (such as choropleth maps, connection maps, point maps, etc.). This section presents these components, as well as possible usage scenarios. Finally, the workflow of the usage is introduced.

2.1 Geospatial Data and Objects

Geospatial data is a type of data represented by values or facts delivering some kind of information. Geospatial data contain information about the location on Earth's surface [2]. It is usually stored in the form of a pair (longitude, latitude).

Most geospatial data that we want to visualize on the map can be represented by three types of graphical objects: points, lines, polygons [3] (examples of their usage are shown in figure 1). Geospatial objects can be labeled with an **identifier**. It is a unique label within a set of geospatial objects. Examples of such identifiers are ISO-3166 country codes [4].

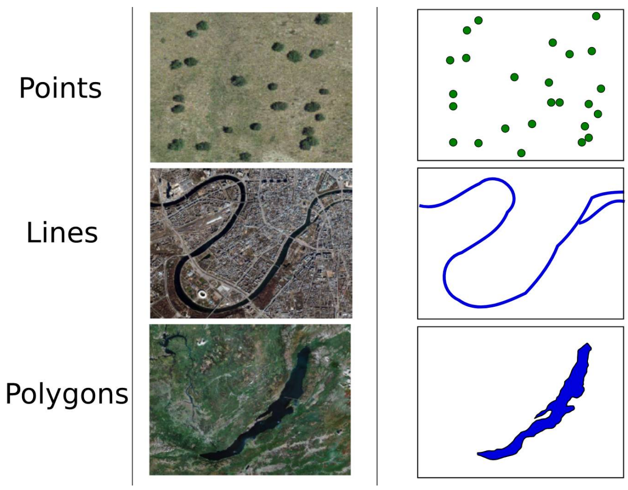

Figure 1. Examples of geographic features and geospatial objects that represent mentioned features (taken from book Introduction to GIS [5]).

2.2 GeoJSON

For encoding a variety of geospatial data structures, the format called GeoJSON is used [6]. This format can encode a variety of geospatial objects, which are called GeoJSON features. The main attributes of GeoJSON features are type, geometry (which also contains coordinates), and properties (e.g., color or stroke of a feature).

2.3 Usage Scenarios

In today's world, the user has the option of choosing from multiple applications or systems that can provide defining graphical objects on the map (section 3). Each of them is focused on different use cases. Hence, firstly, the target users and their requirements were determined and analyzed. Three types of personas were defined:

1. **Network analysts**: one of the groups that will potentially use this authoring system is represented by the IT workers specializing in networking. The reason being is the need for visualization of network traffic.
2. **Architects and cartographers**: primary interaction with the system will be to demarcate some region or territory. Architects might use tools for creating floor plans and demarcate areas in that way. On the other hand, there are also cartographers, who might demarcate areas like regions of the state. The major difference will be in the types of objects used to do so. Architects might use more geometric shapes as opposed to cartographers that might use more organic shapes for natural geographical features.
3. **Academic users**: a lot of services used for the creation of thematic maps are used for educational purposes. Teachers can use these tools for a better explanation of the topic. While students might use tools for the creation of thematic maps for school projects and presentations.

All of these types of users have one in common. They want to connect data with custom thematic maps using user-defined geospatial objects (points, lines, and polygons).

2.4 Workflow

Before defining any geospatial objects, users can optionally import the custom dataset and/or prepared geospatial objects in GeoJSON format, as the diagram of figure 2 shows. Either way, users can start adding and customizing objects. Customization involves changing the appearance of an object, but also applying identifiers. If the dataset was imported at the beginning, the system maps certain data to a geospatial object. Once the user is satisfied with the result, they have the option of exporting the created geospatial objects in GeoJSON format.

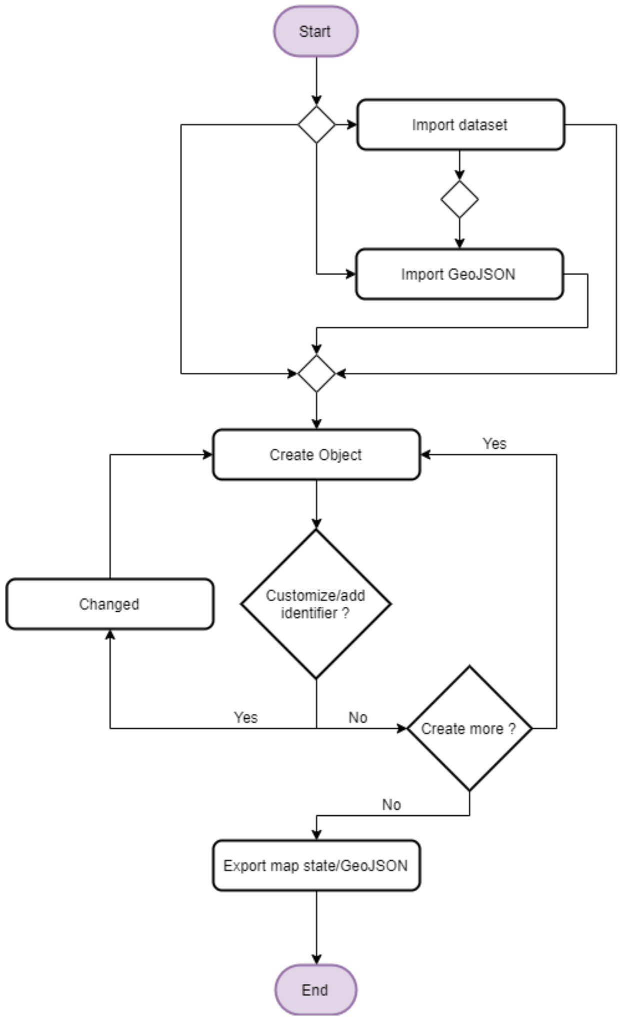

Figure 2. Diagram of the actions a user will take while working with application.

3. Existing Solutions

Before the design of the solution, existing tools were analyzed. The focus of the research was heavily aimed at web applications since the final product will be built with the help of web-based technologies. The solution created with web-based technologies allows users a simple usage of the system because they will need only a browser and the system will be easily accessible online. The solution's primary focus group is desktop users, mainly for real estate computer screen offers, as opposed to mobile ones. We looked at tools for defining geospatial objects, what kinds and options for customization of objects. Moreover, we looked at how and whether the applications can work with data (e.g., can import or export GeoJSON). Applications that were part of the research are Mapme[1], Scribble Maps[2], Smartdraw[3], geojson.io[4], and Click2Map[5].

3.1 Advantages of Existing Solutions

As it was said earlier, the main three geospatial objects are points, lines, polygons. This fact puts a lot of emphasis on the creation of named objects. All applications can handle this task, be it with slight differences. Regarding more complex objects, Scribble maps is in the lead. The application also nicely handles the customization of each object. For handling GeoJSON data, there is geojson.io, which comes with a unique way of working with said data. The interface is separated into 2 parts, there is a map with tools on the left, and on the right, there is an editor into which users can write GeoJSON data that is projected onto the map in real-time.

3.2 Disdvantages of Existing Solutions

As mentioned, all applications can handle the task of creating objects, but only simple ones. Regarding creating more complex objects or simple objects having relations amongst each other, these applications underperform, except for Scribble maps, which, on the other hand, lacks the export of GeoJSON format.

4. Design

This section introduces the key features of the solution. They were derived from market research and from the expected needs of the personas we defined earlier in subsection 2.3.

Looking at the market research, all researched applications had at least tools for creating points, lines, and polygons. Hence, these three features were included in this project. However, in contrast to the presented solution, we put emphasis on the simplicity of their usage and the possibility to create more advanced kinds of those objects:

- **Brush tool**: using simple polygons gets the job done, but a lot of times it can be tedious. For that reason, the project includes a tool that mimics a brush or a pen in a sense. With this tool, users can create strokes covering large areas that will have the same opacity all the way (fig. 4).
- **Joining/subtracting**: the goal of working with tools is recreating the feel of vector programs, where each tool can be used with another one and where each object can interact with another one. Geometric polygons and organic polygons,

[1] https://mapme.com/

[2] https://www.scribblemaps.com/

[3] https://www.smartdraw.com/

[4] http://geojson.io/

[5] https://www.click2map.com/

created with a 'brush' tool, can be united when one of them is selected. On the other hand, when defining a new polygon over a nonselected polygon, it is subtracted (fig. 5). This workflow aims at the simple and easy creation of different kinds of polygons that can represent natural or man-made features.

- **Knife tool**: for manipulation with polygons, there was also added a tool that allows to cut the polygons into multiple pieces.
- **Search feature**: is a feature for locating places and areas on the map. User can either type a place into a text input and the system sends a request to *OpenStreetMap* API[6] receives the response and creates a point in place from response. Or the user picks the country and an administration level and after submitting *Overpass* API[7] which will return data containing the area the user searched for.
- **Network prototypes**: the feature uses points in combination with lines, which allows to prototype networks (fig. 3). It can be used by network analysts for visualizing network traffic.

5. Solution

The application is written in Javascript. In terms of libraries, the project makes use of several, mainly *Leaflet* and *Turf*, both of which specialize in visualizing geospatial data and working with geospatial objects. One of the many advantages is that the Leaflet library has its community that creates all sorts of plugins, that are put to good use in the project.

5.1 Architecture

The solution is implemented as a module for a project called Geovisto [7]. The project Geovisto is basing the solution on a library Leaflet. Geovisto is a library divided into multiple modules. Each module visualizes results on a Leaflet map, where each object is represented by a data structure called *Layer* [8]. The mentioned structure will be used for storing created graphical features like point, line, and polygon. All of the geospatial objects of the map are being stored in *FeatureGroup* [8], which is another Leaflet data structure. *Layer* structure has plenty of needed information and also defines a useful method *toGeoJSON*, that returns an object in GeoJSON format. This allows for data serialization in GeoJSON format, which means taking all geospatial objects of the map and converting them into a particular format. Apart from GeoJSON format serialization, there is serialization in the internal format.

5.2 Features

- **Brush tool**: The main idea behind the 'brush' tool is to create multiple circles on the map that are unified. When the mouse button is clicked, the function tied to that event draws a circle. After moving the mouse and still holding down the mouse button, more circles are being created. Each new circle is unified with the previous one or a previously unified object. Union of these objects is possible with Turf library, which can easily unite two GeoJSON features. (fig. 4)
- **Joining/subtracting**: each brushstroke will subtract from the polygon underneath it. Because of this reason, each stroke has to be separated and an object with key-value pairs has to be used. In this implementation, the key is simply the number of a stroke and the value is the stroke itself. Each time a stroke is created, the system loops through each of the objects on the map and subtracts the stroke from them. (fig. 5)
- **Knife tool**: The 'knife' tool works in a very similar manner. First, the polygon which will be sliced has to be selected. That simply means giving each object a click event listener. Upon activating the listener, the function tied to it stores the clicked object. After selecting a polygon, the user can create a line of any shape over it. At that moment we have a line and polygon, but those cannot be subtracted. In that case, we modify the line by offsetting it and creating a very thin polygon from the original line and offset version, which is suitable for subtracting from the selected polygon. (fig. 6)
- **Search**: for searching places, users have to input the place they are looking for into the text field. This input is taken and used as a query parameter of request to *OpenStreetMap* API, which will send a response in return. The response will contain the coordinates of the searched place. Coordinates are taken and used in the creation of a marker with Leaflet functionality. While searching for areas, the user has to pick a country from a static list and the level of administration. With this information system creates query that will be sent as part of a request to *Overpass* API. Respond will contain OpenStreetMap data that is parsed and GeoJSON data is received. The GeoJSON data contains features which are

[6] https://nominatim.openstreetmap.org

[7] https://wiki.openstreetmap.org/wiki/Overpass_API

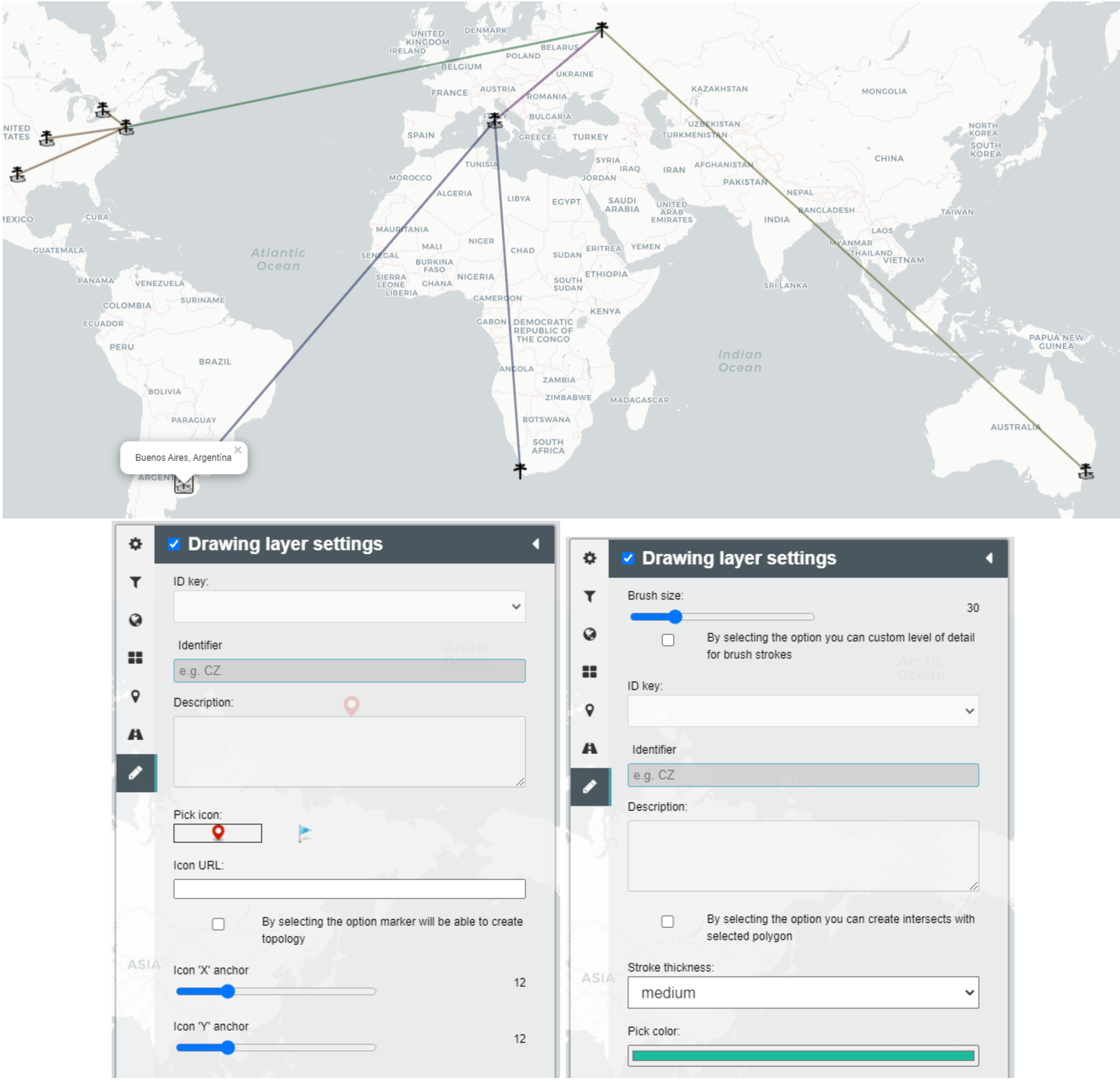

Figure 3. The first row of images displays an example of a topology. There is also a pop-up example, which displays name of the place a user search for. A sidebar with possible customization options for points is displayed on the left of the second row. The image next to it shows a sidebar with possible customization options for polygons and lines.

represented as leaflet structures and appended to the map. (fig. 7)

- **Network prototypes**: the feature uses markers that are given a special attribute. When the marker with the special attribute is created, a particular function runs that will loop through each of the markers with the special attribute, in the order of creation. In each iteration of the loop, there are two markers taken, one of the current iteration and one of the next iteration. The function takes their coordinates and creates a line between them.

5.3 Generic data mapping

An important functionality of the project is taking custom datasets and mapping data onto geospatial objects via applying an identifier to them. Data, that is to be uploaded, has to be in structured JSON format. The issue here is that the system works with generic data, so it does not know what it will get and by which identifier it can map the data. That means first and foremost, the user has to pick the name or header of the column from which the system can pick the identifiers. Then the user can select an identifier. Upon selection, the function for finding a JSON object with a proper identifier value is invoked. JSON object will

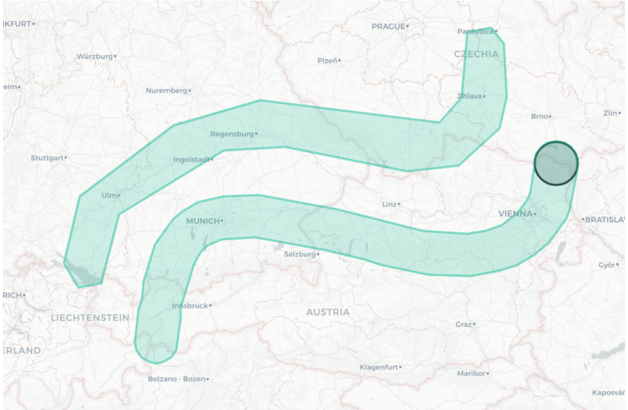

Figure 4. Example of brush strokes. The tool allows modifying so-called tolerance of the created polygon, which means that the user can control the smoothness of it. The lower the tolerance, the smoother the polygon is.

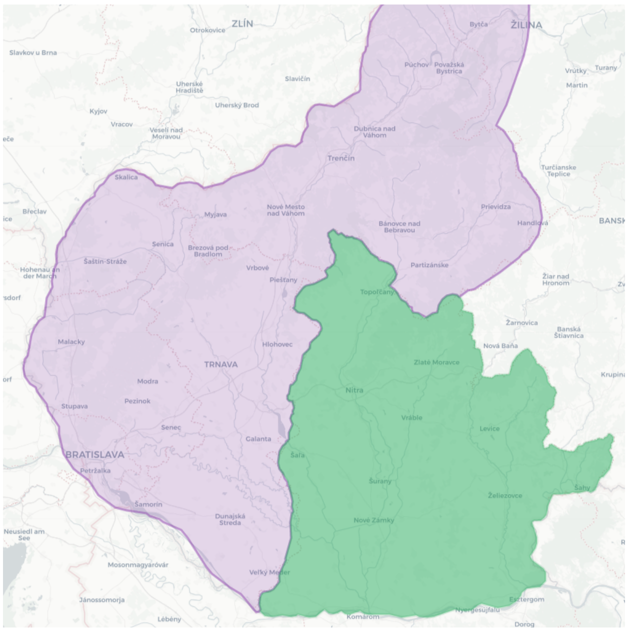

Figure 5. The figure shows the 'brush' tool, particularly, how a user can use it to create natural features like regions. Strokes are being subtracted, therefore giving the user ability to easily create borders.

be then appended to a geospatial object in the form of a context window, also known as 'pop-up'. With this simple workflow, users can create complex maps with their own data.

6. Conclusion

This paper provides information about the authoring system that helps users define geospatial objects and apply custom datasets to the created objects. It can be used to create thematic maps.

The first step of the project was a theoretical analysis, which brings a solid understanding of the basic

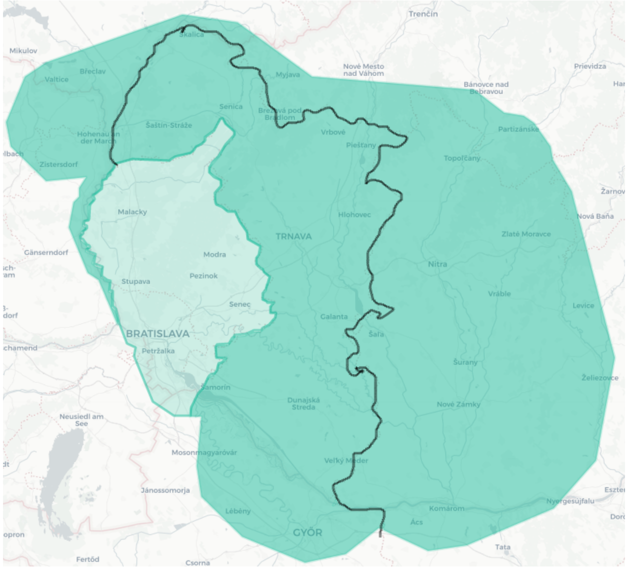

Figure 6. The figure shows a polygon that was created in a 'rough' fashion. This polygon was selected (shown by a more saturated color) and a black line was drawn over it. That ultimately divided the polygon into two parts and sculpted a region of a state in this case.

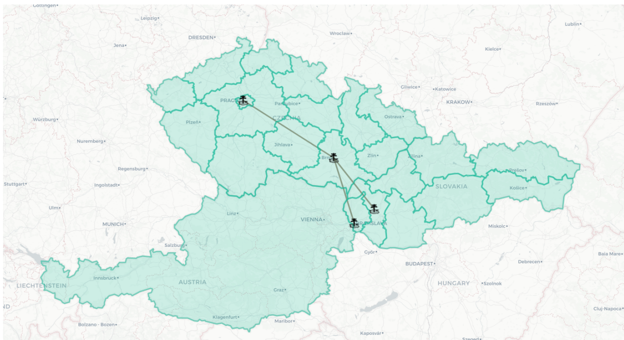

Figure 7. The search functionality allows to easily find certain places and create connections between them. The search for areas is used for quickly displaying known areas that can be further modified, so the user does not have to start from zero.

terms and helps narrow down the focus to specific usage scenarios. Therefore, deciding what kinds of features should be implemented. Ultimately, a palette of features was implemented. This palette contains several features, in particular, features for creating basic geospatial objects (points, lines, polygons), a feature for demarcating large areas, splitting polygons into multiple ones, a feature for prototyping networks, and a tool for searching particular places. The defined geospatial objects are customizable in terms of changing color, stroke width, marker icon, or description.

The advantage of the solution is recreating the functionality of vector programs while having features focused on creating geospatial objects. Defined objects have relations with each other. Users can customize them, change their shape, scale them up or down.

The solution stands out with the option of generic data mapping. This means that users can simply create thematic maps (choropleth map, connection map, point map, etc.) using their custom data.

At the current stage, the main features were initially implemented. There is still a lot of testing being done at the moment. That leads to improving performance, user experience, and changing or adding functionalities on top of the created ones.

Acknowledgements

I would like to thank my supervisor Ing. Jiří Hynek, PhD. for his help, advice, comments, and guidance in the course of making this project.

References

[1] The past, present, and future of geospatial data use. *USGIF 2018 State Future of GEOINT Report*, February 2018.

[2] omnisci. Geodata. https://www.omnisci.com/technical-glossary/geodata. [Online; accessed 23-March-2021].

[3] Geographical object types. https://www.restore.ac.uk/geo-refer/52611cwors00y00000000.php. [Online; accessed 22-March-2021].

[4] Iso 3166. https://www.iso.org/iso-3166-country-codes.html. [Online; accessed 26-March-2021].

[5] Víctor Olaya. *Introduction to GIS*. 2018.

[6] H. Butler. A Border Gateway Protocol 4 (BGP-4). RFC, RFC Editor, August 2016.

[7] Jiří Hynek., Jakub Kachlík., and Vít Rusňák. Geovisto: A toolkit for generic geospatial data visualization. In *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 3: IVAPP,*, pages 101–111. INSTICC, SciTePress, 2021.

[8] Vladimir Agafonkin. https://leafletjs.com/reference-1.7.1.html. [Online; accessed 22-March-2021].