Fine-Grained Map Coloring Web Service for JavaScript

Tetsuya Nakai[1(✉)], Sachio Saiki[1], and Masahide Nakamura[1,2]

[1] Graduate School of System Informatics, Kobe University,
1-1 Rokkodai, Nada, Kobe, Japan
manda@ws.cs.kobe-u.ac.jp, sachio@carp.kobe-u.ac.jp, masa-n@cs.kobe-u.ac.jp
[2] Riken AIP, 1-4-1 Nihon-bashi, Chuo-ku, Tokyo 103-0027, Japan

Abstract. In this paper, we focus on data visualization in the smart city. We propose a more feasible and portable technique of fine-grained map visualization for smart city analytics. More specifically, we develop a Web service called FigMap4SC (Fine-Grained Map-coloring Web service for Smart City). FigMap4SC provides a service of coloring fine-grained city maps from them. Existing visualization methods require users to prepare all geographic data and users have to perform many operations. In FigMap4SC, all geographic data and map drawing operations necessary for coloring a city maps are hidden under the service. By preparing only a dataset, users get fine-grained city maps without specialized knowledge. The service can eliminate user's trouble and operation. Moreover, FigMap4SC exposes Web-API, to which external applications can post the dataset. As a result, any applications and scripts of smart city analytics can integrate FigMap4SC to visualize their own analysis results on colored maps. We have implemented a prototype of FigMap4SC, currently covering Kobe city area only. FigMap4SC has been implemented as a Web application with JavaScript, HTML5 and Maps JavaScript API which can show Google Maps on this service easily. This service was used by the Kobe Fire Department staff which tried on smart city. They evaluated that the visualization was quick and the operation was easy.

Keywords: Smart city · Data visualization · GIS · Choropleth map · Web service

1 Introduction

With the development of the ICT systems and IoT technologies, many companies and local governments are acquiring and accumulating a wide variety of data, and utilization for town development is increasing. Some data are open to the general public as open data and people can use it for innovation. By utilizing the data, aiming for a more efficient and sustainable city, which is so-called *smart city* [6], is now a global trend. Especially in developed countries, data-driven approaches to optimize functions of a city as well as improve the quality of life of residents are quite a hot topic, which we call *smart city analytics* in this paper.

V. G. Duffy (Ed.): HCII 2020, LNCS 12199, pp. 159–174, 2020.
https://doi.org/10.1007/978-3-030-49907-5_11

The data used in smart city analytics are *geospatial information*, which typically tied to administrative divisions (i.e., city, ward, town, or block). Therefore, for understanding and the result of analyzing the data, *visualizing data on a map* is useful than a tabular format [3]. In general, people create a *Choropleth Map* [7] in which areas are colored in proportion to a value associated with the administrative division data. Before the smart city, the granularity of the data analysis was basically at a city or ward level. Hence, the visualization with the map was not a big problem, and even feasible by human hands. In the smart city analytics, however, the volume and the resolution of the data are far beyond the conventional ones. Therefore, the visualization with the map manually is extremely difficult.

A straight-forward approach to implement the fine-grained map visualization is to use conventional *Geographic Information Systems (GIS)* [8]. A GIS provides a variety of features for analyzing, visualizing, optimizing any kinds of geographic data. Since a user can do almost everything for a given map, the GIS can be used extensively in the smart city analytics.

However, as the drawback of the powerful and general-purpose features, a GIS requires a user to learn many proprietary operations for preparing datasets (maps, shapes, etc.), creating layouts (positions, layers, etc.), specifying representations (colors, textures, etc.), and so on. Basically, these operations are not essential for the smart city analytics itself.

Moreover, a GIS is typically a stand-alone application, where a user conducts all tasks within it. Therefore, it is difficult for external applications to dynamically integrate the visualization features of the GIS.

Our research goal is to propose a more feasible and portable technique of fine-grained map visualization for smart city analytics. More specifically, we develop a Web service called *FigMap4SC (Fine-Grained Map-coloring Web service for Smart City)*. FigMap4SC provides a service of creating fine-grained choropleth maps from given data. All geographic data such as the based map and the shape of administrative divisions are hidden under the service. A user just provides a dataset, consisting of key-values of a multi-grained address (i.e., city, ward, town, or block) and its associated value (numeric value and/or color code). For the dataset given, FigMap4SC automatically colors a city map by the designated granularity. Note that the input dataset does not rely on any specific systems or methods of smart city analytics. Thus, FigMap4SC can decouple the process of map-based visualization from the process of individual data analysis. Moreover, FigMap4SC exposes Web-API, to which external applications can post the dataset. As a result, any applications and scripts of smart city analytics can integrate FigMap4SC to visualize their own analysis results on colored maps.

We have implemented a prototype of FigMap4SC using Maps JavaScript API, JavaScript, and HTML5, currently covering Kobe city area only. To obtain Kobe city maps with multiple granularities, we extensively used Google Maps and Google Maps JavaScript API [4]. The shapes of Kobe city, wards, towns, and blocks have been borrowed from open data provided by e-Stat [2].

In order to confirm the effectiveness, we asked staff members of the Kobe City Fire Department which is conducting joint research with us to use this web service. They carried out case studies to visualize their rescue dispatch logs with this web service. As a result, we got feedback that FigMap4SC is more efficient and easier in visualizing the dispatch logs than the GIS used in the Kobe City Fire Department.

2 Preliminary

2.1 Data Visualization on a Map with GIS

A Geographic Information System (GIS) [8] is a system that can operate geographic information on a computer. A GIS provides a variety of features for analyzing, visualizing, optimizing any kinds of geographic data [1,5].

A general procedure for creating a choropleth map in GIS is as follows:

1. Prepare and load datasets (maps, shapes, etc.).
2. Provide the data to visualize.
3. Combine the shapes and the provided data.
4. Set some options (posisions, layers, and colors).

2.2 Research Tasks

The GIS has general and powerful features, however, it has the following problems in visualization.

P1: It requires operations that are not essential for smart city analytics
As mentioned in Sect. 2.1, when analysts create a choropleth map with GIS, they need to prepare not only the source data to visualize but also a based map and geographic (shape) data. In addition, combining the shape data and the source data is a complicated task. These tasks are equivalent to the user creating maps from scratch, which is not essential to the original smart city analytics.

P2: Setting visualization expressions is complicated
In order to create an ideal choropleth map, analysts need to experiment with the color and layout of shapes.

P3: It is difficult to link directly with external applications
A GIS is typically stand-alone applications, so external applications cannot dynamically operate the visualization features of the GIS. Hence, analysts need to manually load analyzed data created by external applications and provide them to the GIS.

P4: Cannot reuse map objects
When sharing the created choropleth map with other people, there are two ways, by converting it to an image, or by using GIS-specific data. In the first way, anyone can see it, however, detail shapes and map data is lost, and users cannot reuse them as objects. in the second way, users can reuse them as objects, however, they cannot see or reuse without GIS.

We are currently conducting a joint research study with the Kobe City Fire Department. Staff members of the Kobe City Fire Department aim to optimize ambulance operations. They analyze rescue dispatch logs accumulated over the years and optimize their resources. Until now, they have been trying to visualize data analyzed with Microsoft Excel on a map of Kobe City with their GIS. However, according to the complexity of the operation and the slow response of the system, they could not visualize the analyzed data satisfactory within a limited working time. These are the same issues as P1 to P4 above.

3 FigMap4SC

Our research goal is to propose a more feasible and portable technique of fine-grained map visualization. For that, we develop a Web service called FigMap4SC (Fine-Grained Map-coloring Web service for Smart City).

3.1 System Requirements

For the functions of FigMap4SC, we describe four system requirements corresponding to four issues.

R1: Hiding geographic information and operations
FigMap4SC prepares and manages the map data and geographic information required to create a choropleth map, and hides them from a user. In addition, FigMap4SC processes provided data on behalf of a user. Thereby, the user can create a choropleth map by only providing the data they want to visualize, and concentrate on their analysis works.

R2: Specify flexible visualization options
FigMap4SC allows a user to easily and flexibly specify visualization options (color, value range). Thereby, they can change the color and expression of the shape on the map easily.

R3: Link directly with external applications
FigMap4SC allows external applications to provide analyzed data to FigMap4SC directly and to operate them. Thereby, users can directly link their tools or programs with FigMap4SC for extensive analysis efficiently.

R4: Reuse visualization results
FigMap4SC saves a created choropleth map in this service as an object with a unique ID. A user can call and reuse the object at any time. By sharing the object ID to other people, they can check and reuse the same map without installing a dedicated application.

3.2 Overall Architecture

Figure 1 shows the overall architecture of FigMap4SC. The middle part of the figure surrounded by lines shows the system of FigMap4SC. the left part of the

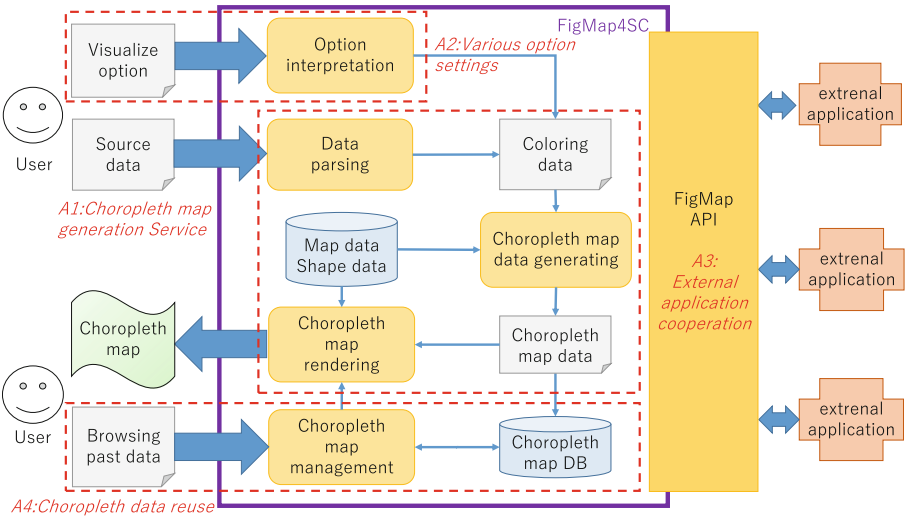

Fig. 1. System architecture

system shows analysts using FigMap4SC, and the right side of the system shows external applications using FigMap4SC. In order to fulfill the requirements R1-R4 described in Sect. 3.1, FigMap4SC is structured based on the following four approaches:

A1: Choropleth map generation service
FigMap4SC creates a choropleth map from the provided data and returns it to a user. The geographic data and operations required to create a choropleth map are included in this service.

A2: Visualization option specification
FigMap4SC accepts options from the user and customizes the colors and value range.

A3: FigMapAPI
This service allows a user to recalled and reused the choropleth maps which FigMap4SC created in the past.

A4: Choropleth map reuse service
This service allows a user to recalled and reused the choropleth maps which FigMap4SC created in the past.

The part enclosed by the dotted line in Fig. 1 and the part that realizes A1-A4 in FigMapAPI are shown. In the following sections, I will explain the input and output of the service first, and then describe the details of each of A1–A4.

3.3 Input Data

We define the input data that the user provides to FigMap4SC.

Visualization Data

We design the input data format so as not to depend on individual applications or analysis methods and so as to define as simple as possible. In this paper, we define the *input dataset* as pairs of *address part* and *value part*. Here, the address part represents an administrative division, the value part represents a value associated with it.

The address part is represented by *towncode* or *address*. The address is written in letters of the address of the administrative division. A user can specify any administrative divisions (i.e., city, ward, town, or block). The towncode is a numerical code uniquely registered for each region. Figmap4SC refers to this code and combines shape and data. In this paper, we adopt eleven digits numerical value used by the Statistics Bureau of Japan as towncode. We obtain *the towncodes from open data* released on e-Stat (Portal Site of Official Statistics of Japan) [2]. The eleven digits numerical value has a hierarchical system. The first five digits (the first to fifth digits) represents the prefecture and city, next three digits (the sixth to eighth digits) represents the town, and the last 3 digits (the ninth to eleventh digits) represents the block. Thereby, a user can select the geographic granularity level to be visualized by the length of towncode In FigMap4SC.

The value part is represented by l *value* or a *color*. The value represents a statistical and numerical value of the area, and a user can specify an integer of a real number. When a user provides a dataset to FigMap4SC, this service automatically creates 10 sections from the given maximum and minimum values and assigns a default color to each section. On the other hand, the color represents the color of the target area. A user specify a color code (#RRGGBB) or a color name (i.e. red, blue, and yellow).

In the address part, a user has to input either towncode or address. If users input both of them, towncode takes precedence. Similarly, in the value part, a user has to input either value or color. If users input both of them, color takes precedence. In addition, each data is allowed to have "description". The description is written in letters, and a user specifies comments.

Table 1. Example of source data format

ID	Address part		Value part	
	Towncode	Address	Value	Color
1	28101001001	Uozaki-Kitamachi 1-chome, Higashinada-Ku	4	#0000FF
2	28101001		40	
3		Higashinada-Ku, Kobe-Shi		Red

Table 1 shows an example of data. For the convenience of explanation, we assign IDs to the data in this example. At ID = 1, both towncode and address exist in the address part and both value and color exist in the value part.

Therefore, FigMap4SC gives priority to the towncode and color pair. Using this pair, the service colors the area of 28101001001 (Uozaki-Kitamachi 1-chome, Higashinada-Ku) in #0000FF (blue). At ID = 2, the towncode is specified by an 8-digit code. The service colors the area of 28101001 (Uozaki-Kitamachi, Higashinada-Ku, Kobe-Shi) in a color corresponding to the value 40. At ID = 3, the address is specified in the ward level, and the service colors the Higashinada-Ku, Kobe-Shi, which area is larger than the others, in red.

Visualization Option

A user can simultaneously provide the data to be visualized and the visualization option used in A2 to FigMap4SC. Users can freely change the colors with the visualization option.

The visualization option is defined by three required attributes, "from", "to" and "color". "from" and "to" are both numeric and define the class division section of the value "[from, to)". "color" represents the color of the target area.

Table 2. Example of option data format

from	to	color
−999	0	White
0	10	Blue
10	20	Yellow
20	30	Red

Table 2 shows examples of visualization options. This example defines four sections. If the value is greater than or equal to −999 and less than 0, paint white. If the value is greater than or equal to 0 and less than 10, paint blue. If the value is greater than or equal to 10 and less than 20, paint yellow. If the value is greater than or equal to 20 and less than 30, paint red. If the value is out of the range, does not color.

3.4 A1: Choropreth Map Generation Service

A1 generates a choropleth map based on the provided data. The center of Fig. 1 shows the structure of A1. A1 consists of master data (shown center cylinder) and three processed (shown rounded rectangles).

Master data has a map and shape of administrative divisions used to show a choropleth map. In FigMap4SC, We use the cloud map service for map master data. This time, we select Google Maps. We get the shape master data by e-Stat and form them. The shapes data is vector data, and have the eleven digits variable-length towncode as a key and arbitrary numbers of longitude and latitude list as the value. By connecting the pairs of longitude and latitude in the list, it represents the boundaries of administrative divisions.

In the *data interpretation process*, FigMap4SC checks the missing data for the provided data and generates "coloring data", which have pairs of administrative divisions and colors. If the visualization option is not specified, the service automatically determines the colors for each value according to the following procedure.

1. Store a default color map consisting of ten colors in the service in advance.
2. Extracts the minimum and maximum value from the provided data set and calculates the difference.
3. Divide the data set into ten class divisions to equal the widths of the data values, and correspond ten colors to each division.
4. Refer to each value and assign the color corresponding to the class division.

In the *coloring map data generation process*, FigMap4SC connects (inner-joins) the coloring data and shape data on towncode, and creates "Choropleth map data" For the connecting, all addressed are converted to the corresponding towncode. The choropleth map data is sent to the choropleth map rendering process and also saved in "Choropleth map database" for other functions.

In the *choropleth map rendering process*, FigMap4SC sends the choropleth map data to the cloud map service and create a choropleth map object. The service displays a Choropleth map on a web browser, and a user can freely enlarge, reduce, and move on their browser.

Accordingly, a user can create a choropleth map from the input data without any data preparation or operation related to map creation.

3.5 A2: Various Option Settings

In A2, FigMap4SC interprets the visualize option user provides and generates "Coloring data" according to the user-defined class division. When a user provides the visualization option, the service gives priority to the user definition.

This allows the users to create a more flexible choropleth map using the desired range and color without any manipulation on the data.

3.6 A3: External Application Cooperation

In A3, FigMap4SC provides a Web-API (we call as FigMapAPI). FigMapAPI allows any external applications to input data directly to FigMap4SC. They can directly input data by specifying query parameters for the URL of FigMap4SC. If users want to provide the data from external application by value, pass the dataset in JSON format to the data parameter. On the other hand, if users want to provide the data by reference, pass the reference URL to the data parameter. The service interprets the data according to the type of the given query parameter and creates a choropleth map through A1 and A2.

Thereby, A3 allows users to call FigMap4SC directly from any program or script without any manual intervention. Hence, users can efficiently visualize data. In addition, the service is not limited to analytics. For example, visualizing the aggregate value of sensors placed in a smart city on a map. The range of services is expanded to dynamic data visualization.

3.7 A4: Choropleth Map Reuse Service

In A4, FigMap4SC calls the past choropleth map data object from the choropleth map database and reuse it. A4 consists of choropleth map database and choropleth map data management. The database stores datasets with unique map IDs as keys and visualization data as values. A user can check the map ID stored in the database via the color map data management. By providing the map ID to the choropleth map data management, it calls the choropleth map object from the database and renders it and show them. By sharing the map ID with other users, they can confirm the same map in different environments. It is also possible to copy, update, and delete the map.

Thereby, a user can reuse a past choropleth map without preparing data again. In addition, anyone who can access Web services can easily share and check the same choropleth map.

4 Implementation

We have implemented a prototype of FigMap4SC, currently covering Kobe city area only.

4.1 Technologies

The technologies for the prototype of FigMap4SC are as follows:

- The shapes of Kobe divisions: World Geodetic System (KML files) by e-Stat
- Borrow and form shape: Python
- Database: MongoDB
- Database access API: Java1.8. Tomcat7.0
- Data process: JavaScript
- Map render: Google Maps JavaScript API
- User interface: HTML5, JavaScript

4.2 Main Screen of FigMap4SC

Figure 2 shows the main screen of FigMap4SC. The left side of the figure shows a data operation part and the right side of the figure shows a map part. In the map part, the service display Google Maps by Maps JavaScript API [4]. A user can perform almost the same operations as the Google Maps web application. Specifically, they can zoom in and zoom out the map, and they can replace an aerial photograph.

In the data operation part, service shows a color bar representing the legend of the data, a form providing the data to visualize, an operating UI for adjusting the visualization result, and a button for saving the visualization result. The color bar displays its color and value based on the class division.

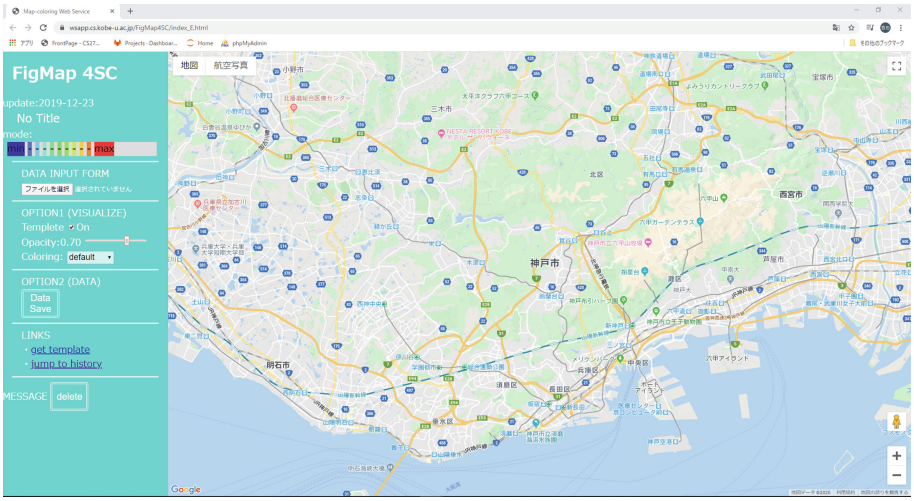

Fig. 2. Main screen of "FigMap4SC"

In this prototype, a user can use the Excel(xlsx) format or JSON format file to visualize. In addition, we prepare an *Excel format template file* so that users can visualize immediately. In this template, there are the address parts of three types of administrative divisions in Kobe city, town, and block that have been entered on three different sheets. Users can easily create visualized data by filling in the value part on the sheet in their desired granularity. A user can also define visualization options in another sheet.

By operating the slider and pull-down menu in the data operation part, a user can partially manipulate the visualization results. By operating the slider, a user can change the transparency of the colors in a choropleth map. In addition, by selecting a different color map from the "Coloring" pull-down menu, a user can change the colors of areas from the default color. When a user gets a satisfactory visualization result, they can save the created choropleth map data object to the database by pressing the DataSave button. By clicking on any area on the map, you can check the information of the area. Since the choropleth map is projected on the Google map, users can zoom in, zoom out, and move the map.

Figure 3 shows the screen of the reusing visualization results of FigMap4SC. The left side of the screen shows a list contains the title and created date of choropleth maps. When a user clicks the title, the service redraws the choropleth map. At the same time, the map ID is copied to the clipboard. By sharing the ID with other users, users can share the same map.

4.3 Direct Integration with External Applications

In this section, we describe the procedure for an external application to provide data directly to FigMap4SC and create a Choropleth map. When the external

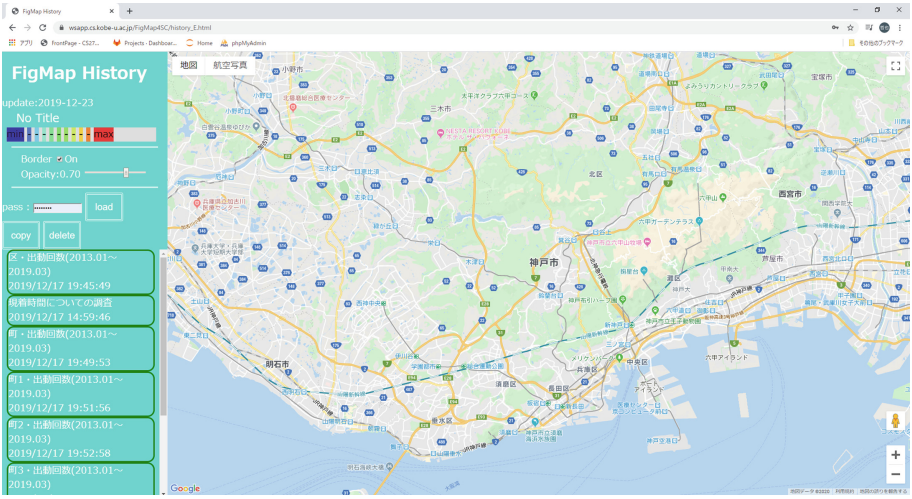

Fig. 3. Past data browseing screen of "FigMap4SC"

application provides its data to FigMap4SC by value, enter the data in JSON format text to the data query parameter, and call the URL of FigMap4SC. On the other hand, when provides by reference, write the JSON format text data to a file with a certain URL, pass the URL to the URL query parameter, and call.

5 Case Study

In order to confirm the effectiveness of FigMap4SC, we conduct a case study to visualize the dispatch logs.

5.1 Case1: Visualization of the Number of Rescue Operations per Year in Block Level

The Kobe City Fire Department records its rescue operations in dispatch logs. For each dispatch log, the date and time of dispatch, the destination address of the ambulance, and the condition of the victim was recorded. By creating a choropleth map about the destination address at the block level, they can find a detailed distribution of emergency demand and regional characteristics. In case1, we count the number of emergency dispatches for each block in 2018 and visualizes by FigMap4SC.

First, we extract the dispatch data from raw data and count them for each block with the external analysis tool (Python) and past the results into an Excel template file. Also, color information is not input because it is automatically assigned by FigMap4SC. Table 3 shows a part of the analysis result.

Figure 4 shows the result of visualization by providing this Excel file into FigMap4SC. Figure 5 shows the result of zooming in around *Higashi-kawasakicho*

Table 3. Source data in case1

towncode	address	value	color
28101001001	Uozaki-Kitamachi 1-chome, Higashinada-Ku, Kobe-Shi	27	
28101001002	Uozaki-Kitamachi 2-chome, Higashinada-Ku, Kobe-Shi	23	
28101001003	Uozaki-Kitamachi 3-chome, Higashinada-Ku, Kobe-Shi	15	
28101001004	Uozaki-Kitamachi 4-chome, Higashinada-Ku, Kobe-Shi	23	
28101002001	Uozaki-Nakamachi 1-chome, Higashinada-Ku, Kobe-Shi	57	
28101002002	Uozaki-Nakamachi 2-chome, Higashinada-Ku, Kobe-Shi	52	

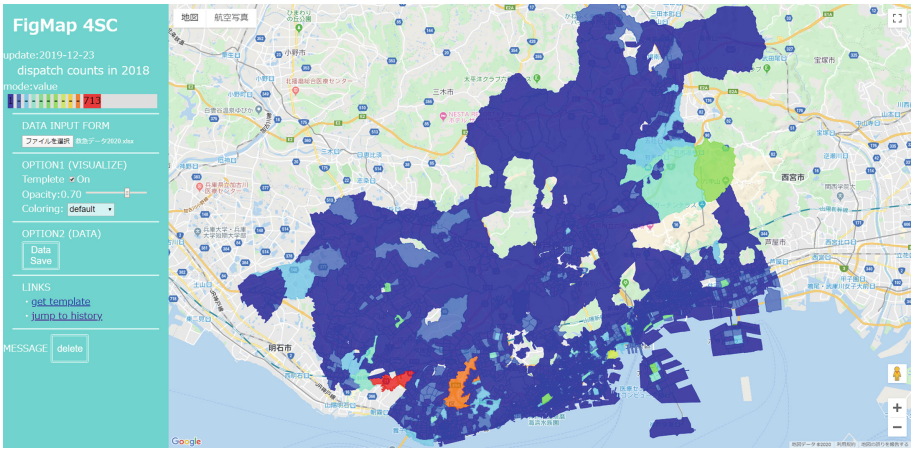

Fig. 4. Visualization of the number of dispatches in 2018

1-chome, Chuo-Ku, Kobe-Shi. The top left of Fig. 4 shows the legend. The number of dispatches decreases as the color of the division approaches blue, and the number of dispatches increases as the color approaches red. Looking at the map part, it can be seen that most of the areas are blue, and the number of dispatches is small. On the other hand, some areas are colored green, yellow, orange, and red, indicating that the number of dispatches is significantly higher than in other areas.

Figure 5 shows the zoomed-up map of the *Chuo-Ku* in the center of the screen in Fig. 4. We can see the difference in the emergency demand at the block level. In this figure, *Higashi-kawasakicho 1-chome* painted in yellow-green had 458 counts, indicating that there was much higher demand than in the surrounding area. We consider that the demand is caused by Kobe Harborland, a large commercial facility. Many people gather to the facility so that there was a lot of emergency demand. In addition, the area near the SN station in the upper right of the figure is colored green. This suggests that rescue operations increase in places where there were heavy traffic of people.

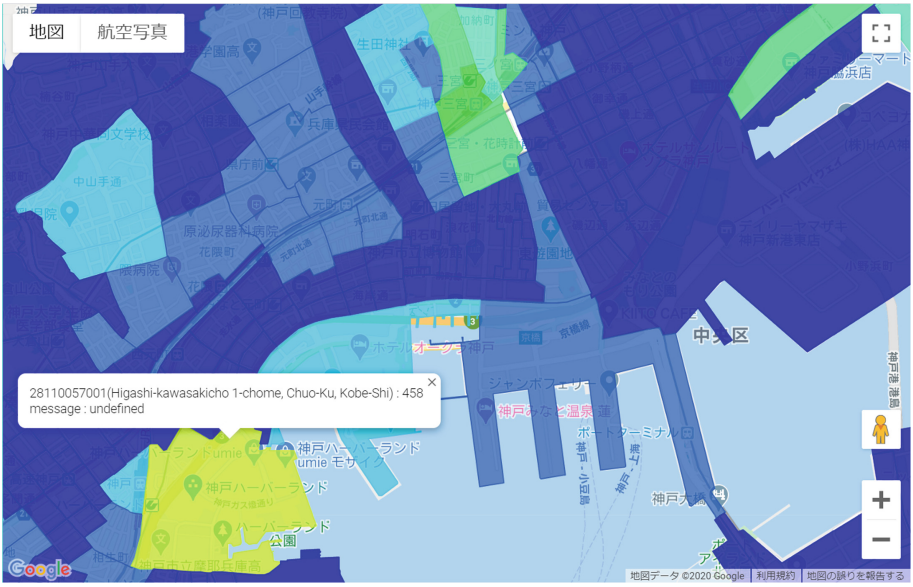

Fig. 5. Detail about the subject area

5.2 Case2: Visualization of Emergency Dispatch Time by FD-CAST

In case2, we provide analyzed data to FigMap4SC with an external application directly and visualize it. We use FD-CAST [9], a developing application in collaboration with the Kobe City Fire Department, as the external application.

FD-CAST (Fire Department Configuration Analysis and Simulation Tool) is a tool that estimates the arrival information of ambulance. At this time, a user can set the **configuration information** such as the location of the fire station and the vehicle formation. This tool is used for the purpose of examining the optimal arrangement and operation of resources.

Figure 6 shows an example of the analysis results on FD-CAST. This figure shows how many meters and minutes it takes ambulances to reach each town in Kobe. The service shows the first to tenth closest ambulances for each area. These values were calculated from **the route on the map**, so please note that different from the actual rescue squad time.

FD-CAST has already implemented the link function with FigMap4SC. By pressing the FigMap button at the top of each column in the table, you can create a choropleth map of the data in that column. Next, we will visualize the distribution of the dispatching time of ambulances from the nearest fire department for each block. Figure 7 shows the result of pressing the figMap button in the first column.

Naturally, areas closer to the fire station are painted blue, and areas farther from the fire station are painted red. Such visualization results can be used for

Fig. 6. Screen of "FD-CAST"

future planning such as changing the location of fire stations or establishing new ones.

5.3 Feedbacks from Practitioners

At present, the Kobe Fire Department staffs are using FigMap4SC for their work. In this section, we introduce the advantages and opinions of FigMap4SC from them.

Advantages of FigMap4SC

- Compared with some GIS, the creation of a choropleth map in FigMap4SC is easy and understanding the result speedily.
- Excel files can be used and instantly visualized on a map. Therefore, ready the data to visualize is very easy. In addition, it is good to be able to try various analyzes by trial and error.

This feedbacks show the effectiveness of FigMap4SC, such as clarity of operation and ease of visualization. On the other hand, they raise the following opinions.

Improvements in FigMap4SC

- Some addresses are not displayed on FigMap4SC.
- I want to display multiple analysis results on the same map.

Investigation of the first opinion reveals that they enter some addresses not covered by FigMap4SC. These are old addresses that are not currently used or addressed using different characters and so on. Hence, they did not match the

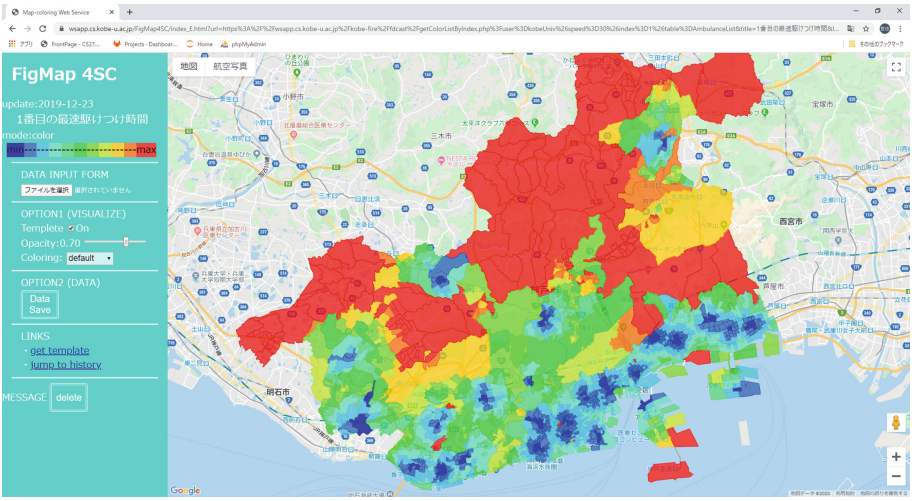

Fig. 7. Visualization of the result FD-CAST analysed

address assumed in FigMap4SC, and they became missing data. In such case, it is necessary to match the given address with the most likely existing address.

The second opinion is not possible with the current implementation. The service can generate only one choropleth map in one visualization. It is technically easy to overlay multiple class maps on the same map, however, the visibility may be low. Hence, the service needs to overlay different analysis results with different expressions for high visibility. For example, the first data is represented by a choropleth map and the second data is represented by pins.

These are future tasks.

6 Conclusion

In this paper, our research goal was to propose a more feasible and portable technique of fine-grained map visualization. In order to achieve this, we proposed FigMap4SC (Fine-Grained Map-coloring Web service for Smart City). The service hides all geographic data and a map under this service. In addition, the service performs the choropleth map generation process automatically and separates the data analysis work from the visualization process. As a result, a user can create a choropleth map easily even if they do not have expertise in geographic information systems. Moreover, the service can directly cooperate with an external application and can share the choropleth map. This enables quick and efficient visualization.

We have implemented a prototype of FigMap4SC, and we conducted a case study to visualize dispatch logs recording by Kobe City Fire Department. From their feedbacks, improvement of operability and trial of efficient analysis were evaluated as the effectiveness of the proposed service Compared with the existing

GIS. As future research, we would like to consider the mechanism to correct different address notations and overlay multiple visualization data.

Acknowledgements. This research is a joint research with Kobe Fire Department. This work was partially supported by JSPS KAKENHI Grant Numbers JP19H01138, JP17H00731, JP18H03242, JP18H03342, JP19H04154, JP19K02973.

References

1. Arcgis—esri japan corporation. https://www.esrij.com/products/arcgis/. Accessed 30 Jan 2020
2. e-Stat: portal site of official statistics of Japan. https://www.e-stat.go.jp/. Accessed 18 Oct 2019
3. Geographic information. https://www.esrij.com/getting-started/what-is-gis/. Accessed 30 Jan 2020
4. Maps JavaScript API. https://developers.google.com/maps/documentation/javascript/. Accessed 18 Oct 2019
5. QGIS project. https://qgis.org/en/site/. Accessed 30 Jan 2020
6. Deakin, M., Al Waer, H.: From intelligent to smart cities. Intell. Build. Int. **3**(3), 140–152 (2011)
7. Kulhavy, K.A.R.W.: Learning and remembering from thematic maps of familiar regions. Educ. Technol. Res. Dev. **46**(1), 19–38 (1998)
8. Maliene, V., Grigonis, V., Palevičius, V., Griffiths, S.: Geographic information system: old principles with new capabilities. Urban Des. Int. **16**(1), 1–6 (2011)
9. Yabuki, N., Saiki, S., Nakamura, M., Oyama, K.: FD-CAST: a tool for analyzing and simulating fire department configurations. In: HCII 2020. LNCS, pp. 199–213. Springer, Heidelberg (2020)