A Web Viewer for Visualizing Crime Data in Fairfax County

Xiaoxuan Li
George Mason University
Dept. of Geography and Geoinformation Science
4400 University Drive
Fairfax, VA 22030-4444
xli50@gmu.edu

ABSTRACT

In this paper, a GIS-based web mapping application is presented for visualizing and analyzing the crime events that occurred in Fairfax county in 2019. The technology stack used in this case study includes Google Geocoding API, CartoDB, HTML, CSS, JavaScript, and Leaflet. Dynamic charts, information window, geospatial markers, and multiple layers controls are generated in the web interface to explore crime data by date, crime type, and location. This application allows users to check the overall crime hotspots via thematic maps or investigate detailed crime statistics related to a specific address. This application is designed to help who are interested in the crime patterns in Fairfax county across different spatial and temporal scales.

Categories and Subject Descriptors

H.2.8 [Database Management]: Database Application – *Spatial databases and GIS.*

General Terms

Design, Human factors, Languages.

Keywords

Web-based GIS, JavaScript, Web mapping, Web design, Crime visualization, Carto, Leaflet

1. INTRODUCTION

Crime is not distributed evenly across regions. People seek to find novel methods to detect those crime hotspots such that crimes are concentrated. Crime mapping is one of the most important tools in visualizing and analyzing these crime events within heterogeneous backgrounds [1]. It provides us the knowledge to highlight suspicious events and spatial crime dynamics. However, due to limited techniques, tradition maps are not easy to access and generate. In contrast, mapping crimes on the web is more advanced and widespread due to its excellent interactivity, reliability, and functionality. Specifically, JavaScript is very powerful to display geographic and socio-economic datasets, including crime events. In this study, a web-based application is presented to map crime events in Fairfax county in 2019.

The objective of this project is to use open-source software, APIs and databases to build a web viewer such that users can explore crime statistics, trends, and patterns at different temporal-spatial scales.

The detailed design and functionalities are illustrated in the following sections. Section 2 describes the primary datasets. Section 3 describes the software, database, and developer environment. Section 4 describes the design of map viewer and functionalities. Section 5 describes the demo, and Section 6 describes the project limitation, design improvement, and future study.

2. DATASETS

There are two datasets used in this study: crime event records and city shapefiles in Fairfax county. All datasets are uploaded on the CartoDB web services for further use.

2.1 Crime records

The crime records were initially distributed by the official governor of Fairfax county. Their datasets, named "My neighborhood," have not been well maintained in proper format. However, an online crime map and crime alert service named "SpotCrime.com" [2] successfully implemented novel methods to harvest those old crime datasets from Fairfax county's official website. To date, The Fairfax county governor has agreed to offer formal and accurate crime information to SpotCrime.com [3]. In this case, we can download high-quality crime data from SpotCrime.com. However, the original data collected from this website cannot be directly used because of missed geographic information. What they provided is just address information, which should be geocoded before uploading to CartoDB. To deal with the geocoding problem, Google Maps Geocoding API from Google Maps Platform was requested in python environment to geocode the crime data collected from SpotCrime.com [4]. After geocoding process, the crime data was uploaded to CartoDB, which consists of 27,015 rows in total spanning the 1st January 2019 through the 15th November 2019 time frame in Fairfax county. Besides, for each crime event, a crime type will be assigned based on their metadata. In this study, 7 crime types are included: assault, arson, arrest, burglary, robbery, shooting, and theft. The crime data stored in CartoDB will be later accessed and visualized in Codepen frontend project via CartoDB APIs.

2.2 City boundary shapefiles

The city boundary shapefiles were distributed by Geospatial Data on Fairfax County official website, which contains 24 cities in total [5]. It should be noted that these city boundaries are not classified based on their administrative boundaries but zip code. Before uploading to CartoDB, for simplifying data visualization, crime events in section 2.1 were spatially joined with these city polygons to obtain the count of crime events for each individual city in ArcGIS Pro. The resulting spatial join counts were then used to create a choropleth map in Codepen.

3. ARCHITECTURE

In this study, the web viewer was built based on three components: database, code editor, and viewer interface. The following contents in this section introduce the database and code editor, while the viewer interface will be described in section 4.

Database: CartoDB is an open-source cloud computing platform that can store, query and map geospatial datasets built in PostgreSQL [6]. In this study, both crime event points and city boundary polygons were uploaded into CartoDB. By requesting Carto APIs in JavaScript, crime datasets can be accessed as JSON features. The following query was used to query and update crime features (part of the query in Carto):

Update crime_point_clean SET month_c = 1 WHERE

EXTRACT(MONTH FROM date) = 1

Update crime_point_clean SET crime_c = 1 WHERE

crime_type = 'Arrest'

Where "crime_point_clean" represents crime points, month_c represents the month index, and crime_c represents the crime type index. The queries showing above calls the function in Carto such that two new columns regarding the month (crime type) index was created and updated by querying the date (crime type) information in original crime datasets. These two columns of index were then used in the following JavaScript functions to visualize crime events.

Code editor: Codepen is a web-based code editor that includes HTML, CSS, and JavaScript for front-end designers and developers to learn and test their programs using web techniques [7]. The map viewer presented was accomplished in JavaScript, HTML, and CSS.

4. WEB INTERFACE

There are two main parts in the web interface: map viewer and chart viewer. In Figure 1, the initial map viewer to the left is displayed with the cluster layer and heatmap layer on. The map viewer contains layer control in the upper right, timeline brush control across the bottom, and ESRI GeoSearch control in the upper left. On the chart canvas to the right, line chart is on the top and bar chart is on the bottom. The following parts will specifically talk about each viewer and functionalities.

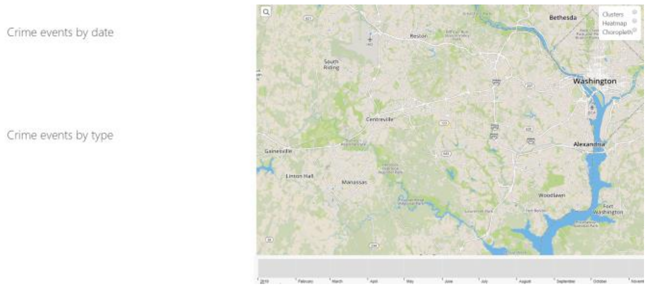


Figure 1. Snapshots of the initial web interface.

4.1 Map viewer

4.1.1 Layer control

There are three layers listed on the switch container of the layer control showing in the upper right. These layers, representing the cluster map, heatmap, and choropleth map, provide the feasibility of visualizing crime events in different ways at a broader scale (Figure 2). To visualize any layer in the switch container, users may click on the radio button on the left of each layer. Because three radio buttons are toggled by only one menu with the same name, users can only click on one radio button at once. When any radio button is clicked, the other two layers with unchecked radio button will be removed from the map. The datasets uploaded in Carto will be used in JavaScript via CartoDB APIs. Cluster map and heatmap use the same dataset named "crime_point_clean" as mentioned in section 3, while choropleth map uses the city boundary data uploaded in Carto. The "join_count" attribute in city data, representing the crime count in each city polygon, is used to assign map color and create a legend control in the lower right. Another interesting thing in this section is that users can zoom in to see the custom icons for different crime types in cluster map (Figure 3). Collected from Free Icons Library [8] and stored in Dropbox, these icons were assigned to each JSON feature based on crime type (Figure 4).

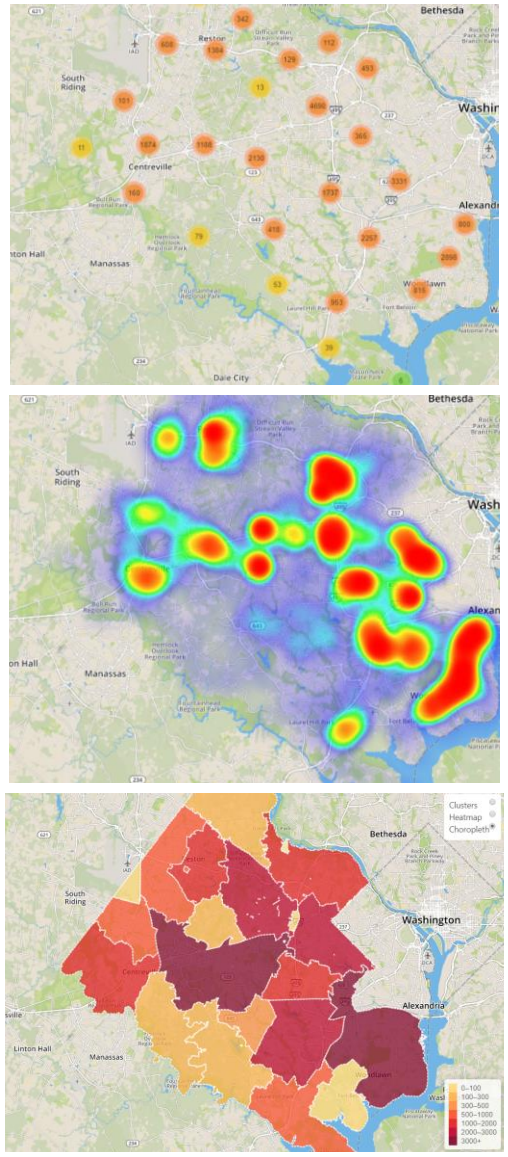


Figure 2. Snapshots of the cluster map (top), heatmap (middle), and choropleth map (bottom).

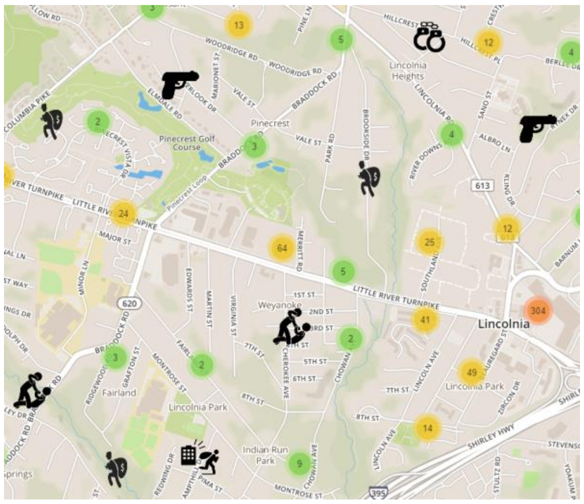


Figure 3. Snapshots of the zoomed cluster map.

Figure 4. Crime types from left to right: theft, arrest, assault, burglary, arson, robbery, and shooting.

4.1.2 *Time brush control*

By scrolling the brush on the bottom, the cluster layer and heatmap can dynamically display the patterns of crime events updated on that specific time range (Figure 5). Due to different datasets used in three layers, this functionality cannot perform on choropleth map and chart generation. Besides, due to special function design, after scrolling the brush or updating searched location, users may click on the brush bar again if they want to visualize cluster map layer.

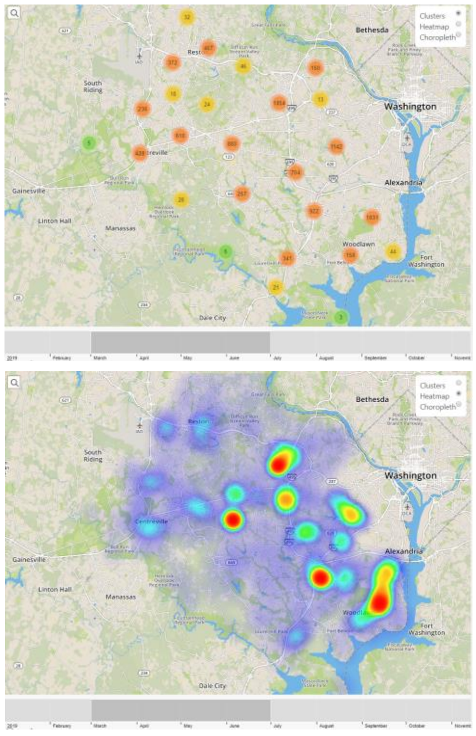
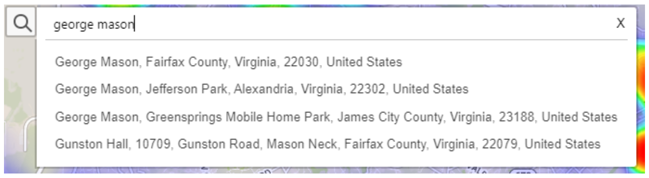


Figure 5. Filtered clusters and heatmap from March to July

4.1.3 *ESRI GeoSearch control*

Served as a kind of leaflet control, ESRI GeoSearch is a plugin function specifically for auto-complete enabled search [9]. Users just click on the search button in the upper left and search any location they like (Figure 6). This leaflet control is capable of obtaining the coordinates and addresses of entered location in the search bar. In this study, a new marker with red pulse will be placed to the targeted location for further use. Also, when the marker is generated, users may close the search bar to see three circles created around the targeted location (Figure 7). These circles represent the 3 km, 6 km, and 10 km buffered areas around this targeted location. The summation of any points within any buffered areas will be calculated and included in statistical analysis and chart generation in section 4.2.



george mason

George Mason, Fairfax County, Virginia, 22030, United States
George Mason, Jefferson Park, Alexandria, Virginia, 22302, United States
George Mason, Greensprings Mobile Home Park, James City County, Virginia, 23188, United States
Gunston Hall, 10709, Gunston Road, Mason Neck, Fairfax County, Virginia, 22079, United States

Figure 6. Search a location like what you did on Google Maps

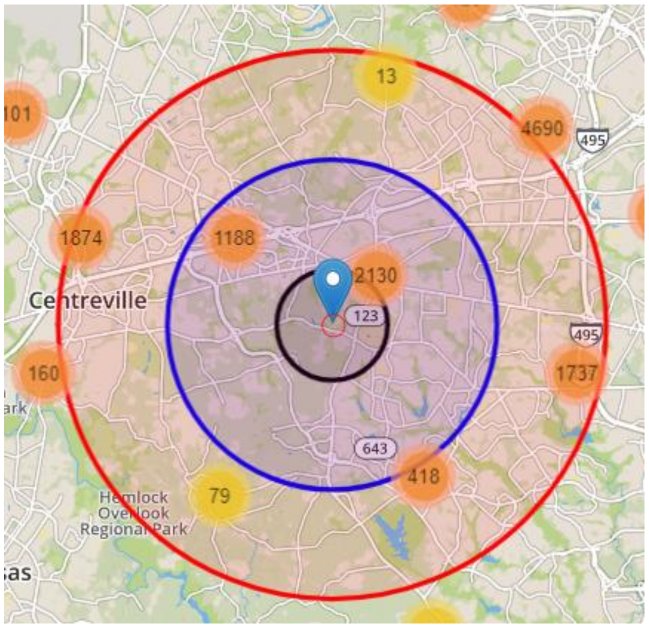
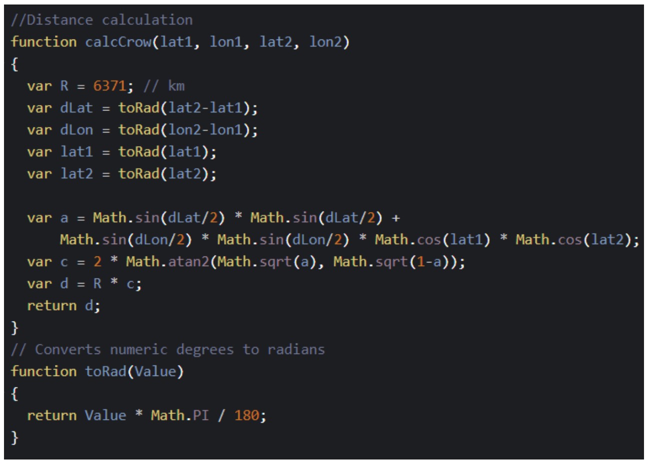


Figure 7. Target pulse marker and three buffered areas.

4.2 Chart viewer

The chart generation event will be triggered when users click on the marker created in section 4.1. The distance calculation function (Figure 8) is primarily for calculating the distance between two points whose coordinates are in degree. The resulting distance returns a distance in kilometer. For those points whose distance to the targeted location is smaller than the radius of the buffered area, they will be selected to calculate the count value. Then, crime counts calculated within three buffered areas will be used to form the elements of the line chart and bar chart. The colors of line chart and bar chart are assigned based on different buffered areas (3 km: black, 6 km: blue, 10 km: red). When the marker is clicked, two charts will be generated on the canvas to the right. A popup window showing the address information will also be displayed on the map. Furthermore, the crime event counts within three buffered areas for the specific location are displayed in the lower right of the map (Figure 9).



```
//Distance calculation
function calcCrow(lat1, lon1, lat2, lon2)
{
  var R = 6371; // km
  var dLat = toRad(lat2-lat1);
  var dLon = toRad(lon2-lon1);
  var lat1 = toRad(lat1);
  var lat2 = toRad(lat2);

  var a = Math.sin(dLat/2) * Math.sin(dLat/2) +
      Math.sin(dLon/2) * Math.sin(dLon/2) * Math.cos(lat1) * Math.cos(lat2);
  var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
  var d = R * c;
  return d;
}
// Converts numeric degrees to radians
function toRad(Value)
{
  return Value * Math.PI / 180;
}
```

Figure 8. Snapshot of distance calculation function.

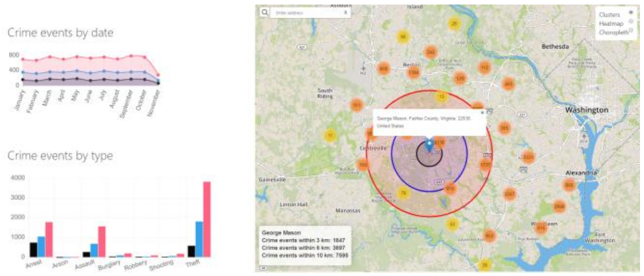


Figure 9. Line chart (upper on the canvas), bar chart (lower on the canvas), popup (on the marker), and crime counts (lower left on the map) for the specific location searched.

In the line chart, three lines represent the crime counts within three corresponding buffered areas categorized by month. While in the bar chart, three bars represent the crime counts within three corresponding buffered areas categorized by crime type. The chart will dynamically change once users click on a new marker placed on the new location marked via the ESRI GeoSearch Control.

5. Demo

The demo will present all the functionalities mentioned in previous sections. For map viewer, the demo may first try the radio buttons in the switch container to show/hide three map layers. Then, by scrolling the time brush on the bottom, the cluster layer and heatmap layer will change accordingly. In this case, the demo will click on the first button to show the cluster layer and remove heatmap layer and choropleth layer. To ensure that the brush works well, the demo needs to click on the brush bar again to show the resultant cluster layer. Then the demo will check the second radio button to display heatmap layer. Finally, the demo will check the last one to show the choropleth layer. Before we move to the chart viewer part, the demo should search a location on the ESRI GeoSearch control in the upper left. After entering a targeted location, the demo will close the search bar, and a marker with three buffered areas will show up. By clicking on the marker, the line chart and bar chart will dynamically generated. Furthermore, a popup window and a statistic window will also show up to display address information and the count of crime events within specific buffered areas. To start over, the demo can re-enter a new location on the search bar, then a new marker with three buffered areas will be created after closing the search bar. Finally, I would conclude that the demo has the potential to build a web viewer such that users can explore crime statistics, trends, and patterns at different temporal-spatial scales.

6. Future work

The web-based mapping application is a basic version such that study sites, data preparation, code improvement, and mapping technologies are limited to the work schedule and deadline. Future work will focus on an even broader study area, such as the Washington Metropolitan Area I mentioned in my project outline. Accordingly, the data is going to be much larger compared to what I collected in this research. A big challenge for this project is that the reaction time for click events and data loading is relatively long. It will be an issue if a vast volume of data is included. In this case, a further code improvement should be made to keep pace with new datasets. For example, turf.js is an excellent JavaScript toolset for processing geospatial data [10]. Further study may investigate how to use turf.js to deal with dynamic buffer, brush control, and intersection problems.

7. Acknowledgment

Many thanks to Professor Pfoser at George Mason University for providing valuable guidance and assistance on this project.

8. REFERENCES

[1] Eck, J., Chainey, S., Cameron, J., & Wilson, R. (2005). Mapping crime: Understanding hotspots.

[2] SpotCrime. (n.d.). Retrieved from https://spotcrime.com/.

[3] CrimeReport. (2019, October 10). Retrieved from https://www.fairfaxcounty.gov/crimereports/.

[4] Geocoding. (n.d.). Retrieved from https://developers.google.com/maps/documentation/geocoding/start.

[5] GISData. (2019, October 10). Retrieved from https://www.fairfaxcounty.gov/gis-data/.

[6] CartoDB. (2019, September 27). Retrieved from https://en.wikipedia.org/wiki/CartoDB.

[7] CodePen. (2019, November 16). Retrieved from https://en.wikipedia.org/wiki/CodePen.

[8] Free Icons Library. (n.d.). Retrieved from https://icon-library.net/.

[9] Smeijer. (2019, August 19). smeijer/leaflet-geosearch. Retrieved from https://github.com/smeijer/leaflet-geosearch

[10] TURF. (n.d.). Retrieved from https://turfjs.org/.