PolyGuide: A Polygon-based Approach for Sptial Feedback Capturing

Plácido A. Souza Neto, Behrooz Omidvar-Tehrani, Tiago Oliveira Lisboa, Francisco B. Silva Junior, and Felipe F. Pontes

Federal Institute of Rio Grande do Norte (Brazil)
placido.neto@ifrn.edu.br
{tiago.oliveira,bento.francisco,freire.pontes}@academico.ifrn.edu.br
University of Grenoble Alpes (France)
behrooz.Omidvar-Tehrani@univ-grenoble-alpes.fr

Abstract. In this paper we present a solution to capture region preferences from implicit feedbacks. Given a point from a spatial dataset, a set of regions captured from gaze and mouse tracking, the set of region intersections, the PolyGuide captures the implicit feedback of analysts and exploits it to highlight potentially interesting options. The analysis consider the concept of relevance and diversity of a given point with the others in the same data set, from the regions tracked during the analysis. To tackle this challenge, we extend GeoGuide[6] approach by using ST-DBSCAN[3] algorithm to map the region preferences from implicit tracking over time. We capture, analyse, generate and save region preferences in order to highlight informations, from any spatial dataset that can be useful to the analyst. So, this work aims to answers questions about interactivity and guidance in spatial solutions. We evaluate the efficiency of the proposed approach experimentally considering spatial datasets.

Keywords: PolyGuide, ST-DBSCAN, mouse-tracking, implicit preferences, regions

1 Introduction

Nowadays, there has been a meteoric rise in the generation of spatial datasets in various fields of science, such as transportation, lodging services and social science. As each record in spatial data represents an activity in a precise geographical location, analyzing such data enables discoveries grounded on facts. Analysts are often interested to observe spatial patterns and trends to improve their decision making process. There exist many applications which require the analysis of spatial data such as smart city management, disaster management and autonomous transport [8, 9].

Why do we need feedback? Why do we need implicit feedback in spatial analysis?

Spatial data analysis is often performed in *exploratory context*, i.e., the analyst does not have a precise query in mind and she explores data in iterative steps in order to find potentially interesting results. Traditionally, an exploratory analysis scenario on spatial data is performed in trial-and-error iterations: the analyst first visualizes a subset of data using a query in an off-the-shelf product (e.g., Tableau[1], Exhibit[2], Spotfire[3]). The result will be illustrated on a geographical map. Then she investigates on different parts of the visualization by zooming in/out and panning the map in order to discover patterns and trends of interest. The analyst may iterate on this process several times by issuing different queries and focusing on different aspects of data.

The literature in spatial data analysis has a focus on *efficiency* of exploratory iterations: *"how can analysts navigate in spatial data fluidly?"* The common approach is to design pre-computed indexes which enable efficient retrieval of spatial data (e.g., [5]). However, there has been fewer attention to the *wisdom* derived from spatial data. Despite the huge progress on efficiency front, an analyst may easily get lost in the plethora of geographical points because *i*. she doesn't know what to investigate next in an exploratory context and *ii*. she may get distracted and miss interesting points by visual clutter caused by huge point overlaps. In other words, although iteration transitions (between one analysis step to the other) can be performed efficiently, the decision which forms a transition, remains unclear.

In this paper, we focus on the question of *"what to see next"* in exploratory analysis of spatial data. We believe that there should be a *guidance layer* on top of the raw visualization of spatial data to provide recommendation on next potential options that the analyst should focus on. This guidance should be a function of analyst feedback: the system should recommend options similar to what the analyst has already appreciated. Hence, feedback capturing lies at the core of such guidance system.

While there has been a focus on explicit feedback on the literature, our focus is on implicit feedback.

Analyst feedback can be either explicit or implicit. An explicit feedback is an interaction of the analyst with the visualization layer by clicking a point, selecting a region, or asking for more details about a point. With this explicit feedback, the system mines analyst's preferences and discovers similar options in consecutive iterations as "guidance" [**?**,**?**,**?**] However, it is often the case in spatial data analysis that analysts look at some regions of interest but do not provide an explicit feedback. Example 1 describes illustrates this case in practice.

Example 1. Benicio is planning to live in Paris for a season. He decides to rent a home-stay from Airbnb website[4]. He likes to discover the city, hence he is open to any type of lodging in any region with an interest to stay in the center of Paris.

[1] *http://www.tableau.com*
[2] *http://www.simile-widgets.org/exhibit/*
[3] *http://spotfire.tibco.com*
[4] *http://www.airbnb.com*

The system returns 1500 different locations. As he has no other preferences, an exhaustive investigation needs scanning each location independently which is nearly infeasible. While he is scanning few first options, he shows interest in the region of Trocadero (where the Eiffel tower is located) but he doesn't feel necessary to click a point there. An ideal system should capture this implicit feedback in order to short-list a small subset of 1500 locations that Benicio should consider as high priority.

In spatialtemporal data analysis, it is often the case that analysts look at some regions of interest but forget to provide an explicit feedback. To capture user preferences is necessary go beyond filling in attributes and data provided by the user. It is important to understand the user behaviour while using the application to infer about his possible preferences. [7] affirms that temporal change in spatial patterns are increasingly common in geographical analysis. This work explore an approach to the spatialtemporal analysis of polygons that are spatially distinct and experience discrete changes though time. It presents challenges considering changes of regions (polygons) during the time. Works like [4] and [3] present solutions for clustering spatialtemporal data. These solutions are relevant to define regions by each cluster that contains important informations for the user.

While we have already discussed different ways of implicit feedback capturing in our GeoGuide paper, we focus on "mouse trackin" in this work, as the most common way for analysts to interact with a map.

It is shown in [1] that mouse gestures have a strong correlation with "user engagement". Intuitively, a point receives a positive feedback if the cursor moves around it frequently. We consider privacy issues by tracking the mouse cursor for each user. We consider the mouse movements an implicit user feedback. In most spatial datasets, there is a profile page dedicated to each point. Examples are restaurant pages in Yelp and lodging pages in Airbnb. In this paper, we consider the amount of time that the user spends in a page as also an implicit feedback. For instance, if the user spends few minutes in a page for an French cuisine restaurant or in a page for an apartment with 2 room and garage, this counts as positive feedback for this type of restaurants.

Discovering patterns and provide tendencies in spatial data applications may improve insights for planning and decision making for smart city solutions. Many systems and datasets consider space information. In this way, find spatial preferences can offer interactive and guidance solutions. For example, when users look for a house or hotel to spend a season, they consider one or more regions of their preference. These regions are intrinsic to each user, or user group. However, when navigating the application, the user also considers regions that seem interesting, for different reasons, such as the priority of some tourist spot, restaurants, clubs, security, etc. Thus, capturing region preferences over time can help to guide the user to find better places.

Given a dataset of spatial points and from the mouse tracking movements by the user, our approach generates a set of highlighted regions based on its

preferences. Each region is related with a subset of highlighted points which are illustrated using visual variables such as size and color intensity. The regions are also highlighted.

The outline of the paper is as follows: Section 2 describes the data model definition for our proposal, in Section 4 we present algorithms to overlap regions over time. Section 5 shows some experiments and, finally, Section 6 presents some conclusions and future directions.

2 Data Model Definition

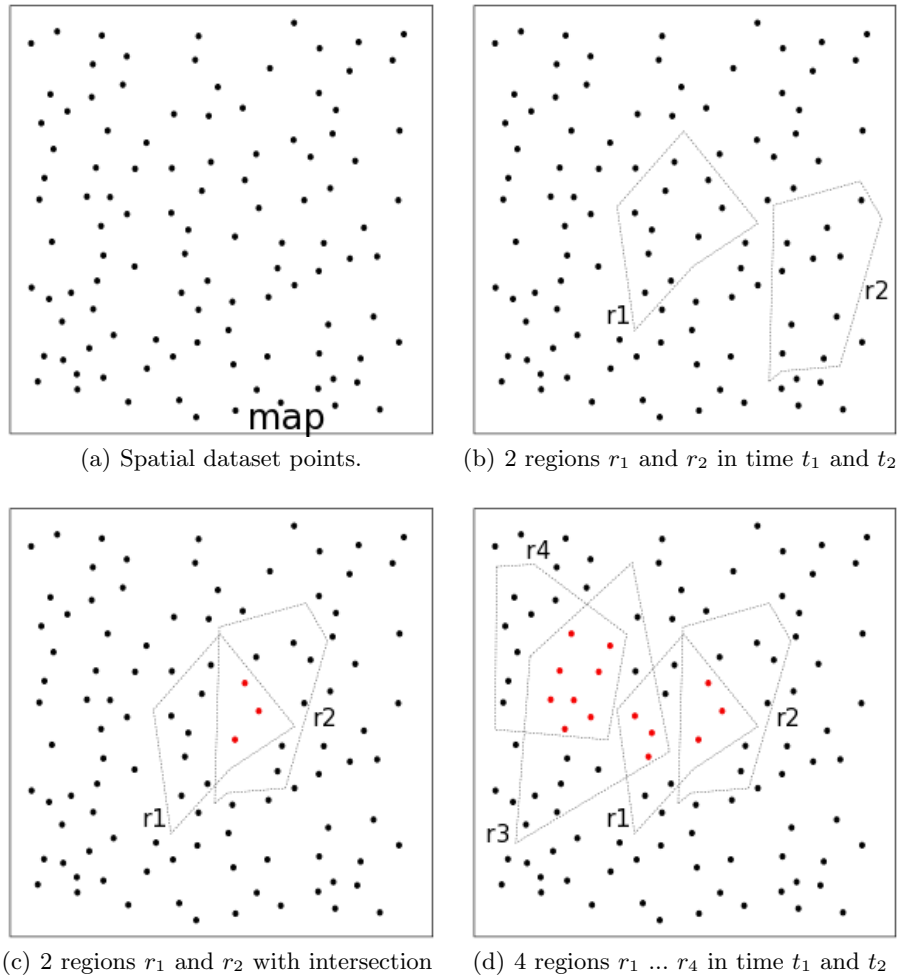
In our proposal we consider a spatial database $\mathcal{D}$ consisting $\langle \mathcal{R}, \mathcal{P}, \mathcal{A}, \rangle$ where $\mathcal{R}$ is the set of regions, $\mathcal{P}$ is the set of geographical points contained in the regions, and $\mathcal{A}$ is the set of point attributes. We consider a tuple $\langle lat, lon \rangle$ where lat and lon denote p's geographical coordinates (latitude and longitude respectively). The set $\mathcal{A}_p$ contains attribute-values for p over the schema of $\mathcal{A}$. There is a relation $\mathcal{B}$ ($\mathcal{R}$, $\mathcal{P}$), where for each $r \in \mathcal{R}$ there is one or more $p \in \mathcal{P}$ associated. For instance, $r\mathcal{B}p_1$, $r\mathcal{B}p_2$, ..., $r\mathcal{B}p_n$.

Each region $r \in \mathcal{R}$ is defined in a time interval $t \in \mathcal{T}$. So, there is a relation $\mathcal{C}$ ($\mathcal{T}$, $\mathcal{R}$), where for each $t \in \mathcal{T}$ is related with one or more $r \in \mathcal{R}$. For instance, $t\mathcal{C}r_1$, $t\mathcal{C}r_2$, ..., $t\mathcal{C}r_n$.

Each region is defined by clustering mouse tracking set of points $mt \in \mathcal{M}$. The set of mouse tracking points is captured in a time interval $t \in \mathcal{T}$. From the set of mouse tracking points we use the ST-DBSCAN [3] algorithm to create the regions. The algorithm starts with a first point $mt \in \mathcal{M}$. The algorithm retrieves all points density-reachable from mt with respect to two distance metrics to define the similarity by a conjunction of two density tests. The first metric is used for spatial values to measure the closeness of two points geographically, while the second metric is used to measure the similarity of non-spatial values. If $mt \in \mathcal{M}$ is a core object, a cluster is formed. If $mt \in \mathcal{M}$ is a border object, no points are density-reachable from mt and the algorithm visits the next point. The process is repeated until all of the points have been processed. So, from the mouse tracking points over time, it is possible to infer about the set of regions.

Points from mouse tracking are captured every 200 milliseconds, for example, or by another time interval $t \in \mathcal{T}$ defined by the analyst. At each time interval t a new set of points is created. This tracking is done to check possible user regions preferences. For each time interval t, the ST-DBSCAN algorithm is executed for the definition of clustered points. Each cluster will represent a region, through the limites of the mouse points captured by the mouse. The region is defined from the execution of Quickhull [2] algorithm. After x times t, the intersection between regions is calculated, where x is the number of times the mouse points will be captured.

The figures in1 show an example of regions and their intersections. The points shown in the figure are spatial dataset points (not mouse tracking points), which are highlighted from the regions captured by the mouse tracking. In figure 1(a), it shows the points of the arranged dataset. In figure 1(b) it shows 2 regions (r_1



(a) Spatial dataset points.
(b) 2 regions r_1 and r_2 in time t_1 and t_2
(c) 2 regions r_1 and r_2 with intersection
(d) 4 regions r_1 ... r_4 in time t_1 and t_2

Fig. 1. Definition and intersection of regions

and r_2) defined from the mouse tracking at a time t_1 and t_2 where $t_1 \prec t_2$ and that there is no intersection between them. Figure 1(c) also shows 2 regions (r_1 and r_2) defined from mouse tracking at a time t_1 and t_2 where $t_1 \prec t_2$, and where there is an intersection between them. The intersection between these regions highlights 3 points from the spatial dataset. Figure 1(d) shows 4 regions (r_1, r_2, r_3 and r_4) defined from mouse tracking at a time t_1 and t_2 where $t_1 \prec t_2$. Regions r_1 and r_4 were captured at time t_1, and regions r_2 and r_3 at time t_2. After capturing the regions, we notice that there are 3 intersections, being the intersections $\{r_1$ and $r_2\}$,$\{r_1$ and $r_3\}$ and $\{r_3$ and $r_4\}$. For each intersection we have 3, 3 and 8 points highlighted in the dataset, respectively.

3 Problem Definition

Challenge: 1. It not clear how implicit feedback (i.e., mouse moves) can contribute to the understanding of the system for analysts preferences. 2. Considering each mouse movement as a positive feedback drastically increases the amount of false positives.

Our intuition is that its not each single point which matters in feedback but the regions of interest and their density.

Hence we propose an algorithm to form polygons out of mouse overs and use the intersections between polygons to measure the density of regions. The density then counts as the implicit positive feedback.

4 Algorithm

We need to have a figure which summarizes the main steps of our approach.

This section presents some definitions and concepts in order to describe (*i*) how the polygons are created,(*ii*) in which delay of time the set of polygons are grouped, (*iii*) why and how the polygons are overlapped and (*iv*) how the regions of preferences over time are defined.

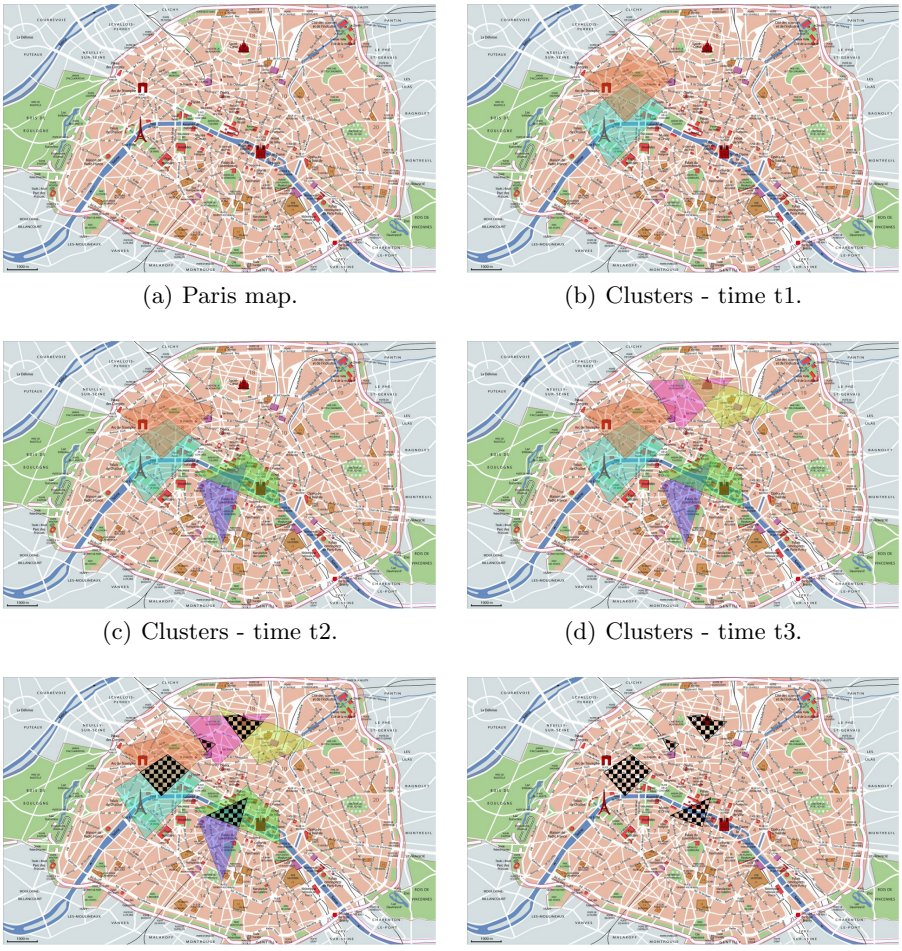
Figure 2 presents a sequence of maps of Paris, showing a possible evolution of how can be captured preference regions from *Benício* search.

Figure 2(a) shows the map before the to use by *Benício*. The Figure 2(b) shows 2 regions where the user searched for locations in a time interval t_1 (lets consider n seconds as a time interval, and $n < 60$). These two regions intersect each other. The figure 2(c) presents other 2 regions where apartments were also searched at a time interval t_2, where $t_1 \prec t_2$. In the same way, figure 2(d) presents 2 more regions where apartments were searched in a time interval t_3, where $t_1 \prec t_2 \prec t_3$.

Figures 2(e) and 2(f) highlight the intercessions of all regions. These intercessions show that these regions have been trafficked by the analyst more than once. It can inform a certain preference in these regions. These regions can be captured from mouse tracking or even by gaze techniques. Recognizing highlighted regions can help the analyst to consider one more parameter in the query to identify points in a particular dataset that may be more in line with their needs.

Considering a preference order of regions for *Benício*, the possible areas for rent would be: (*i*) the intersection of the regions ($r_1 \cap r_2 \cap r_n$), (*i*) the union of the regions minus their intersections (($r_1 \cup r_2 \cup r_n$) - ($r_1 \cap r_2 \cap r_n$)), and finally (*iii*) the entire dataset.

The Algorithm 1 goal is to identify which regions are intersect considering the clusters identified during the analyst search. The inputs are a set o regions $\mathcal{R}$ and a time interval Δt. The list of preferred regions is defined in line 1. For each region $r_n \in \mathcal{R}$ (line 2), the algorithm verifies if there are intersections with



(a) Paris map. (b) Clusters - time t1.

(c) Clusters - time t2. (d) Clusters - time t3.

(e) Intersection of regions. (f) Highlighted regions.

Fig. 2. Highlighting preference of regions from mouse tracking.

Algorithm 1: REGION HIGHLIGHTER Algorithm

Input: $\mathcal{R} = \{ r_1, r_2, ..., r_n \}$, $\Delta t \in T$

```
1 R_p ← ∅
2 for (r_n ∈ R) do
3 |   for (r_{n+1} ∈ R) do
4 |   |   if (r_n ∩ r_{n+1} ≠ ∅ ∧ get_time(r_n) ⊆ t ∧ get_time(r_{n+1}) ⊆ t then
5 |   |   |   R_p ← (r_n ∩ r_{n+1}) ∧ R_p
6 |   |   end
7 |   end
8 end
9 return R_p
```

a region $r_{n+1} \in \mathcal{R}$ (line 4). If there is a intersection and the region time is in the time interval, the intersection $r_n \cap r_{n+1}$ is included in R_p.

Algorithm 2 modifies the original HIGHLIGHTER proposal presented in GeoGuide [6] approach . The original begins by retrieving the most relevant points to p by simply retrieving the k highest ranking points in $\mathcal{L}_p$ (line 1) and function $get\_next(\mathcal{L}_p)$ (Line 2) returns the next point p_{next} in $\mathcal{L}_p$ in sequential order. Line 3 initialize the set of points that will be retrieved by the highlighted regions. At the beginning we consider that there is no preferred regions. So, the sets $\mathcal{S}_{rp}$ and $\mathcal{S}_p$ are the same. Lines 4 to 18 iterate over the inverted indexes to determine if other points should be considered to increase diversity while staying within the time limit and not violating the relevance threshold with the selected point.

The algorithm then looks for a candidate point $p_{current} \in \mathcal{S}_p$ to replace in order to increase diversity. If the candidate point is presented in a region r (line 8), the point is included in $\mathcal{S}_{rp}$, considering the boolean function *diversity_improved*() (line 6). This function checks if by replacing $p_{current}$ by p_{next} in $\mathcal{S}_p$, the overall diversity of the new $\mathcal{S}_p$ increases. If the point is not in the preferred region, it is included into $\mathcal{S}_p$.

5 Experiments

6 Conclusion

References

1. I. Arapakis, M. Lalmas, and G. Valkanas. Understanding within-content engagement through pattern analysis of mouse gestures. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, pages 1439–1448, New York, NY, USA, 2014. ACM.
2. C. B. Barber, D. P. Dobkin, and H. Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Trans. Math. Softw.*, 22(4):469–483, Dec. 1996.
3. D. Birant and A. Kut. St-dbscan: An algorithm for clustering spatial-temporal data. *Data Knowl. Eng.*, 60(1):208–221, Jan. 2007.

Algorithm 2: GeoGuide [6] + Region Highlighter Algorithm

Input: $p \in \mathcal{P}$, σ, k, *tlimit*, $R_p = \{ r_{p1}, r_{p2}, ..., r_{pn} \}$

$\mathcal{S}_p \leftarrow get\_top\_k(\mathcal{L}^p)$
$p_{next} \leftarrow get\_next(\mathcal{L}^p)$
$\mathcal{S}_{rp} \leftarrow \mathcal{S}_p$
while ($tlimit$ *not exceeded* $\wedge\ relevance(p, p_{next}) \geq \sigma$) **do**
  for $p_{current} \in \mathcal{S}_p$ **do**
    if $diversity\_improved(\mathcal{S}_p, \mathcal{S}_{rp}, p_{next}, p_{current})$ **then**
      for $r_{current} \in R_p$ **do**
        if $p_{current} \in r_{current}$ **then**
          $\mathcal{S}_{rp} \leftarrow replace(\mathcal{S}_{rp}, p_{next}, p_{current})$
          break
        end
      end
      $\mathcal{S}_p \leftarrow replace(\mathcal{S}_p, p_{next}, p_{current})$
      break
    end
  end
  $p_{next} \leftarrow get\_next(\mathcal{L}_p)$
end
return $\mathcal{S}_{rp} \cup \mathcal{S}_p$

4. M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 226–231. AAAI Press, 1996.
5. L. Lins, J. T. Klosowski, and C. Scheidegger. Nanocubes for real-time exploration of spatiotemporal datasets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2456–2465, 2013.
6. B. Omidvar-Tehrani, P. A. Souza-Neto, F. M. F. Pontes, and F. Bento. Geoguide: An interactive guidance approach for spatial data. In *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 1112–1117, June 2017.
7. C. Robertson, T. A. Nelson, B. Boots, and M. A. Wulder. Stamp: spatial–temporal analysis of moving polygons. *Journal of Geographical Systems*, 9(3):207–227, Sep 2007.
8. J. F. Roddick, M. J. Egenhofer, E. G. Hoel, D. Papadias, and B. Salzberg. Spatial, temporal and spatio-temporal databases - hot issues and directions for phd research. *SIGMOD Record*, 33(2):126–131, 2004.
9. A. Telang, D. Padmanabhan, and P. Deshpande. Spatio-temporal indexing: Current scenario, challenges and approaches. In *Proceedings of the 18th International Conference on Management of Data*, COMAD '12, pages 9–11, Mumbai, India, India, 2012. Computer Society of India.