# Understanding Hotspots: A Topological Visual Analytics Approach

Jonas Lukasczyk
TU Kaiserslautern
jl@jluk.de

Ross Maciejewski
Arizona State University
rmacieje@asu.edu

Christoph Garth
TU Kaiserslautern
garth@cs.uni-kl.de

Hans Hagen
TU Kaiserslautern
hagen@cs.uni-kl.de

## ABSTRACT

Analysis of spatio-temporal event data is of central importance in many domains of science and policy making. Current visualization methods rely on animation, small multiples, and space-time cubes to enable spatio-temporal data exploration. These methods require the user to remember state spaces or deal with layout occlusions when exploring their data. To overcome such issues, we propose a novel visualization technique for such data that applies the topological notion of Reeb graphs to identify hotspots as areas of relatively high event density within kernel density estimates. We illustrate that the topological identification of hotspots proposed in this paper is able to elucidate lifetime, properties, and relationships of hotspots by visualizing their temporal evolution based on the spatio-temporal Reeb graph. To validate our approach, we demonstrate our method on an epidemiological and a crime dataset. The resulting visualizations assist users in quickly identifying and comprehending important dates, events, hotspot properties, and relationships between hotspots.

## Categories and Subject Descriptors

G.3. [**Mathematics of Computing**]: Probability and Statistics—*Statistical computing*; J.2 [**Computer Applications**]: Physical Sciences and Engineering—*Mathematics and Statistics*

## Keywords

Spatio-Temporal Event Data, Hotspots, Geovisualization, Density Estimation, Topology, Reeb Graph

## 1. INTRODUCTION

Spatio-temporal datasets are common measurements and appear in many research fields. In general, a spatio-temporal dataset is a finite collection of points, where each point $e_i = (s_i, t_i)$ represents the location $s_i \in \mathbb{R}^2$ and time $t \in \mathbb{R}$ of an event $e_i$. For instance, Geoscience researchers often work with measurements at latitude-longitude points over time. Common examples of such datasets are occurrences of diseases, fires, earthquakes, tsunamis, robberies, and emergency calls. Given such a dataset, analysts are interested in information hidden within the data, e.g., where are areas with a high rate of events, how do those areas behave over time, and what is the relationship between them? Such areas are commonly referred to as hotspots, and their properties are essential for understanding spatio-temporal event data. Spatio-temporal analysis becomes even more difficult if the datasets are a large collection of events and contain a complex evolution of hotspots. In those cases, visualizing the data points directly is prone to change blindness, ambiguity, and occlusion [13, 31]. Therefore, a common approach is to visualize an aggregation of the events, such as a spatio-temporal density estimate [22, 27]. Yet, to understand the evolution of hotspots within these estimates, analysts have to manually compare density images for different time steps without additional visual support, e.g., analysts have to repeatedly watch animations until they are convinced that they have recognized all of the important information. However, if the animation contains a huge amount of information and if the analysts have to keep track of several areas simultaneously, then it is likely that the analysts may miss or misinterpret important information. Thus, the repeated examination of animations can be exhausting, unreliable, and leads to a high cognitive load [14, 17].

In this work, we present a method for extracting and visualizing the spatio-temporal trajectories and relationships of hotspots. While there has been much recent work on trajectory analysis, clustering and visualization, e.g. [1, 30], our work differs in that we are not explicitly working with trajectory or origin-destination data. Instead, we are calculating a spatio-temporal density estimate of the input data, and, subsequently, we are utilizing topological methods to understand the underlying relationship between hotspots and their movements. For example, emergency room records are routinely collected and such records may contain an underlying notion of disease spread, but the records themselves have no explicit definition of movements. Instead, each record has only a patient's address, time of visit, and visit reason. Looking at some measure of movement within the spatio-temporal events could provide analysts with insights into various spread patterns. We visualize these movements by depicting the evolution of areas within the density estimate with a high event rate, which we will further refer to as hotspots. Although this hotspot definition does not consider additional information, such as which infected persons

*SIGSPATIAL'15,* November 03 - 06, 2015, Bellevue, WA, USA
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3967-4/15/11...$15.00.
DOI: http://dx.doi.org/10.1145/2820783.2820817

were in direct contact with each other, our method is capable of visualizing the underlying pattern of disease spread that is contained in the density estimate. Such spread patterns are not limited only to the emergency room events. We can expand this idea to any spatio-temporal event data, such as criminal incident reports, economics, and social trends.

In order to find hotspots within the statistical data, both spatial and temporal patterns need to be explored during the analysis phase. Our technique visualizes trajectories and relations of hotspots within event-based data sources by deriving a spatio-temporal Reeb graph. For an introduction to the notion of Reeb graphs we refer the reader to Edelsbrunner et al. [11]. In general, Reeb graphs are used to illustrate the evolution of level sets of a real-valued function on a manifold. In the case of our approach, the manifold is a subset of the spatio-temporal density estimate, and the level sets are the extracted hotspots. These graphs enable analysts to quickly identify important topological events, such as when hotspots appear, disappear, merge, or split. As a first step, our technique approximates the underlying data distribution over time through the application of a spatio-temporal kernel density estimator. This provides us with a continuous functional representation of the data. Next, we extract and track areas within the estimates with a density value above a user specified threshold. Then, our method derives topological features of these extracted hotspots and visualizes them as spatio-temporal Reeb graphs. Furthermore, the thickness and location of edges encode the size and location of their associated hotspots, respectively. We developed a web-based visualization software to demonstrate the applicability of our method. The density estimates and graphs are calculated on the fly and users are able to interactively adjust algorithm parameters, such as the threshold value and kernel bandwidths. In this way, we can explore the spread patterns of spatio-temporal event data.

The major contributions of our paper are as follows:

1) The algorithmic extraction of hotspot trajectories and hotspot evolution within event-based datasets
2) The visualization of those features by Reeb graphs
3) An automatic Minard inspired flow map generation

## 2. RELATED WORK

Since spatio-temporal datasets are common measurements and appear in many research fields, several techniques have been proposed which derive and illustrate properties of those datasets. Typically, analysts have to manually explore temporal features of spatio-temporal data by either evaluating abstract mathematical properties, repeatably observing animations, or examining and comparing time-dependent color-coded static images, which are likely to be cluttered and ambiguous as the datasets become larger. This work proposes a new method to analyze and visualize features of spatio-temporal data by combining spatio-temporal kernel density estimations and topological based methods for time-varying scalar functions.

### 2.1 Kernel Density Estimations

One important feature of spatio-temporal datasets are areas with a high rate of events, commonly referred to as hotspots. Specifically, analysts want to explore how those areas behave over time and what their relationship is between each other. The works of Malik et al. [22], Peters et al. [27], and Maciejewski et al. [21] describe the problem and present solutions of tracking and visualizing hotspots inside spatio-temporal data. A common approach for this task is to use a kernel density estimate of the observed point patterns. Such a density estimate extrapolates from single point events to the entire spatial domain by blurring the points in space and time, thus yielding a scalar field over the spatial domain. In contrast to simply plotting the observed points, visualizing the scalar fields with heatmaps and iso-contours makes it easy for analysts to quickly identify and examine areas of interest. For instance, in the work of Malik et al. [22] a visualization method is presented which uses such kernel density estimates of recorded incidents to highlight areas where future incidents may occur. It was shown that this approach supports decision makers in their task of gaining insight into the data and allocating resources. Also the methods proposed in Maciejewski et al. [21] deal with the problem of identifying and visualizing hotspots. Again, a density estimate was used to highlight areas with a high frequency of events. They propose a system which enables analysts to calculate the density estimates for different points in time, visualizes the estimates in linked views, and allows the users to search for hotspots in space and time. Once more, in the work of Peters et al. [27] a kernel density estimate was used to examine temporal trends of hotspots. They propose a technique which color codes the density estimate according to the temporal trend, i.e., projecting the temporal component of the dataset via colors onto the density estimate. This approach yields good results as long as the dataset has a relatively simple temporal trend, but the color coding fails as soon as clusters overlap in time. Furthermore, to extract topological features of hotspots, all presented approaches require the user to manually compare several static images for different time steps and recognize patterns without additional visual support.

### 2.2 Spatio-Temporal Visualizations

In recent years, many spatio-temporal geo-visualization techniques have been developed for exploratory data analysis. Many of them have been facilitated to inspire creative thinking and provide new insights into the previously unknown characteristics of original data [2, 4, 28]. Some of the techniques focus on data with spatial and temporal information [10, 16, 18]. In the case of spatio-temporal datasets, most tools display the events as markers at points in a three-dimensional space-time cube [13, 15], and offer the user various techniques to project the data onto the spatial and temporal domain, thus creating a two-dimensional representation. For instance, the toolkit *GeoTime* [10] is capable of projecting graphs, paths, and scatter plots inside the space-time cube onto the geographical map or the timeline. Although the space-time cube is a useful tool in geovisualizations [20], it is still hard for users to recognize the location and time of certain events. Moreover, it becomes more difficult to analyze the temporal trends for frequently occurring events. For example, Nakaya and Yano [25] use spatio-temporal kernel density estimations of crime events to render crime density by volume rendering techniques inside the space-cube. However, it is still a problem to comprehend temporal trends and volume rendering also introduces the problem of occlusion. In this work, we extract volumes inside a spatio-temporal density estimate and use topological based methods to represent those volumes as graphs, which in turn can be easily projected onto the spatial and temporal domain.

## 2.3 Topological Methods for Time-Varying Scalar Fields

Previous methods lack a visualization which supports analysts in keeping track of the relationship between hotspots over time. To track these topological features, this work proposes to visualize the evolution of hotspots within the time-dependent kernel density estimates using Reeb graphs. The concept of representing the relationship between specific areas of time-varying scalar fields through graphs has already been proposed in several topology related work [11, 19, 23, 26, 29, 34]. The work of Mascarenhas et al. [23], Samtaney et al. [29], and Ji et al. [19] propose iso-contour based visualization techniques to track changes of time-varying scalar fields. After extracting an iso-contour for two adjacent time steps, they present methods to examine the deviation of the contours and construct Reeb graphs accordingly. These visualizations have been proven to be useful tools in understanding real-valued space-time data from computational simulations of physical processes, especially since the graph structures make it easy for analysts to comprehend the evolution of level sets and detect important iso-values [11]. Similar methods can even be applied in higher dimensions as proposed in Doraiswamy et al. [35] and Weber et al. [34]. In both works the researches extract iso-volumes of four-dimensional temperature fields and subsequently construct Reeb graphs of the extracted iso-volumes. The resulting Reeb graphs, which they call tracking graphs, support users in tracking the evolution of burning regions in combustion simulations. These tracking graphs are similar to our timeline projections, but our graphs are augmented with additional information and are also projected onto the spatial domain to illustrate hotspot location, trajectory, and size, i.e., in contrast to tracking graphs, our graphs also enable geographical analysis. In our method, a hotspot is a subset of scalar field that exceeds a user specified threshold. This is comparable to the work of Doraiswamy et al. [9] in which such areas are detected clouds. They also propose a method to illustrate the trajectory of these areas by visualizing the optical flow between to time steps. On the other hand, our approach uses topological features of the density estimate to visualize hotspot trajectories by projecting the spatio-temporal Reeb graph onto the spatial domain. The approach of Doraiswamy et al. [8] also extracts areas with a high density from a kernel density estimate. In their case, the density estimate represents the availability of taxis at a spatial location during a fixed time period and uses Reeb graphs to visualize the topology of these areas. However, they derive Reeb graphs for a fixed point in time, thus visualizing only the spatial evolution of these areas, whereas our approach aims to represent the evolution of hotspots in space and time. Oesterling et al. [26] also apply topological analysis to density estimates of multidimensional point clouds. Specifically, they compute and visualize the nesting of clusters identified as density maxima. In contrast, our method aims to describe, extract, and depict the evolution and trajectories of density clusters (i.e. hotspots).

# 3. METHOD

The proposed method consists of two steps. First we calculate a spatio-temporal density estimate of the input data. Then we extract hotspots from this estimate and derive their topological features via the newly introduced spatio-temporal Reeb graphs. Finally, the results are visualized in our web-based *spatio-temporal analysis toolbox (STAT).*

## 3.1 Spatio-Temporal Kernel Density Estimation

In the case of spatio-temporal datasets, we propose the use of a multivariate kernel density estimation in space and time. In particular, the proposed approach performs separate density estimations for the spatial and temporal coordinates of sample points. The spatio-temporal kernel density estimate formula is given in Definition 1.

*Definition 1.* (Spatio-Temporal Kernel Density Estimate). The spatio-temporal kernel density estimate of a dataset $\{e_1, ..., e_n\}$ with $e_i = (\ (x_i, y_i)\ ,\ t_i\ ) \in \mathbb{R}^2 \times \mathbb{R}$ is given by

$$\lambda(x,y,t) = \frac{1}{n h_t h_s^2} \sum_{i=1}^{n} K_T\left(\frac{t_i - t}{h_t}\right) K_S\left(\frac{x_i - x}{h_s}, \frac{y_i - y}{h_s}\right), \quad (1)$$

where $h_t$ is the temporal bandwidth, $h_s$ is the spatial bandwidth, and $K_T$ and $K_S$ are the temporal and spatial kernel functions, respectively.

Definition 1 allows analysts to use different kernels for the spatial and temporal domain, i.e., Equation 1 can be interpreted as an ordinary multivariate spatial kernel density estimate, where the temporal kernel additionally calculates a weight for each sample point. The temporal bandwidth $h_t$ is thereby used to model already known properties of the dataset, e.g., if it is known that an infectious disease has an incubation period of approximately one month, then the likelihood of a new outbreak should only depend on the previous and upcoming thirty days. Nevertheless, it is a known problem of spatio-temporal kernel density estimators that the density estimation near the boundaries is not consistent. There are boundary correction methods available to correct for those boundary effects, but this is not the scope of this work.

For the case studies presented in this work, we use a one dimensional triangular kernel

$$K_T(u) = (1 - |u|)\mathbb{1}_{\{|u|<1\}} \quad (2)$$

as a temporal kernel function, where $\mathbb{1}$ is the indicator function. Since we do not intend to predict new outbreaks, but rather want to gain insight into collected historic data, we choose a symmetric temporal kernel. This incorporates the fact that the probability of an event occurring at location $s$ at time $t$ is also higher if events occurred in the near future of this location. However, one could also choose an asymmetric temporal kernel to explore the data from a predictive point of view [5]. In fact, Equation 1 allows analysts to use any kernels they think models the underlying scenario best. As for the spatial kernel function, we use a multivariate multiplicative Epanechnikov kernel:

$$K_S(u,v) = \frac{9}{16}(1-u^2)(1-v^2)\mathbb{1}_{\{\ |u|<1\ \wedge\ |v|<1\ \}}. \quad (3)$$

After the analyst has chosen a spatial bandwidth, temporal bandwidth, and appropriate kernels, Equation 1 is discretely evaluated on a grid over the spatial domain. Thus, each evaluation of Equation 1 for a constant point in time $t$ over the gird points yields a two dimensional matrix $\lambda_t$. To allow analysts to explore the effect of different kernels and bandwidths on the fly, we implemented the density estimation on the GPU through a custom WebGL shader. Specifically, the kernel parameters are passed to the shader as uniforms, and the dataset as a floating point texture. Therefore, the number of points is limited by the largest supported texture size, which is for most devices at least a resolution of $2048^2$ points. Our current implementation supports a resolution of $4096^2$ points up to 1000 time steps over a grid of $128^2$ cells and is still able to achieve interactive frame rates due to various optimizations.

## 3.2 Spatio-Temporal Reeb Graph Derivation

This work proposes the use of Reeb graphs to represent the topological relationship between hotspots within the spatio-temporal denstiy estimate. Hence, this section formally defines hotspots and describes a construction algorithm for spatio-temporal Reeb graphs.

In this work, hotspots are subsets of the domain of the spatio-temporal density estimate $\lambda : S \times T \to \mathbb{R}$ with a density value above a user specified threshold $\tau$ (see Definition 2). Each connected component of such a subset $D$ is a single hotspot, i.e., two points belong to the same hotspot if and only if both points are contained in the same connected component of $D$. The choice of $\tau$ thereby influences the significance of hotspots. For instance, a value of $\tau$ near $\max_{s\in S,t\in T} \lambda(s,t)$ would only extract areas with a very high density. The proposed approach aims to visualize the evolution of those hotpots given by the volume $D \subset S \times T$. Therefore, we define an auxiliary function $\phi : D \to T$ which returns the temporal coordinate of a point inside the volume. In other words, $\phi$ is the height function of this volume, where the temporal coordinate of the volume is interpreted as its height. It is possible to extract for $t \in T$ level sets $\phi^{-1}(t)$ of $D$, and hence an ordinary Reeb graph can be constructed that represents the topology of the evolution of these level sets (see Definition 3).

*Definition 2.* (Hotspot). Let $D$ be a subset of the domain of the spatio-temporal density estimate $\lambda : S \times T \to \mathbb{R}$ such that $(s,t) \in D$ iff $\lambda(s,t) \geq \tau$ for some fixed value $\tau \in [0, \max_{s\in S,t\in T} \lambda(s,t)]$. Then each connected component of $D$ is called a hotspot of $\lambda$.

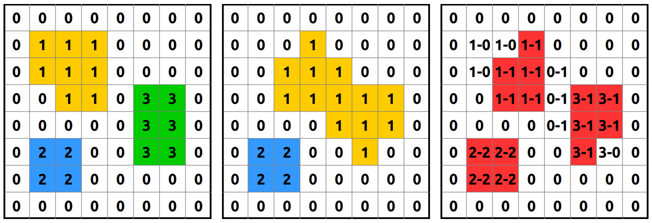
*Definition 3.* (Spatio-Temporal Reeb Graph). For some value $\tau$, let the subset $D$ be given as in Definition 2, and the function $\phi : D \to T$ be given as $\phi(s,t) = t$. Then the Reeb graph for $\phi$ is called the spatio-temporal Reeb graph.

However, in most cases the spatio-temporal density estimate $\lambda$ is evaluated on a grid for several points in time, and therefore the estimate is given as a three-dimensional lookup table $\hat{\lambda}[x,y,t]$. In this case, $D$ is the set of all cells with $\hat{\lambda}[x,y,t] \geq \tau$, which can be interpreted as a voxel volume. Algorithm 1 is capable of extracting hotspots from such a spatio-temporal lookup table. The algorithm requires four input parameters: the discretized spatio-temporal density estimate $\hat{\lambda}$, the temporal boundary points $t_0$ and $t_n$, and the threshold value $\tau$ for each cell to be part of a hotspot (see Definition 2). The algorithm constructs an approximation of the spatio-temporal Reeb graph in a bottom-up approach. In particular, it uses a data structure called the level set matrix (LSM), which is a regular matrix of the same dimension as the spatial grid of the discretized density estimate. An entry of this matrix for time $t$ is either 0 if the corresponding cell is not part of the voxel volume $D$, or the value is the unique identifier (ID) of the component the corresponding cell is part of. The procedure *calculateLSM* computes, for a given time slice of the spatio-temporal density estimate, the LSM for this slice. The algorithm initializes the LSM with zero and, subsequently, all cells with a value larger than $\tau$ are marked by $-1$. So far, a binary partition according to $\tau$ has been obtained, but it is not possible to differentiate between connected components. Thus, for each cell with value $-1$ a common *flood fill* algorithm is used to assign to each 8-adjacent cell with the same value a unique ID. After all *flood fill* procedures have been performed, each connected component has a unique ID for this time slice. Figure 1a and 1b display two exemplary LSMs for adjacent time steps, where the first matrix contains three components and the second matrix two components.

**Algorithm 1** Calcualte Spatio-Temporal Reeb Graph

```
 1: procedure SpatioTemporalReebGraph($\hat{\lambda}, t_0, t_n, \tau$)
 2:     // initialize sets of vertices, edges, and components
 3:     $V = \emptyset,\ E = \emptyset,\ C = \emptyset$
 4:     // $M_1$ is the LSM of the previous time step
 5:     $M_1 = null$
 6:     // $M_2$ is the LSM of the initial time step
 7:     $M_2 = calculateLSM(\hat{\lambda}[:,:,t_0], \tau)$
 8:     // create nodes for initial LSM
 9:     $createNodes(V, M_2)$
10:     // for each subsequent time step...
11:     for ($t = t_0 + 1 : t_n$) do
12:         // $M_1$ becomes LSM of last time step
13:         $M_1 = M_2$
14:         // $M_2$ is LSM of the current time step
15:         $M_2 = calculateLSM(\hat{\lambda}[:,:,t], \tau)$
16:         // create nodes for $M_2$
17:         $createNodes(V, M_2)$
18:         // connect nodes to preceding nodes
19:         $connectNodes(V, E, M_1, M_2)$
20:     end for
21:     // mark critical nodes of the graph $(V, E)$
22:     $markCriticalNodes(V, E)$
23:     // compute components of the graph $(V, E)$
24:     $C = computeComponents(V, E)$
25:     return$(V, E, C)$
26: end procedure
```

Algorithm 1 initializes in line 3 the sets of vertices, edges, and components as empty sets. Since there is no previous



|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 3 | 3 | 0 |
| 0 | 0 | 0 | 0 | 0 | 3 | 3 | 0 |
| 0 | 2 | 2 | 0 | 0 | 3 | 3 | 0 |
| 0 | 2 | 2 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

(a)

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 2 | 2 | 0 | 0 | 1 | 0 | 0 |
| 0 | 2 | 2 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

(b)

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1-0 | 1-0 | 1-1 | 0 | 0 | 0 | 0 |
| 0 | 1-0 | 1-1 | 1-1 | 0-1 | 0 | 0 | 0 |
| 0 | 0 | 1-1 | 1-1 | 0-1 | 3-1 | 3-1 | 0 |
| 0 | 0 | 0 | 0 | 0-1 | 3-1 | 3-1 | 0 |
| 0 | 2-2 | 2-2 | 0 | 0 | 3-1 | 3-0 | 0 |
| 0 | 2-2 | 2-2 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

(c)

Figure 1: Example level set matrices for two time steps: (a) $t_1$ and (b) $t_2$. (c) shows the overlap of the level set matrices $t_1$ and $t_2$ marked in red.

time step, the LSM of the previous time step is set to null in line 5. Then the LSM of the initial time step is calculated in line 7 with the previously described procedure *calculateLSM*. Afterwards, the procedure *createNodes* creates, for each connected component of the LSM, a new node located at the center of mass of that component. Each node also stores the size of its associated component. Subsequently, for each following time step, an LSM and new nodes are calculated in line 15 and 17, respectively. The procedure *connectNodes* in line 19 is used to test the level set matrices of the current and previous time step for overlaps of their connected components, which is similar to the methods proposed in G. Ji et al. [19] and D. Silver et al. [32]. Although more sophisticated approaches could be used, the simple test for overlaps suffices to demonstrate the concept. If two components overlap, then a new edge between their associated nodes is created. Both IDs just have to be unequal to zero to qualify for an overlap, hence the identifiers themselves can be different. In the example of Figure 1, the components of time slice $t_1$ with ID 1 and 3 overlap with the component of time slice $t_2$ with ID 1. Hence, the nodes associated with component 1 and 3 of time slice $t_1$ are connected to the node associated to the component with ID 1 of time slice $t_2$. Also the component with ID 2 of time slice $t_1$ has an overlap with component 2 of time slice $t_2$, so their associated nodes are connected as well. After the for loop from line 11 to 20, the intermediate graph $(V, E)$ does not fulfill the definition of a Reeb graph since there also exist nodes at uncritical points. Thus, in line 22 the method *markCriticalNodes* marks those nodes which have no predecessor, no successor, or more than one predecessor or successor. As a final step, the procedure *computeComponents* parses the graph $(V, E)$ and groups all edges togehter that belong to a path that starts and ends at a critical node, and contains only non-critical nodes in between. Thus, each component represents an edge of the Reeb graph.

## 3.3 Spatio-Temporal Reeb Graph Visualization

To visualize the spatio-temporal Reeb graphs, we implemented the previously described theory in our web-based *Spatio-Temporal Analysis Toolbox* (*STAT*). Since the application is web-based, the visualization is easily accessible, cross-platform compatible, and does not require the installation of additional software or plug-ins. The input of the software is a CSV-File, which contains a list of latitude-longitude positions and dates. First, the software calculates a bounding box of the spatial locations and normalizes them according to the bounding box such that all points are contained in the unit-square. Then, the software uses the method described in Section 3.1 to calculate a spatio-temporal density estimate of the normalized data, where the user can select the spatial bandwidth, temporal bandwidth, and the used kernels in real-time. Afterwards, users can also vary the threshold $\tau$ at interactive frame rates to extract hotspots from the density estimate and derive spatio-temporal Reeb graphs as described in Section 3.2. The output of the proposed method is highly sensitive to the selected threshold. In fact, there are no $\tau$ values that suit every dataset, thus, the analyst has to vary the threshold to gain insight into the data by pealing through the density estimate. On the other hand, this enables analysts to use the threshold as a tool to explore the data from different angles, e.g., high thresholds would filter out insignificant hotspots, while lower thresholds bundle hotspots which are close to each other in the spatial domain. We allow users to adjust the threshold and bandwidths in real time, which enables them to analyze hotspot properties and nesting. To further support the exploration of different thresholds, we also display the density maxima of the extracted hotspots by a simple line chart. This chart supports the user in comparing and filtering the extracted hotspots.

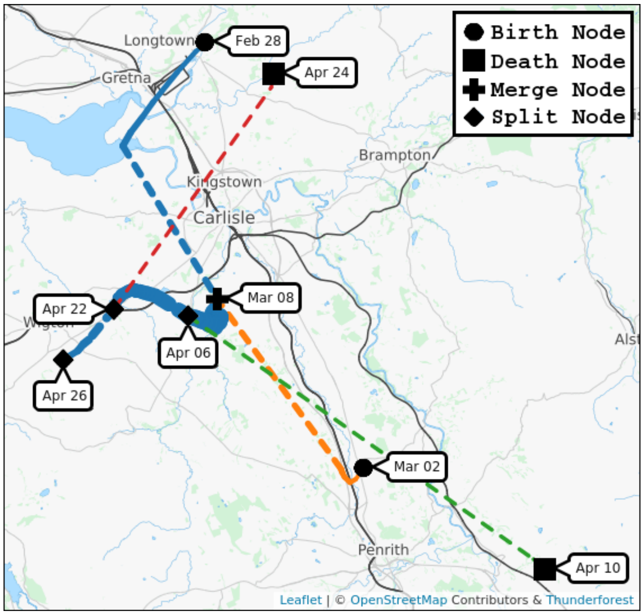


Figure 2: Map projection of the spatio-temporal Reeb graph for $\tau = 0.12max(\lambda)$ of the FMD dataset until April 26. Solid lines indicate continues movement of hotspots, and dashed lines merge/split events of hotspots.

We visualize the spatio-temporal Reeb graphs in two ways: as a projection onto the map, and; as a projection onto the timeline. Critical nodes of the graphs are displayed by different symbols, i.e., birth nodes are visualized by disks, death nodes with squares, merge nodes with crosses, and split nodes with diamonds as shown in Figure 2. Besides the graph connectivity, which represents the topological features of the hotspots, the thickness of an edge at time $t$ encodes the size of the associated hotspot at this time. Furthermore, each edge has the color of its associated hotspot. If a new hotspot appears, i.e., an edge starts with a birth node, then it gets a new color. When hotspots merge, then the resulting hotspot inherits the color of the largest merging hotspot. If a hotspot splits, then the largest new hotspot inherits the color of the preceding hotspot, and all the other hotspots get new colors.

To create a Minard inspired flow map visualization [24], we project the spatio-temporal Reeb graph onto a map. In particular, we use the *Leaflet Javascript API* to generate the map in the background, which provides geo-spatial context. For the projection, we simply discard the temporal coordinate of the graph and add a date caption at each critical node. The solid lines of the graph represent movement of their associated hotspots, while the dashed lines indicate discontinuities of the hotspot centers (see Figure 2). For instance, consider the green hotspot of Figure 3b that splits up from the orange hotspot of Figure 3a. In this particular case, the green hotspot did not move away from the orange hotspot, instead their areas are simply not connected anymore, causing the center of mass of the green hotspot to *jump* to a new location. These *jumps* can also occur during merge events, i.e., in this case the centers of mass of the merging hotspots *jump* to the center of the resulting hotspot (see Figure 3c-d). Therefore, we use solid and dashed lines to differentiate between these jumps and actual movement. To read the graph, users could start at a birth node and follow the edges until they reach another critical node. Reading the graph this way enables users to follow the history of a hotspot, which is similar to Minard's flow map visualization [24]. The map also contains a WebGL-canvas to render the density estimate, the extracted level sets, and the point data. The density estimate can be rendered by a user specified number of contour lines through a custom WebGL-shader, and the areas of the level sets are rendered in the color of their associated Reeb graph edge. Their boundaries are not rendered with a fixed width; instead, we draw a pseudo border by rendering the density with a value near the threshold value black. This has the additional advantage that we can visualize these borderline areas.

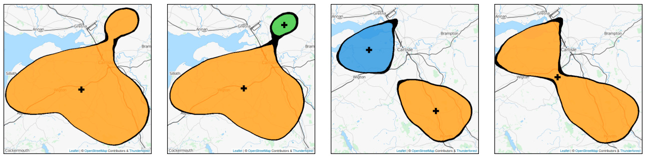

(a) (b) (c) (d)

Figure 3: Examples for jumps of hotspot centers during split events (a-b) and merge events (c-d). Black crosses indicate hotspot centers.

To support users in comprehending the temporal evolution of hotspots and navigating through time, we also project the spatio-temporal Reeb graph onto the timeline. Since the spatio-temporal Reeb graph is a part of the spatio-temporal domain, i.e., it has a latitude, longitude, and a time coordinate, it is possible to project the spatio-temporal Reeb graph onto the timeline by discarding one spatial dimension. We chose to discard the longitude dimension, so that the timeline and the map share the same y-domain, which makes it easier for users to compare the graphs between both projections (compare Figure 2 and 5a).

# 4. CASE STUDIES

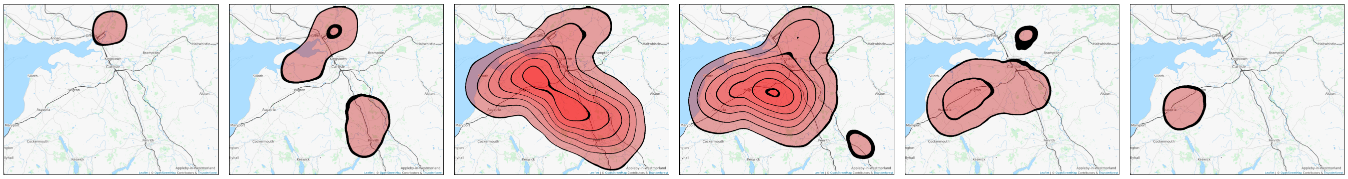
In this section, the application of the proposed method is demonstrated on two datasets. Section 4.1 analyzes an outbreak of *foot-and-mouth disease* (FMD) during the UK 2001 FMD epidemic, and Section 4.2 analyzes reported crimes in the City of Chicago (US) from 2013 to 2015.

## 4.1 Foot-and-Mouth Disease Dataset

The foot-and-mouth disease dataset is part of the $R$ package *stpp* [12] and consists of 648 reported outbreaks in the northern part of Cumbria county (UK). FMD is a highly infectious and severe viral disease of farm livestock that affects cloven-hoofed animals by causing fever and blisters on the feet and mouth. The disease can be transmitted by direct contact with infected animals or through airborne particles [33]. The UK 2001 FMD epidemic was one of the largest outbreaks in history and lasted from February till September. Besides the massive environmental fallout, the disease also had serious economical consequences, such as costs for countermeasures and compensations for farmers [6].

This case study focuses on the time period starting at the beginning of the outbreaks in end of February until the disease was brought under control in the end of April. To calculate the spatio-temporal density estimate $\lambda$ according to Section 3.1, one has to choose appropriate kernels and bandwidths $K_T$, $K_S$, $h_t$ and $h_s$. In this case study, we model the FMD outbreak according to official reports [6, 7], university reports [3], and guidelines [33]. FMD has an approximate incubation period between two and fourteen days [7, 33], therefore we set the temporal bandwidth to $t_h = 14\ days$ and use the triangular temporal kernel given by Equation 2. In order to quarantine infected farms, the US Department of Agriculture suggests in their FMD response plan to establish control areas with a radius of approximately $10km$ around the infected premises [33]. Thus, we use this radius as the spatial bandwidth, i.e., $h_s = 10km$. The spatial kernel used in this case study is a multivariate Epanechnikov kernel as given in Equation 3. Finally, the lattice on which Equation 1 is evaluated has a spatial resolution of $128 \times 128$ grid points for each day. The resulting density estimates for representative dates are shown in Figure 4.

Before we analyze the density estimate with our proposed technique, we evaluate the estimate by comparing it to statements made in official records and reports. Figure 4a shows that the first hotspot of the epidemic emerged north of Carlisle on February 28. In fact, the indicated area around Longtown is considered to be ground zero of the FMD epidemic in Cumbria, since many already infected animals have unwittingly been distributed through the Longtown cattle market, which is the largest sheep market in England [3, 6, 7]. This lead to outbreaks at close susceptible farms, which caused the first hotspot to grow. At the same time, a new hot spot emerged in the south east of Carlisle (see Figure 4b). Both hotspots grew further and merged during March (see Figure 4c). Beginning in April, the epidemic was slowly brought under control by effective and organized countermeasures such as vaccinations and culling of animals, causing the main infection to shrink and split into smaller hotspots until April 28 [6] (see Figure 4d-f).



(a) Feb 28 (b) Mar 5 (c) Mar 22 (d) Apr 08 (e) Apr 23 (f) Apr 28

Figure 4: Representative plots of the spatio-temporal kernel density estimate $\lambda$ of the FMD dataset.

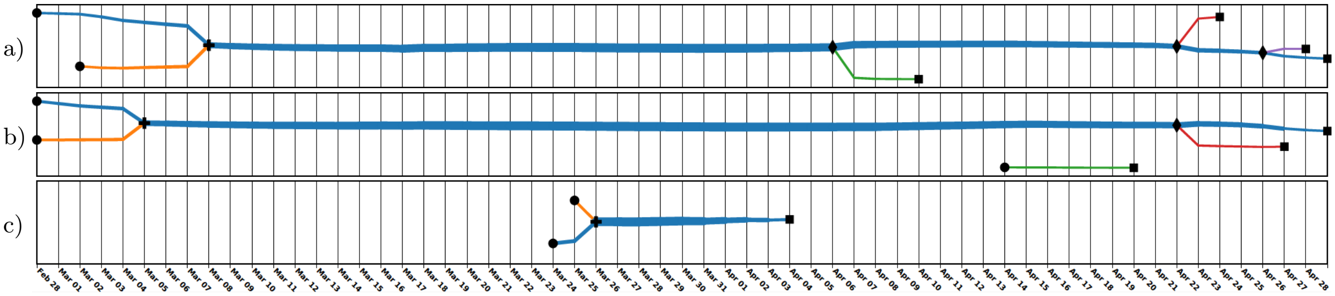


Figure 5: Projections of spatio-temporal Reeb graphs of the FMD dataset onto the timeline for different $\tau$ values: (a) $\tau = 0.12max(\lambda)$, (b) $\tau = 0.07max(\lambda)$, and (c) $\tau = 0.9max(\lambda)$.

The major contribution of this work is the spatio-temporal Reeb graph, which is capable of visualizing such stories algorithmically in a single figure. Thus, analysts are no longer required to manually compare static density images for different points in time. For instance, we can calculate the spatio-temporal Reeb graph for the threshold equal to the first contour line of Figure 4, i.e., $\tau = 0.12max(\lambda)$. Figure 2 and 5a show the projection of this graph onto the map and the timeline, respectively. The timeline projection makes it very easy to identify important days and to understand the underlying hotspot structure, while the spatial projection additionally provides geographic references. Note, both projections directly visualize the statements made in the previous paragraph. The graphs in both projections show that the first hotspot emerged on February 28, while the spatial projection also indicates the correct origin near Longtown. Then, a new hotspot emerged on the March 2, which merged with the first hotspot around March 8. The thickness of the lines also encode the size of the associated hotspots, e.g., Figure 5a shows that the blue hotspot reached its maximum at the end of March, coinciding with the official reports which state that the disease reached its peak around that time [3, 6]. On April 6, April 22, and April 26, small hotspots split from the main hotspot, indicated by the diamond symbols and the green, red, and purple lines, respectively. The square symbols mark the disappearance of these small hotspots on April 10, April 24, and April 27. However, Figure 4 does not show the existence of the purple hotspot due to the low temporal sampling rate. To compensate for that, analysts would have to choose a finer temporal resolution and thus manually compare a much higher number of density images. The spatio-temporal Reeb graph, on the other hand, is capable of extracting these events algorithmically and visualizing them in graphs, which are easy to read and contain all the important information in just one static image. This shows that the spatio-temporal Reeb graph is capable of telling the same story as the official reports.

As mentioned in Section 3.3, our method requires the user to vary the threshold to analyze the data. For example, a smaller $\tau$ value causes Algorithm 1 to additionally consider areas which did not exceed the previous threshold, but exceed the smaller one. This can be used for anomaly detection, e.g., Figure 5b shows the timeline projection for $\tau = 0.07max(\lambda)$. For this threshold, the graph consists of two trees, i.e., the main infection and a remote hotspot. Note, for the threshold $\tau = 0.12max(\lambda)$ the green hotspot is discarded for being considered insignificant. By choosing a low threshold, we smooth out features of the main hotspot, such as the small hotspots of Figure 5a, but we are now able to detect hotspots with lower density, such as the green hotspot of Figure 5b. Conversely, choosing a high threshold results in a graph that only represents intense hotspots, e.g., Figure 5c shows the timeline projection for $\tau = 0.9max(\lambda)$. This graph shows that the outbreak was most intense from March 24 through April 4, which is again consistent with the official reports [3, 6].

## 4.2 Chicago Crime Dataset

Next we analyze a crime dataset that is available on the City of Chicago's Data Portal website. Specifically, we investigate a collection of 1,680,393 reported incidents of thefts and robberies in Chicago from 1 January 2001 until 22 June 2015.

Similar to the first case study, we start by choosing appropriate parameters for Equation 1. Again, a triangular and a multivariate Epanechnikov kernel are used as a temporal and spatial kernel, respectively, and the used lattice has a resolution of 128 × 128 grid points for each day. To detect small temporal variances in the data, the temporal bandwidth has to be small enough not to smooth out those variances. Therefore, we use a temporal bandwidth of three days. To explore the spatial distribution of crimes, we vary the spatial bandwidth between three and seven kilometers.

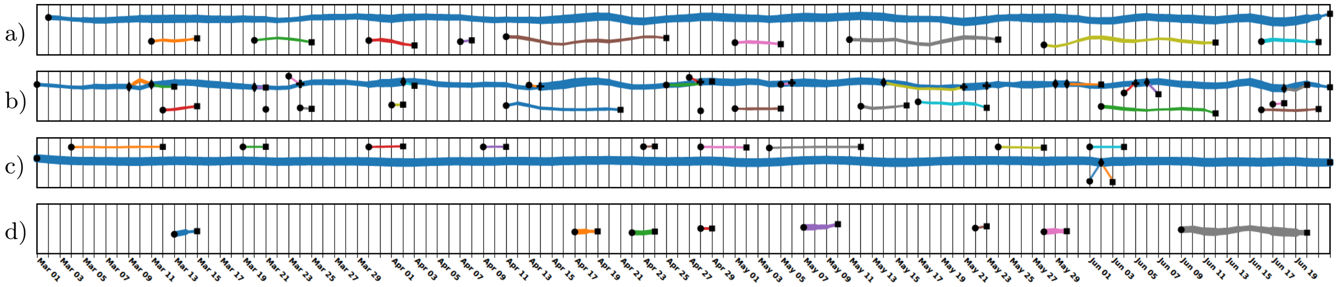
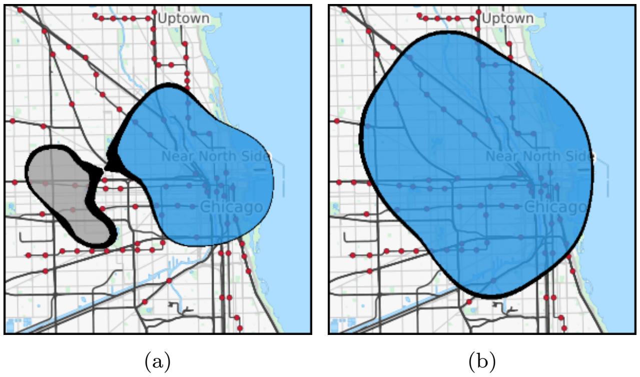


Figure 6: Timeline projections of the spatio-temporal Reeb graphs of the Chicago crime dataset in 2015 for different bandwidths and thresholds: (a) $(\tau, h_s) = (0.36max(\lambda),\ 7\,km)$, (b) $(\tau, h_s) = (0.36max(\lambda),\ 3\,km)$, (c) $(\tau, h_s) = (0.05max(\lambda),\ 3\,km)$, and (d) $(\tau, h_s) = (0.86max(\lambda),\ 3\,km)$, where $\lambda$ is the density estimate with $(h_t, h_s) = (3\,days,\ 3\,km)$.



(a)

(b)

Figure 7: Extracted hotspots of the Chicaco crime dataset at 19 June 2015 for (a) $h_s = 3\,km$ and (b) $h_s = 7\,km$.

Using a larger spatial bandwidth causes more smoothing, resulting in simple graphs which can be used to get an overview of the data. For instance, Figure 6a shows a timeline projection for $h_s = 7\,km$. The projection shows that there exists a permanent crime hotspot and another hotspot that exceeds the current selected density threshold only in intervals. To further examine the spatial distribution of these hotspots, one can compare these projections with the graphs for a lower spatial bandwidth, since a lower bandwidth represents the spatial distribution in more detail. By comparing 6a and 6b one can see that the hotspots of Figure 6a split into smaller components, e.g., at 19 June 2015 the permanent crime hotspot of Figure 6a consists of two separate hotspots as shown in Figure 6b and 7. Another example is the cyan hotspot from June 16 to June 21 of Figure 6a which consists for a lower bandwidth also of two components as shown in Figure 6b. Just like the spatial bandwidth can be used to explore the spatial distribution of hotspots, analysts can vary the temporal bandwidth to explore the temporal distribution of hotspots. Since there are no optimal bandwidths, it is crucial that analysts can modify bandwidths and observe the resulting density estimates in real-time, which is possible with our current implementation.

As shown in the previous paragraph, the bandwidths can be used to explore the distribution of hotspots in space and time. The threshold, however, can be used to to explore the intensity of hotspots. For example, choosing a low threshold bundles hotspots that are spatially close to each other. This can be used to identify hotspots which are located remotely to the bundle, e.g., the large blue hotspot of Figure 6c bundles all hotspots of Figure 6b and thus represents them in one line. At the same time, new hotspots are detected which are not connected to the bundle and therefore are located remotely to it. In fact, for $\tau = 0.05max(\lambda)$ it was possible to identify the O'Hare airport as a crime hotspot, where the timeline projection shows for which intervals the rate of crimes was above the threshold. The timeline projection also shows an anomaly between June 1 and June 3. In particular, on June 1 a hotspot appeared close to the bundle and grew until the next day, causing the hotspot to be considered as part of the bundle on June 2. On June 3 the hotspot starts to shrink, and therefore is separated from the bundle again. Figure 8 shows the hotspot bundle (blue), the airport hotspot (cyan), and the hotspot anomaly (orange). The map projection of the spatio-temporal Reeb graph does not contain solid lines, which represents the fact that the hotspots remain stationary. The dashed, orange line indicates that the orange hotspot was once considered part of the bundle until June 2.

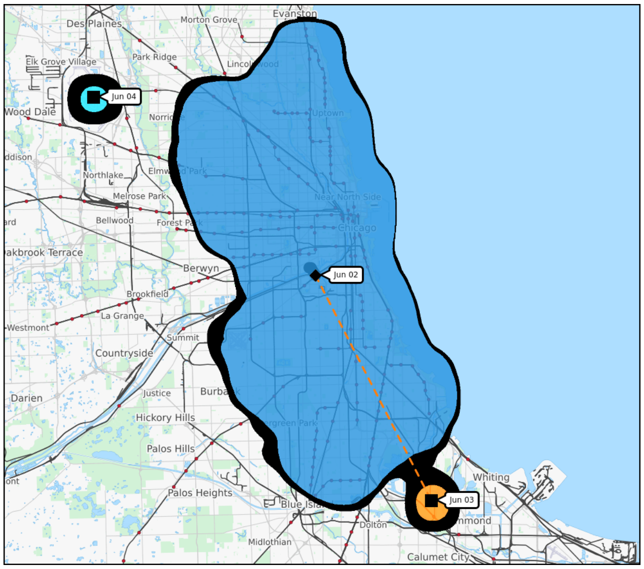


Figure 8: Map projection of the spatio-temporal Reeb graph and extracted hotspots for $(\tau, h_s) = (0.05max(\lambda), 3\,km)$ of the Chicago crime dataset at 3 June 2015.

If one selects a high threshold, most hotspots will be filtered out and only areas with a very high density are detected as hotspots. This can be used to search for time intervals and locations with a relatively high rate of crime. Figure 7d shows such time intervals for $\tau = 0.86max(\lambda)$. The spatial projections of these hotspots are all located around the city center, indicating that most of the crimes are committed in this area, which intuitively coincides with reality. Each year form 2001 to 2015 a hotspot is detected in the middle of March, which could be related to the city's St. Patrick's Day Parade. The timeline projections of all years also show that the crime rate is most intense from June to August. Again, this could be related to the increased number of festivals and other large events during summer.

The benefit of the interactive $\tau$ exploration is that the choice of bins for coloring maps is an extremely difficult task in spatio-temporal data. If an analyst changes the bin range each day, they lose coherence between the colors; however, if the bins are designed to match the entire range, extreme hotspots, as seen in this case, may be obfuscated by the binning technique. Given that the cartographic rule of thumb is 5-7 color intervals on a map, the use of an interactive threshold combined with the projected Reeb graph can improve the spatio-temporal analysis process.

## 5. CONCLUSION AND FUTURE WORK

This work proposes a novel technique for visualizing hotspots within spatio-temporal datasets. In particular, it was shown how hotspots can be extracted from spatio-temporal kernel density estimates, and how they can be used to derive spatio-temporal Reeb graphs. These graphs elucidate lifetime, properties, and relationships of hotspots by visualizing their temporal evolution in static images. Moreover, the graphs are easy to read, algorithmically computed, and visualize trajectory information of event-based datasets. We demonstrated the application of the method on two real datasets in our WebGL-based *Spatio-Temporal Analysis Toolbox*. With this software, users can explore the density estimates, the extracted hotspots, and the spatio-temporal Reeb graphs in real-time. In both case studies, the results are consistent with the ground truth, yield further insight into the data, enable reliable data extraction, and visualize information in static images that analysts before had to extract manually from animations. As such, the proposed method can be used by analysts as an additional tool to easily explore hotspots and their properties within event-based spatio-temporal datasets.

While we have not currently engaged domain experts, we believe that this work provides benefits not found in current geovisualization methods. As mentioned previously, current methods for spatio-temporal analysis rely on either small multiple, animations, or space time cubes for exploratory data analysis. Our visual analytics solution extracts hotspots and directly plots their growth and decay in a single image. While our approach cannot answer every question an analyst may ask, it does provide a succinct means of summarizing important characteristics in such datasets. As such, analysts can immediately answer questions about when, where, and for which threshold hotspots emerge, disappear, merge, or split, while in previous techniques this would require mentally combining small multiples, replaying animations, or rotating the space-time cube. By providing these methods with a suite of interactive linked views, we are able to support a variety of analytical processes.

In future work, we plan to extend our approach in several ways. First, we are going to explore a way to visualize the underlying hotspot hierarchy for varying bandwidths and thus provide the user with an interface to effectively select bandwidths and track their effects. We will also investigate the applicability of asymmetric kernels and if we can use them to actually predict hotspots and their properties. Furthermore, we aim to derive a list of data specific threshold values to support the analysts in exploring critical events, such as when hotspots are filtered out. Although the threshold value $\tau$ enables analysts to filter hotspots, we also plan to extend our technique to track the actual topology of the density estimate over time, i.e., to explicitly track the topology of the underlying, time-varying scalar function. Our method also has some limitations that have to be solved. For instance, if a hotspot does not contain its center of mass, which can be the case for non-convex hotspots, then the edges of the spatio-temporal Reeb graph might not correctly indicate the position of the hotspots. Another limitation is that the method only analyzes event-based spatio-temporal data in its most basic form, i.e., a location-time pair. It would be possible to adapt the density estimate to also support event data with associated scalar values, such as the severity of a reported outbreak, or, more interestingly, datasets that already contain information about the relationship between events. For example, epidemiological datasets that contain contact information between infected persons, or crime datasets that contain the identity of criminals and links between incidents.

## 6. ACKNOWLEDGMENTS

Some of the material presented here was sponsored by Department of Defense and is approved for public release, case number 15-383 and upon work supported by the NSF under Grant No. 1350573.

## 7. REFERENCES

[1] G. Andrienko, N. Andrienko, H. Schumann, and C. Tominski. Visualization of Trajectory Attributes in Space-Time Cube and Trajectory Wall. In *Cartography from Pole to Pole*, pages 157–163. Springer Berlin Heidelberg, 2014.

[2] N. Andrienko, G. Andrienko, and P. Gatalsky. Exploratory spatio-temporal visualization: an analytical review. *Journal of Visual Languages & Computing*, 14(6):503–541, 2003.

[3] K. Bennett, T. Carroll, P. Lowe, and J. Phillipson. Coping with Crisis in Cumbria: the Consequences of Foot and Mouth Disease. Technical report, Centre for Rural Economy, Newcastle University, 2002.

[4] C. A. Brewer. *Designing Better Maps: A Guide for GIS Users*. Environmental Systems Research Institute Inc.,U.S., 2005.

[5] C. Brunsdon, J. Corcoran, and G. Higgs. Visualising space and time in crime patterns: A comparison of methods . *Computers, Environment and Urban Systems*, pages 52–75, 2007.

[6] Comptroller and Auditor General. The 2001 Outbreak of Foot and Mouth Disease. Technical report, National Audit Office (NAO), 2002.

[7] Department for Environment, Food and Rural Affairs (DEFRA). Origin of the UK Foot and Mouth Disease epidemic in 2001. Technical report, 2002.

[8] H. Doraiswamy, N. Ferreira, T. Damoulas, J. Freire, and C. Silva. Using Topological Analysis to Support Event-Guided Exploration in Urban Data. *IEEE Symposium on Visualization and Computer Graphics*, 20(12):2634–2643, 2014.

[9] H. Doraiswamy, V. Natarajan, and R. Nanjundiah. An Exploration Framework to Identify and Track Movement of Cloud Systems. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2896–2905, 2013.

[10] R. Eccles, T. Kapler, R. Harper, and W. Wright. Stories in GeoTime. In *IEEE Symposium on Visual Analytics Science and Technology*, pages 19–26, 2007.

[11] H. Edelsbrunner, J. Harer, A. Mascarenhas, V. Pascucci, and J. Snoeyink. Time-varying Reeb Graphs for Continuous Space-Time Data. *Computational Geometry*, 41(3):149–166, 2008.

[12] E. Gabriel, B. Rowlingson, and P. J. Diggle. stpp: An R Package for Plotting, Simulating and Analysing Spatio-Temporal Point Patterns. *Journal of Statistical Software*, 53(2), 2013.

[13] P. Gatalsky, N. Andrienko, and G. Andrienko. Interactive analysis of event data using space-time cube. In *Eighth International Conference on Information Visualisation*, pages 145–152, 2004.

[14] K. Goldsberry and S. Battersby. Issues of change detection in animated choropleth maps. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 2009.

[15] T. Hagerstrand. What about people in Regional Science? *Papers of the Regional Science Association*, 24(1):6–21, 1970.

[16] F. Hardisty and A. C. Robinson. The geoviz toolkit: using component-oriented coordination methods for geographic visualization and analysis. *International Journal of Geographical Information Science*, pages 191–210, 2011.

[17] M. Harrower. The cognitive limits of animated maps. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 2009.

[18] T. Hengl, P. Roudier, D. Beaudette, and E. Pebesma. plotKML: Scientific Visualization of Spatio-temporal Data. *Journal of Statistical Software*, pages 1–23, 2014.

[19] G. Ji, H.-W. Shen, and R. Wenger. Volume Tracking Using Higher Dimensional Isosurfacing. In *Proceedings of the 14th IEEE Visualization*, pages 28–, Washington, DC, USA, 2003. IEEE Computer Society.

[20] M. Kraak. The space-time cube revisited from a geovisualization perspective. pages 1988–1996, 2003.

[21] R. Maciejewski, S. Rudolph, R. Hafen, A. Abusalah, M. Yakout, M. Ouzzani, W. S. Cleveland, S. J. Grannist, and D. S. Ebert. A Visual Analytics Approach to Understanding Spatiotemporal Hotspots. *IEEE Transactions on Visualization and Computer Graphics*, 2009.

[22] A. Malik, R. Maciejewski, S. Towers, S. McCullough, and D. S. Ebert. Proactive Spatiotemporal Resource Allocation and Predictive Visual Analytics for Community Policing and Law Enforcement. *IEEE Transactions on Visualization and Computer Graphics*, 2014.

[23] A. Mascarenhas and J. Snoeyink. Isocontour based Visualization of Time-varying Scalar Fields. In *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, Mathematics and Visualization, pages 41–68. Springer Berlin Heidelberg, 2009.

[24] C. Minard. *Des tableaux graphiques et des cartes figuratives*. Thunot, 1862.

[25] T. Nakaya and K. Yano. Visualising Crime Clusters in a Space-time Cube: An Exploratory Data-analysis Approach Using Space-time Kernel Density Estimation and Scan Statistics. *Transactions in GIS*, 14(3):223–239, 2010.

[26] P. Oesterling, C. Heine, H. Janicke, G. Scheuermann, and G. Heyer. Visualization of High-Dimensional Point Clouds Using Their Density Distribution's Topology. *IEEE Transactions on Visualization and Computer Graphics*, 17(11):1547–1559, 2011.

[27] S. Peters and L. Meng. Spatio Temporal Density Mapping of a Dynamic Phenomenon. In *GEOProcessing 2014*. Department of Cartography - Technische Universitaet Muenchen, 2014.

[28] D. J. Peuquet and M.-J. Kraak. Geobrowsing: Creative Thinking and Knowledge Discovery Using Geographic Visualization. *Information Visualization*, 1(1):80–91, 2002.

[29] R. Samtaney, D. Silver, N. Zabusky, and J. Cao. Visualizing Features and Tracking Their Evolution. *Computer*, 27(7):20–27, 1994.

[30] R. Scheepens, N. Willems, H. van de Wetering, and J. van Wijk. Interactive density maps for moving objects. *Computer Graphics and Applications, IEEE*, 32(1):56–66, 2012.

[31] A. Shrestha, B. Miller, Y. Zhu, and Y. Zhao. Storygraph: Extracting Patterns from Spatio-temporal Data. In *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics*, pages 95–103. ACM, 2013.

[32] D. Silver and X. Wang. Tracking and visualizing turbulent 3D features. *IEEE Transactions on Visualization and Computer Graphics*, 3(2):129–141, 1997.

[33] United States Department of Agriculture (USDA). Foot-and-Mouth Disease Response Plan, 2014.

[34] G. Weber, P.-T. Bremer, M. Day, J. Bell, and V. Pascucci. Feature Tracking Using Reeb Graphs. In *Topological Methods in Data Analysis and Visualization*, pages 241–253. Springer Berlin Heidelberg, 2011.

[35] W. Widanagamaachchi, C. Christensen, P.-T. Bremer, and V. Pascucci. Interactive exploration of large-scale time-varying data using dynamic tracking graphs. In *IEEE Symposium on Large Data Analysis and Visualization*, pages 9–17, 2012.