SHORT-PAPER

A Fast Line Density Visualization Plugin for Geographic Information Systems

TSZ NAM CHAN, Shenzhen University, Shenzhen, Guangdong, China

BOJIAN ZHU, Hong Kong Baptist University, Hong Kong, Hong Kong

DINGMING WU, Shenzhen University, Shenzhen, Guangdong, China

YUN PENG, Guangzhou University, Guangzhou, Guangdong, China

LEONG HOU U, University of Macau, Taipa, Macao

WEI TU, Shenzhen University, Shenzhen, Guangdong, China

View all

Open Access Support provided by:

Shenzhen University

University of Macau

Guangzhou University

Hong Kong Baptist University

PDF Download
3722212.3725075.pdf
27 January 2026
Total Citations: 0
Total Downloads: 681

Published: 22 June 2025

Citation in BibTeX format

SIGMOD/PODS '25: International Conference on Management of Data
June 22 - 27, 2025
Berlin, Germany

Conference Sponsors:
SIGMOD

A Fast Line Density Visualization Plugin for Geographic Information Systems

Tsz Nam Chan
College of Computer Science and Software Engineering,
Shenzhen University
Shenzhen, China
edisonchan@szu.edu.cn

Bojian Zhu
Department of Computer Science,
Hong Kong Baptist University
Hong Kong, China
csbjzhu@comp.hkbu.edu.hk

Dingming Wu
College of Computer Science and Software Engineering,
Shenzhen University
Shenzhen, China
dingming@szu.edu.cn

Yun Peng
Institute of Artificial Intelligence and Blockchain, Guangzhou University
Guangzhou, China
yunpeng@gzhu.edu.cn

Leong Hou U
Department of Computer and Information Science,
University of Macau
Macao, China
ryanlhu@um.edu.mo

Wei Tu
School of Architecture and Urban Planning, Shenzhen University
Shenzhen, China
tuwei@szu.edu.cn

Ruisheng Wang
School of Architecture and Urban Planning, Shenzhen University
Shenzhen, China
ruiswang@szu.edu.cn

Abstract

Line Density Visualization (LDV) has been widely used in different domains, e.g., transportation science, urban planning, and criminology. Therefore, various geographic information systems, e.g., ArcGIS and QGIS, can also support this tool. However, all these GIS platforms mainly adopt the naïve algorithm for generating LDV, which cannot be scalable to high resolution sizes and large-scale line-segment datasets. To tackle this issue, we have developed a new QGIS plugin, called Fast Line Density Analysis, for efficiently supporting two types of accurate approximation, namely (1) generating LDV with an ϵ-relative error guarantee (ϵLDV) and (2) generating LDV with multiple thresholds (τLDV). In this demonstration, we have prepared three large-scale datasets (with up to 15.7 million line segments) for participants to compare our plugin with QGIS and ArcGIS in terms of accuracy and efficiency. In particular, participants can also tune different parameters in our plugin for understanding how they can affect the visualization quality and efficiency. The demonstration video is available in the YouTube and Bilibili links, which are https://www.youtube.com/watch?v=EYl3Wieb-2I and https://www.bilibili.com/video/BV1aJwaeMEHr, respectively.

This work is supported in part by National Key R&D Program of China 2023YFC3321300, 2022YFB2602105, National Natural Science Foundation of China under Grants 23IAA00610, 62202401, 62372308, 62472116, 42311530335, 42471496, GuangDong Basic and Applied Basic Research Foundation 2023A1515011619, 2023A1515030273, the Innovation Team of the Department of Education of Guangdong Province (2024KCXTD013), the Science and Technology Development Fund Macau SAR (0003/2023/RIC, 0052/2023/RIA1, 0031/2022/A, 001/2024/SKL for SKL-IOTSC), Joint Foundation of Guangzhou and Universities on Basic Research (2025A03J3178), and the Key Technological Innovation Program of Ningbo City under Grant No. 2024Z297. Dingming Wu is the corresponding author of this paper.

CCS Concepts

• **Information systems → Geographic information systems**.

Keywords

Line Density Visualization; Fast Plugin; GIS

ACM Reference Format:
Tsz Nam Chan, Bojian Zhu, Dingming Wu, Yun Peng, Leong Hou U, Wei Tu, and Ruisheng Wang. 2025. A Fast Line Density Visualization Plugin for Geographic Information Systems. In *Companion of the 2025 International Conference on Management of Data (SIGMOD-Companion '25), June 22–27, 2025, Berlin, Germany*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3722212.3725075

1 Introduction

Line Density Visualization (LDV) [1, 4] is widely used for analyzing flow data/trajectory data in various domains, e.g., transportation science, urban planning, and criminology. Transportation scientists [11, 15] and urban planners [10, 16] adopt LDV to analyze the traffic flow/trajectory patterns in different cities, while criminologists [13, 14] adopt LDV to understand crime patterns in different geographical regions. Figure 1 shows how to adopt LDV to analyze flow patterns of the Los Angeles bicycle mobility dataset [3]. Observe that the eastern and western regions of Los Angeles have the high line density values compared with other regions.

Due to the popularity of LDV, many geographic information systems, including QGIS [4] and ArcGIS [1], can also support this visualization tool. However, LDV is computationally expensive,

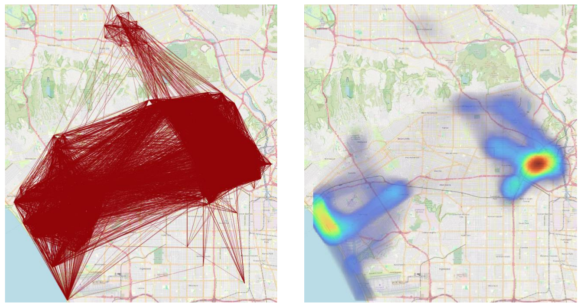

(a) Line segment (flow) dataset
(b) Line density visualization (LDV)

Figure 1: Generate LDV (in (b)) for the Los Angeles bicycle mobility dataset, where the red line segments in (a) denote the bicycle flows that move from one place to another place.

which takes $O(XYn)$ time, where $X \times Y$ and n denote the resolution size and the number of line segments (e.g., red line segments in Figure 1a), respectively. Consider the Los Angeles bicycle mobility dataset (with 0.402 million line segments) and the 1280×960-resolution as an example. Generating a single LDV for this dataset takes at least 0.494 trillion operations. As such, LDV cannot be efficient (or even feasible) to generate high-resolution LDV for a large-scale line-segment dataset.

To handle the efficiency issues of LDV, we develop a new QGIS plugin, called Fast Line Density Analysis [2], in this paper. First, this plugin is based on our preliminary work, LARGE [9], for efficiently generating approximate LDV with a non-trivial relative error guarantee, namely ϵLDV. Second, since domain experts [14, 16] need to generate LDV with a fixed number of color levels, τLDV (e.g., density values[1] in the ranges of [0, 0.25), [0.25, 0.5), [0.5, 0.75), and [0.75, 1] (i.e., four levels) can be colored by blue, light green, yellow, and red, respectively), we further modify our efficient solution in [9] for this setting. Compared with the traditional LDV tools in QGIS [4] and ArcGIS [1], Fast Line Density Analysis is the first plugin that simultaneously (1) generates accurate LDV with non-trivial guarantees (i.e., ϵLDV and τLDV) and (2) achieves significant efficiency improvement. **Until 20th April 2025, the total number of downloads of this new plugin has already reached 2,705.**

The rest of the demonstration paper is summarized as follows. We first discuss the technical overview of our plugin, Fast Line Density Analysis, in Section 2. Then, we illustrate how to use our plugin in Section 3. Lastly, we provide the demonstration plan in Section 4.

2 Technical Overview

In this section, we first provide the formal problem definitions (ϵLDV and τLDV) in Section 2.1. Then, we briefly discuss the length-aggregation-based grid structure, LARGE [9], in Section 2.2. Next, we further illustrate the bound functions, based on LARGE, in Section 2.3. After that, we discuss how to incorporate these bound functions into the filter-and-refinement framework for solving ϵLDV and τLDV in Section 2.4. Lastly, we discuss a simple yet efficient approach for specifying the thresholds of τLDV in Section 2.5.

2.1 Problem Definitions

Recall that we need to color each pixel $\mathbf{q}$ based on the line density function $\mathcal{L}(\mathbf{q})$, which computes the accumulated length of all line segments that are within the bandwidth parameter b over the circular searching area (see Figure 2), for generating LDV. We formally state LDV in Definition 1.

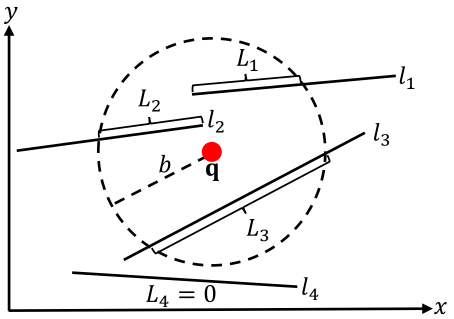


Figure 2: Illustration of computing the line density function, where $\mathcal{L}(\mathbf{q}) = \frac{L_1+L_2+L_3}{\pi b^2}$ in this example.

Definition 1. *Given a set of line segments $\mathbb{L} = \{l_1, l_2, ..., l_n\}$ with size n and a bandwidth parameter b, we need to compute $\mathcal{L}(\mathbf{q})$ for each pixel $\mathbf{q}$.*

$$\mathcal{L}(\mathbf{q}) = \frac{1}{\pi b^2} \sum_{i=1}^{n} L_i \tag{1}$$

where L_i denotes the length of the i^{th} line segment that is within the bandwidth parameter b from $\mathbf{q}$.

With Definition 1, we then formally define two types of approximation, ϵLDV and τLDV, in Problem 1 and Problem 2, respectively.

Problem 1. *(ϵLDV) Given a relative error ϵ, we need to compute $A(\mathbf{q})$ for each pixel $\mathbf{q}$ so that $(1-\epsilon)\mathcal{L}(\mathbf{q}) \leq A(\mathbf{q}) \leq (1+\epsilon)\mathcal{L}(\mathbf{q})$.*

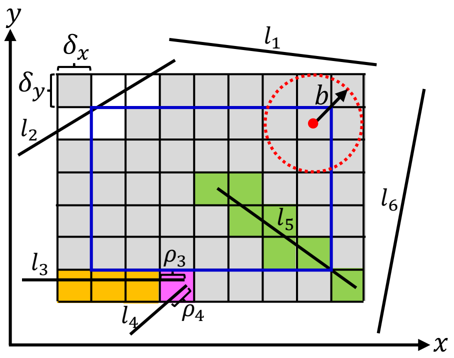
Problem 2. *(τLDV) Given a set of D thresholds $\tau_1, \tau_2, ..., \tau_D$, we need to classify $\mathcal{L}(\mathbf{q})$ to be different color levels $C(\mathbf{q})$, where*

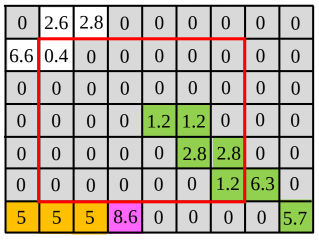
$$C(\mathbf{q}) = \begin{cases} 0 & \text{if } \mathcal{L}(\mathbf{q}) < \tau_1 \\ 1 & \text{if } \tau_1 \leq \mathcal{L}(\mathbf{q}) < \tau_2 \\ \vdots & \vdots \\ D & \text{if } \mathcal{L}(\mathbf{q}) \geq \tau_D \end{cases} \tag{2}$$

2.2 LARGE: A Length-Aggregation-based Grid Structure

In order to efficiently solve the ϵLDV and τLDV problems, we need to construct a Length-AggRegation-based Grid structurE (LARGE) [9], which is shown in Figure 3. Given an original $X \times Y$-plane (bounded by the blue rectangle in Figure 3a), where each pixel has size $\delta_x \times \delta_y$, we need to first find the extended region, which covers the search range (decided by the bandwidth b) of each pixel (i.e., all grey pixels in Figure 3a). With this extended region, we can then build the grid structure, where each grid stores the accumulated length of all line segments (e.g., the pink grid stores the value $\rho_3 + \rho_4 = 8.6$ in Figure 3a). By constructing the prefix-sum grid structure (see Figure 3b), it only takes $O(1)$ time to obtain the length aggregation value with any rectangular region in the grid structure. As an example, we only need to access (at most) four green entries in Figure 3b (i.e., $39.8 - 23.6 - 11.6 + 5 = 9.6$) in order to compute the length aggregation value that is in the red rectangular region of the grid structure in Figure 3a (i.e., $0.4 + 1.2 + 1.2 + 2.8 + 2.8 + 1.2 = 9.6$). More details that are related to LARGE can be found in our preliminary work [9].

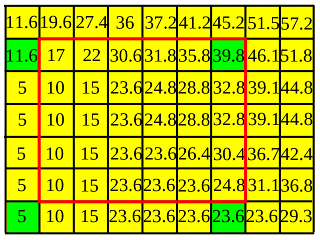
[1]In this example, we assume that the density values have been normalized to be [0, 1].





0	2.6	2.8	0	0	0	0	0	0
6.6	0.4	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	1.2	1.2	0	0	0
0	0	0	0	0	2.8	2.8	0	0
0	0	0	0	0	0	1.2	6.3	0
5	5	5	8.6	0	0	0	0	5.7

(a) Find the accumulative length for each grid in the grid structure.



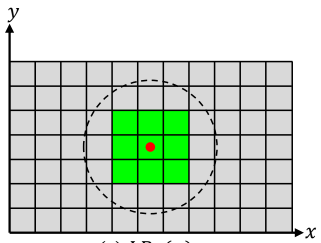
11.6	19.6	27.4	36	37.2	41.2	45.2	51.5	57.2
11.6	17	22	30.6	31.8	35.8	39.8	46.1	51.8
5	10	15	23.6	24.8	28.8	32.8	39.1	44.8
5	10	15	23.6	24.8	28.8	32.8	39.1	44.8
5	10	15	23.6	23.6	26.4	30.4	36.7	42.4
5	10	15	23.6	23.6	23.6	24.8	31.1	36.8
5	10	15	23.6	23.6	23.6	23.6	23.6	29.3

(b) Obtain the prefix-sum grid structure (LARGE).

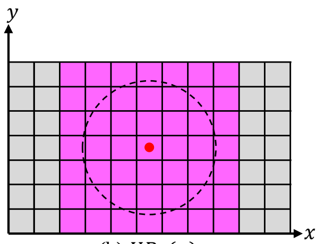
Figure 3: Constructing the prefix-sum grid structure, LARGE (in (b)), for an example dataset with six line segments (in (a)).

2.3 Bound Functions

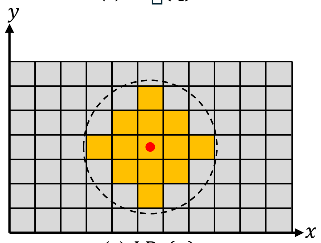
Observe from Figure 2 that computing $\mathcal{L}(\mathbf{q})$ is based on calculating the accumulated length of all line segments that are within the bandwidth (or search range) b from $\mathbf{q}$. Therefore, all grids that are within the search range (i.e., Figure 4a and Figure 4c) and that cover the search range (i.e., Figure 4b and Figure 4d) can act as the lower bound functions and the upper bound functions, respectively.



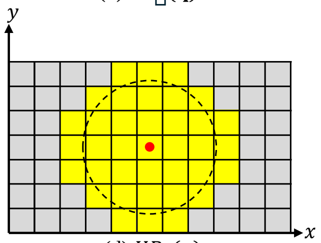
(a) $LB_{\square}(\mathbf{q})$



(b) $UB_{\square}(\mathbf{q})$



(c) $LB_a(\mathbf{q})$



(d) $UB_a(\mathbf{q})$

Figure 4: Illustration of the square-shaped (a and b) and arbitrary-shaped (c and d) bound functions.

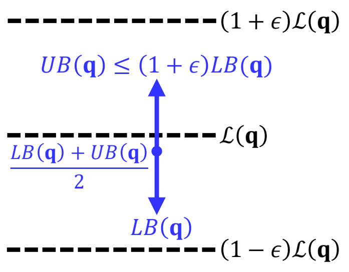
With the prefix-sum grid structure, LARGE, which can calculate the aggregation value of any rectangular region in $O(1)$ time, our preliminary work [9] states that the square-shaped bound functions ($LB_{\square}(\mathbf{q})$ and $UB_{\square}(\mathbf{q})$) and the arbitrary-shaped bound functions ($LB_a(\mathbf{q})$ and $UB_a(\mathbf{q})$) can be computed in $O(1)$ time and $O(\min(X, Y))$ time (by simply regarding each row of grids or each column of grids as a rectangular region in Figure 4c and Figure 4d), respectively.

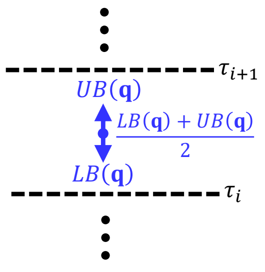
2.4 Filter-and-Refinement Framework

Once we have the lower and upper bound functions, we adopt the filter-and-refinement framework [6–8] for solving the ϵLDV and τLDV problems.

ϵLDV. Observe from Figure 5a that we can have the valid approximation value $A(\mathbf{q}) = \frac{LB(\mathbf{q})+UB(\mathbf{q})}{2}$ for each pixel $\mathbf{q}$ in the ϵLDV problem if the condition $UB(\mathbf{q}) \leq (1+\epsilon)LB(\mathbf{q})$ holds. Therefore, we progressively check whether the $(LB_{\square}(\mathbf{q}), UB_{\square}(\mathbf{q}))$-pair and $(LB_a(\mathbf{q}), UB_a(\mathbf{q}))$-pair can fulfill the condition. If both pairs cannot fulfill this condition, we adopt the state-of-the-art R-tree structure [12] for computing the exact value of $\mathcal{L}(\mathbf{q})$.

τLDV. Observe from Figure 5b that we have $C(\mathbf{q}) = i$ in the τLDV problem if $\tau_i \leq LB(\mathbf{q}) \leq UB(\mathbf{q}) < \tau_{i+1}$, where $0 \leq i \leq D$ (Here, we define two dummy parameters $\tau_0 = 0$ and $\tau_{D+1} = \infty$). Like ϵLDV, we first adopt the progressive approach ((1) $(LB_{\square}(\mathbf{q}), UB_{\square}(\mathbf{q}))$-pair and (2) $(LB_a(\mathbf{q}), UB_a(\mathbf{q}))$-pair) to check whether the condition is fulfilled and then adopt the R-tree structure for computing $\mathcal{L}(\mathbf{q})$ if the condition cannot be satisfied by all pairs of bound functions.


$(1+\epsilon)\mathcal{L}(\mathbf{q})$
$UB(\mathbf{q}) \leq (1+\epsilon)LB(\mathbf{q})$
$\mathcal{L}(\mathbf{q})$
$\frac{LB(\mathbf{q})+UB(\mathbf{q})}{2}$
$LB(\mathbf{q})$
$(1-\epsilon)\mathcal{L}(\mathbf{q})$
(a) ϵLDV



(b) τLDV

Figure 5: The filtering condition of using the lower and upper bound functions for solving the ϵLDV and τLDV problems.

2.5 Specification of Thresholds for τLDV

In practice, it can be challenging to set the suitable threshold values (i.e., $\tau_1, \tau_2, \ldots, \tau_D$) for $C(\mathbf{q})$ (see Equation 2) before knowing $\mathcal{L}(\mathbf{q})$ of all pixels $\mathbf{q}$. However, evaluating exact $\mathcal{L}(\mathbf{q})$ can be very inefficient. Therefore, we propose to adopt the most efficient $LB_{\square}(\mathbf{q})$ and $UB_{\square}(\mathbf{q})$ to estimate the range $[\mathbb{L}, \mathbb{U}]$ of $\mathcal{L}(\mathbf{q})$, where (for all pixels $\mathbf{q}$)

$$\mathbb{L} = \min_{\mathbf{q}} LB_{\square}(\mathbf{q}) \leq \mathcal{L}(\mathbf{q}) \leq \max_{\mathbf{q}} UB_{\square}(\mathbf{q}) = \mathbb{U} \tag{3}$$

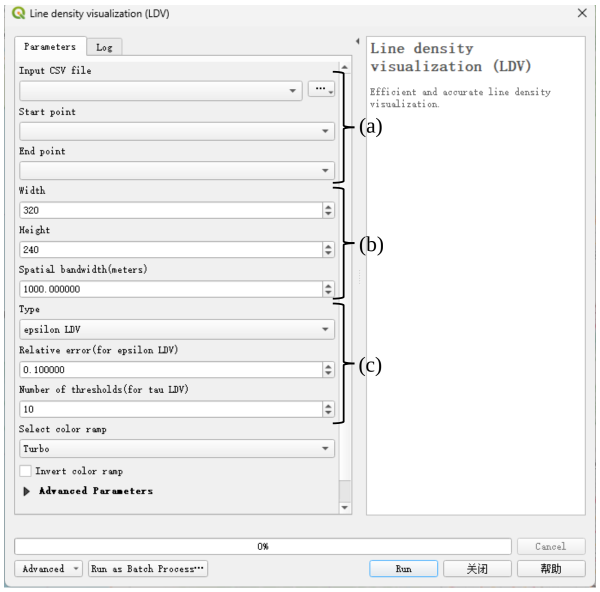
With this range $[\mathbb{L}, \mathbb{U}]$, we can set the thresholds τ_i ($1 \leq i \leq D$) by dividing this range with equal-size intervals, where

$$\tau_i = \mathbb{L} + i \times \left(\frac{\mathbb{U} - \mathbb{L}}{D+1}\right) \tag{4}$$

As a remark, since the time complexity of computing $LB_{\square}(\mathbf{q})$ and $UB_{\square}(\mathbf{q})$ is $O(1)$, we can specify these D thresholds (by evaluating $\mathbb{L}$ and $\mathbb{U}$) in $O(XY)$ time, which is efficient.

3 Illustration of Our Plugin

Based on the techniques in Section 2, we further develop the QGIS plugin, called Fast Line Density Analysis [2] (see Figure 6). In this section, we discuss different components of this plugin.



Line density visualization (LDV)

Efficient and accurate line density visualization.

Input CSV file:

Start point:

End point:

Width: 320

Height: 240

Spatial bandwidth(meters): 1000.000000

Type: epsilon LDV

Relative error(for epsilon LDV): 0.100000

Number of thresholds(for tau LDV): 10

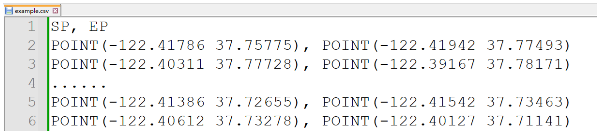
Select color ramp: Turbo

Invert color ramp

Advanced Parameters

Figure 6: The user interface of our plugin, namely Fast Line Density Analysis, for QGIS.

Component (a) (Details of the input file): Users need to provide a csv file name in the box of "Input CSV file". Observe from Figure 7 that the first row specifies the attribute names of the starting point ("SP") and the ending point ("EP") of all line segments. Moreover, each row in this file stores two attributes of POINT(longitude latitude) to represent the starting point and the ending point of each line segment. Therefore, users need to choose "SP" and "EP" in the drop-down list of "start point" and "end point", respectively.


example.csv

```
SP, EP
POINT(-122.41786 37.75775), POINT(-122.41942 37.77493)
POINT(-122.40311 37.77728), POINT(-122.39167 37.78171)
......
POINT(-122.41386 37.72655), POINT(-122.41542 37.73463)
POINT(-122.40612 37.73278), POINT(-122.40127 37.71141)
```

Figure 7: An example input file.

Component (b) (Parameters of LDV): Users need to set the resolution size in the "Width" and "Height" boxes (e.g., 320 pixels and 240 pixels in the width (x-axis) and height (y-axis), respectively, in Figure 6) and set the bandwidth value b (e.g., 1000m in Figure 6).

Component (c) (LDV (or Problem) type): Users need to choose the LDV types, which are either ϵLDV (see Problem 1) or τLDV (see Problem 2). If ϵLDV is selected, the "Relative error" box is enabled for users to set the value of relative error ϵ (e.g., 0.1). If τLDV is selected, the "Number of thresholds" box is enabled for users to set the number of thresholds D (e.g., 10).

Once all parameters have been provided in the plugin, users can click the "Run" button (see Figure 6) for generating the approximate LDV (either ϵLDV or τLDV).

4 Demonstration Plan

In this demonstration, we adopt three large-scale datasets (with up to 15.7 million line segments), which are (1) Los Angeles bicycle mobility dataset [3], (2) San Francisco taxi mobility dataset [5], and (3) Shenzhen vehicle mobility dataset (obtained from the Shenzhen Transportation Bureau), for testing. Here, we consider four demonstration scenarios.

Investigation of the choice of the relative error ϵ in ϵLDV. Recall from Problem 1 that users need to choose the relative error ϵ in order to generate ϵLDV. Therefore, users can have this question. *Which ϵ should be chosen in practice?* To answer this question, we first generate (1) multiple ϵLDVs by varying ϵ (e.g., 0.05, 0.1, 0.15, and 0.2) and (2) the exact LDV for each dataset. Then, we compare the visualization quality and the efficiency of ϵLDVs with the exact LDV, which can show the trade-off between these two aspects.

Investigation of the number of thresholds D in τLDV. Like ϵLDV, users also need to choose the number of thresholds D in advance for generating τLDV (see Problem 2). Therefore, we also compare the visualization quality and the efficiency for generating multiple τLDVs (by choosing $D = 5$, 10, 15, and 20) and the exact LDV so that we can understand the trade-off between them.

Comparisons of Fast Line Density Analysis with QGIS and ArcGIS. Since QGIS [4] and ArcGIS [1] can support LDV, we also compare the efficiency and the visualization quality of our plugin with them using the three large-scale datasets in the demonstration.

Investigation of bandwidth tuning. Note that the bandwidth parameter b (see Figure 2 and Definition 1) can affect the visualization quality of LDV. In our demonstration, we vary b for generating multiple ϵLDVs and τLDVs so that users can understand how to choose the best bandwidth b for each dataset.

References

[1] [n. d.]. ArcGIS. https://pro.arcgis.com/en/pro-app/latest/tool-reference/spatial-analyst/how-line-density-works.htm.
[2] [n. d.]. Fast Line Density Analysis. https://plugins.qgis.org/plugins/fast_line_density_analysis/.
[3] [n. d.]. Los Angeles Bike Trip Data. https://bikeshare.metro.net/about/data/.
[4] [n. d.]. QGIS. https://docs.qgis.org/3.28/en/docs/user_manual/processing_algs/qgis/interpolation.html#line-density.
[5] [n. d.]. San Francisco taxi trajectories. https://figshare.com/articles/dataset/San_Francisco_taxi_trajectories/12302243.
[6] Tsz Nam Chan, Reynold Cheng, and Man Lung Yiu. 2020. QUAD: Quadratic-Bound-based Kernel Density Visualization. In *SIGMOD*. 35–50.
[7] Tsz Nam Chan, Leong Hou U, Reynold Cheng, Man Lung Yiu, and Shivansh Mittal. 2022. Efficient Algorithms for Kernel Aggregation Queries. *IEEE Trans. Knowl. Data Eng.* 34, 6 (2022), 2726–2739.
[8] Tsz Nam Chan, Man Lung Yiu, and Leong Hou U. 2019. KARL: Fast Kernel Aggregation Queries. In *ICDE*. 542–553.
[9] Tsz Nam Chan, Bojian Zhu, Dingming Wu, Yun Peng, and Leong Hou U. 2024. LARGE: A Length-Aggregation-based Grid Structure for Line Density Visualization. *Proc. VLDB Endow.* 17, 13 (2024), 4585–4598.
[10] Anita Graser. 2021. An exploratory data analysis protocol for identifying problems in continuous movement data. *Journal of Location Based Services* 15, 2 (2021), 89–117.
[11] Chris Hill, Marcus Young, Simon Blainey, Stefano Cavazzi, Chris Emberson, and Jason Sadler. 2024. An integrated geospatial data model for active travel infrastructure. *Journal of Transport Geography* 117 (2024), 103889.
[12] Scott T. Leutenegger, Mario Alberto López, and J. M. Edgington. 1997. STR: A Simple and Efficient Algorithm for R-Tree Packing. In *ICDE*. 497–506.
[13] Justin Song, Richard Frank, Patricia L. Brantingham, and Jim LeBeau. 2012. Visualizing the spatial movement patterns of offenders. In *SIGSPATIAL*. ACM, 554–557.
[14] Lisa Tompson, Henry Partridge, and Naomi Shepherd. 2009. Hot routes: Developing a new technique for the spatial analysis of crime. *Crime Mapping: A Journal of Research and Practice* 1, 1 (2009), 77–96.
[15] Wei Yang, Jie Hu, Yong Liu, and Wenbo Guo. 2023. Examining the influence of neighborhood and street-level built environment on fitness jogging in Chengdu, China: a massive GPS trajectory data analysis. *Journal of transport geography* 108 (2023), 103575.
[16] Xue Yang, Xuejiao Zheng, Yanjia Cao, Hao Chen, Luliang Tang, and Honghai Yang. 2023. Connectivity analysis in pedestrian networks: A case study in Wuhan, China. *Applied geography* 151 (2023), 102843.