

Demers Cartogram with Rivers

Abstract

Cartograms serve as representations of geographical and abstract data, employing a value-by-area mapping technique. As a variant of the Dorling cartogram, the Demers cartogram utilizes squares instead of circles to represent regions. This alternative approach allows for a more intuitive comparison of regions, utilizing screen space more efficiently. However, a drawback of the Dorling cartogram and its variants lies in the potential displacement of regions from their original positions, ultimately compromising legibility, readability, and accuracy. To tackle this limitation, we propose a novel hybrid cartogram layout algorithm that incorporates topological elements, such as rivers, into Demers cartograms. The presence of rivers significantly impacts both the layout and visual appearance of the cartograms. Through a user study conducted on an Electronic Health Records (EHR) dataset, we evaluate the efficacy of the proposed hybrid layout algorithm. The obtained results illustrate that this approach successfully retains key aspects of the original cartogram while enhancing legibility, readability, and overall accuracy.

Keywords: Cartogram, Demers Cartogram, Visual Analytics, EHR Visualization

1. Introduction and Motivation

Cartograms are representations of geographical and abstract data based on a value-by-area mapping combining statistical and geographical information [1, 2]. Various styles of cartograms have been proposed and implemented for applications such as urban planning [3, 4], natural hazard forecasting [5, 6], conservation and environmental planning [7, 8], political and social demographics [9, 10], and decision-making for public health [11, 12].

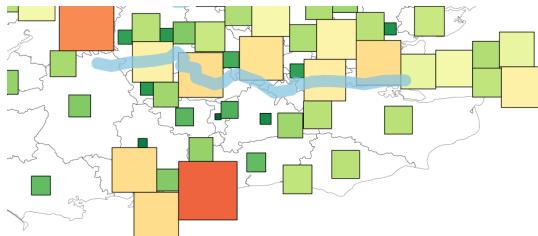


Figure 1: The River Thames passing through a cartographic representation of the NHS CCGs in London and surrounding areas.

Among the four types of cartogram categorized in a survey by Nusrat and Kobourov [13] (See Section 2 for definitions of contiguous, non-contiguous, rectangular, and Dorling), a trade-off is made between types of accuracy (See Table 2). For this project, we focus on non-contiguous cartograms like Demers cartograms because they facilitate statistical comparison between regions, they can make good use of screen space, and comparison of regions is useful when studying Electronic Health Records (EHR) data. Demers cartograms offer the advantage in cases where the data is not directly correlated to region sizes. In other words, Demers cartograms are useful when a data dimension is not describing the geography of the region it represents but is tied to something else, for example the health

of its population. Also, the comparison of magnitudes becomes an area estimation task, which is effective for a numeric data encoding [14]. See Nickel et al. [15] for a more complete description of the advantages that Demers cartograms offer. Building on Demers cartograms [16], we introduce and develop novel features, such as rivers, aiming to improve the readability and geographical accuracy without sacrificing statistical accuracy. Standard Demers cartograms are composed of square nodes that represent geographic enumeration units. As such, this can reduce their legibility. We implement a new hybrid cartographic layout algorithm that combines rivers with the placement of the nodes representing geographic enumeration units, in this case, a Clinical Commissioning Group (CCG). We hypothesize that introducing rivers improves the overall legibility of a cartogram. By *legibility* we mean readability and ability to interpret the cartogram. To assess this hypothesis, we designed an experimental setup where participants engaged in correspondence and location tasks as part of a user study. To reduce error and make efficient use of screen space, the algorithm also updates the position of rivers to accommodate the node layout. We then apply the algorithm to a real-world case study using EHR data to evaluate the result. We present a user study that demonstrates its effectiveness.

Our contributions include:

- A new variant of Demers cartograms that incorporates rivers to improve readability and recognizability,
- A novel hybrid layout algorithm that combines node positions with features such as rivers,
- A user study evaluation of the technique with an application to EHRs.

The results of the user study indicate that rivers can improve the legibility of cartograms. One of the major challenges involved is developing a layout algorithm that handles different

shapes. In other words, the hybrid layout algorithm is novel because it handles different types of elements: square node representing regions and polylines representing rivers. Another challenge we overcome in developing the algorithm is to resolve stalemate situations where nodes become congested due to constraints imposed by rivers, while ensuring error minimization.

2. Related Work

This section introduces the characteristics of various cartogram styles, describes relevant applications of cartograms, and provides a brief overview of some real-world implementations of cartogram-based visualizations.

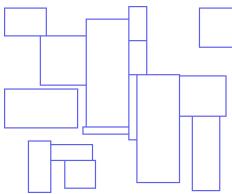
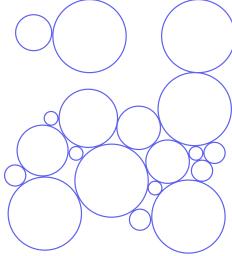
2.1. Definitions

While we focus on rectangular cartogram variants, we start with brief definitions of contiguous and non-contiguous cartograms: Contiguous cartograms preserve topology, maintain connectivity with their adjacent neighbors, but are also subject to distortion in shape. Non-contiguous cartograms sacrifice topological connectivity with neighbors to enable expansion or reduction in size while maintaining their polygonal shape [1].

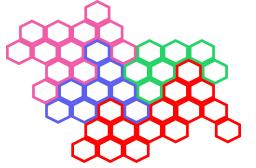
Nusrat and Kobourov [13] define and summarize three major accuracy dimensions for cartograms: statistical, geographical, and topological. Each cartogram design may make various types of accuracy trade-offs between dimensions. We provide a comparison of these trade-offs as introduced by Nusrat and Kobourov [13] in Table 2. In addition, we include the Demers as it is the focus of our work.

Dorling cartograms, as a variant of non-contiguous cartograms, generally do not preserve geography and topology. A Dorling cartogram is statistically accurate, regions are represented by circles and the data dimensions of interest are represented by the circle area [22]. In a Demers cartogram, a variant of Dorling, squares are used instead to capture a certain level of topology, as described by Cano et al. [23] in their related work section. Dorling cartograms are unable to maintain topological accuracy as circles are often repositioned to remove overlap. Here we focus on Demers cartograms as we use squares to depict regions. This style of cartogram offers the advantages that the comparisons between regions are intuitive and screen space utilization is more efficient. This is important in our use case scenario involving EHRs. Demers cartograms, where regions are represented by squares, often have the advantage of preserving a higher level of topology at the cost of geographical accuracy [16].

Rectangular cartograms are contiguous and do not preserve geographical accuracy [24, 25]. Depending on the variant, a rectangular cartogram may trade-off between statistical and topological accuracy.



Mosaic cartograms usually use square or hexagonal tiles to depict regions, and are contiguous and sacrifice statistical accuracy to preserve some level of geographical accuracy [23]. Some variants are able to preserve topological accuracy as well.



2.2. Peer-reviewed Applications

There has been a substantial amount of research done in this area, here we review some of the important work that has inspired our work. Warf and Winsberg [17] use a Dorling cartogram to represent religious diversity in the US. Sun and Li [18] depict 1996 US election data and 2005 Chinese population data using Dorling, Mosaic, and contiguous cartograms. Gao et al. [11] present a Dorling cartogram to illustrate COVID-19 infections in China. Tong et al. [20] use a Demers cartogram to visualize health-related data by CCG regions in England, the work introduces a novel technique to remove the overlap of squares based on topological features, aiming to improve both geographical and topological accuracy. Nusrat et al. [21] investigate the memorability of contiguous and Dorling cartograms using multiple data sets that include demographic, agriculture, and retail data in the US. See Table 1 for a list of literature that adopts cartograms for visualization with corresponding geographical regions and node counts.

Our work extends the algorithm described by Tong et al. [20] which incorporates a static topological feature into Demers cartograms. Our work enhances that of Tong et al. [20] in multiple ways. First, we introduce multiple features (rivers) into the layout, as opposed to a single river. Second, we make topological features dynamic and further improve legibility and geographical accuracy. By the term *dynamic*, we mean that the position of the rivers is updated as part of the layout algorithm. In previous work, the river is static and serves merely as a boundary. Third, we improve the algorithm to resolve stalemates. Finally, the way we evaluate the cartograms is also different. Tong et al. [20] count river crossings to evaluate error (a statistical metric). Here our focus is on readability, and thus we include a user study.

2.3. Cartograms in Media

Cartograms are an engaging visual representation and therefore they are a popular choice of representation in covering various topics by the media. The Washington Post uses cartograms to visualize the US overseas economic assistance, in arm sales (Mosaic) [26], the 2016 US Election (contiguous) [27], and the Brexit Referendum (Mosaic) [28]. National Geographic uses contiguous and Mosaic cartograms to analyze the 2016 US Election results [29], the same topic is also covered by the Financial Times with a Dorling cartogram [30]. Cruz [19] adapts a Dorling cartogram with both contiguous and non-contiguous cartograms to represent the gender pay gap in Portugal. Sandberg [31] reports the 2018 US midterm Election with a Mosaic cartogram, the same approach is used to cover the 2020 US Election by the New York Times [32] and Bloomberg [33].

Literature	Cartogram Type(s)	Geographic Region(s)	Number of Nodes	Year
Warf and Winsberg [17]	Dorling	US	3,142	2008
Sun and Li [18]	Dorling, Mosaic, Neighbor-preserving	US, China	34 - 49	2010
Cruz [19]	Dorling, Non-contiguous, Neighbor-preserving	Portugal	2,882	2017
Tong et al. [20]	Demers	England	209	2018
Gao et al. [11]	Dorling	China	34	2020
Nusrat et al. [21]	Contiguous, Dorling	US	49	2020
Nickel et al. [15]	Non-contiguous, Demers	US, Netherlands, World	49 - 342	2022

Table 1: Related work with non-contiguous cartogram-based visualizations. **Cartogram type** is the type of cartogram used. **Geographic region** is the geographic region depicted by the cartogram. **Number of nodes** is the number of nodes (representing geographic enumeration units) depicted in the cartogram.

Cartogram Variant	Accuracy			
	Statistical	Geographical	Topological	Contiguity
Contiguous	Variable	Variable	Accurate	Yes
Non-contiguous	Accurate	Shape is accurate	Inaccurate	No
Rectangular	Variable	Shape is inaccurate	Variable	Yes
Dorling	Accurate	Inaccurate	Variable	No
Demers	Accurate	Inaccurate	Variable	No

Table 2: Trade-off between dimensions. Dimension sacrificed in order to improve target dimension's accuracy.

One of the disadvantages of Dorling and Demers cartograms is legibility. The layout algorithms may displace regions far from their original position and make the maps more difficult to interpret. Nickel et al. [15] present a method to compute stable Demers cartograms with multiple constraints to maintain adjacencies with no overlapping nodes.

In this paper, we introduce a new type of topological feature, a river, as a constraint to compute the final layout, with the aim of improving the interpretation, readability, and accuracy of this class of cartograms.

3. Data Description

Processing heterogeneous data can be challenging, especially when an EHR dataset is involved, because the data comes from multiple sources [35]. The first step is to obtain both geospatial boundaries and EHR data. The second step is to preprocess the EHR data to remove empty and erroneous values. The final step is to transform the data into a format that is suitable for cartograms. Geospatial boundaries, or shapefiles, were obtained from the sources described here.

3.1. Choropleth Shapefile

Clinical Commissioning Groups (CCGs) are the primary administrative and geographic unit of the National Health Service (NHS) in the UK [36]. The number of CCGs changes over time due to NHS reorganization. The most up-to-date shapefile is available from the Open Geography Portalx [34]. We decided to use the CCG shapefile from 2020 at the time of writing, due

to the absence of published public EHR data based on the latest CCG reorganizations that took place in 2021 and 2022.

3.2. River Shapefiles

We used OpenStreetMap [37] as our data source to obtain shapefiles for the Thames River, the Trent River, and the Great Ouse River in England. These rivers were chosen as they are well-known rivers, pass through regions with dense populations, and provide informative geographical and topological cues. Although including smaller rivers is technically feasible, it may not increase the legibility of the cartogram. This is an open question for future work.

We first obtain a relation ID by searching for a river, e.g. River Thames, on OpenStreetMap. The relation ID is used to construct a query (See Appendix Appendix A) which enables the user to download the entire river shapefile using Overpass Turbo [38].

After acquiring the shapefiles, we used QGIS [39] to manually adjust the projections and convert them into GeoJSON files. Finally, mapshaper [40] is used to merge and convert the GeoJSON files into a TopoJSON file [41]. TopoJSON eliminates redundant coordinates in the data, improving the rendering speed of our implementation. See Table B.3 on page 13 for the pre-

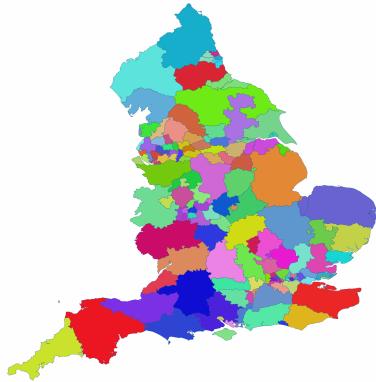


Figure 2: A map of 135 CCGs in England as of 2020, obtained from the Open Geography Portalx [34] with EPSG:4326 (WGS84 - World Geodetic System) as the Coordinate Reference System (CRS).

processing result. We describe the one-time preprocessing steps in more detail in Appendix B.

3.3. EHR Data

We obtained the Clinical Commissioning Group Outcomes Indicator Set (CCG OIS) from NHS Digital [42]. The OIS is a set of indicators that are used to measure the quality of care and the associated health outcomes in the NHS. Some datasets include:

- Under 75 mortality: cardiovascular disease, respiratory disease, liver disease, and cancer
- Emergency hospital admission: stroke, alcohol-specific admission and readmission, coronary heart disease, readmissions within 30 days of discharge, children with lower respiratory tract infections

For all datasets, a spreadsheet including the following is provided:

- Reporting period: Calendar year of registration
- Period of coverage: Start and end date or reporting period
- Breakdown: Organization type
- ONS code: UK Office for National Statistics CCG code
- Level: CCG Code
- Level description: CCG Name
- Gender
- Indicator value: Directly standardized mortality rate
- CI lower: lower 95% confidence interval
- CI upper: upper 95% confidence interval
- Denominator: The count of registered patients
- Numerator: Number of deaths

Each CCG has a unique ONS code, which is used to link the CCG shapefile with the statistical data.

4. Demers Cartogram with Rivers

Algorithm 3 and Figure 3 provide an overview of the hybrid cartogram layout process that includes rivers.

4.1. Initialization with Rivers

We first load and (optionally) render the CCG geospatial boundaries. For each CCG we compute the centroid and represent it using a square node, \mathbf{n} , with the initial size, $s = 1$ pixel. We then load the river shapefiles and render the rivers. Since the vertices of the river in the shapefiles are not in sequential order, we first render the starting vertex, followed by the next nearest vertex. We do not need the original river resolution to incorporate them into cartograms. We reduce their resolution to match that of the cartogram nodes in order to facilitate node-river intersection tests. This rendering approach enables us to adjust the river resolution as shown in Figure 4. We further apply simplification by removing vertices that are too close to each other. The initialization procedure is a one-time process that can be saved for reuse.

4.2. Node Layout and Overlap Removal

We first apply the Fast Node Overlap Removal (FNOR) algorithm that solves the Variable Placement with Separation Constraints (VPSC) problem [43] in order to remove overlap between square nodes. We chose FNOR over other node overlap removal algorithms because FNOR is capable of minimizing spread and node movement while maintaining a good level of global shape preservation [44]. Initialized with a pixel size of unity, we gradually increase the node size by one unit at a time to ensure smooth transitions. An increase in s can cause the nodes to overlap. During overlap removal, we compute node trajectories (See Algorithm 2) and translate nodes to their new position. Nodes that cross a river, denoted $\mathbf{n}_{n_r r}$, are translated back to their previous position. If a node oscillates across a river, we identify this as a stalemate. One iteration of the layout ends when 1) no node overlap is present; AND 2) no nodes cross a river. We then increase s by one unit and repeat the algorithm until the average cartographic error, ϵ_c , a measure set by the user, reaches its maximum value $\epsilon_{c_{max}}$. The gradual size increase process provides stability to the layout, as can be seen in the accompanying video.

4.3. River Intersection Testing

The logic for translating the position of a node is detailed in Algorithm 2. When a node's position changes, we test if the node's trajectory intersects any segment of a river. See Algorithm 7. A bounding box intersection test between the edge defined by node translation and river edges can be performed to reduce the number of edge intersection tests required. Using the intersection test, we identify all nodes that cross the river as a result of the initial FNOR algorithm. We label these nodes, $\mathbf{n}_{n_r r}$.

4.4. Translating Rivers

For all nodes $\mathbf{n}_{n_r r}$ that cross a river, r , we compute an average vector \mathbf{v}_{avg} used to translate r . Whenever a node, \mathbf{n} , crosses a river, we store a vector $\overrightarrow{\mathbf{n}n_t}$ that points in the direction of the translation. We then use a heuristic to translate r using the average vector of node intersection

$$\mathbf{v}_{avg} = \sum_{i=1}^{\epsilon_t} \frac{\overrightarrow{\mathbf{n}_i n_i}}{\epsilon_t}$$

, where ϵ_t is the number of nodes intersecting the river. This step intends to create space for the next iteration of node translation without crossing a river. The detailed procedure for translating rivers is provided by Algorithm 4.

When a river, r , is translated by \mathbf{v}_{avg} , this can trigger a scenario where nodes are crossed by a translated river, denoted $\mathbf{n}_{r,n}$. As a heuristic, we also translate these nodes by \mathbf{v}_{avg} . The reasoning behind this is that \mathbf{v}_{avg} indicates which direction the river needs to be translated to create space for the nodes that are too crowded together. In practice, \mathbf{v}_{avg} is multiplied by a scaling factor $\propto \mathbf{v}_{avg}$. Thus, we can influence how far r is translated in each iteration of the layout algorithm. We can use this to ensure smooth transitions between iterations.

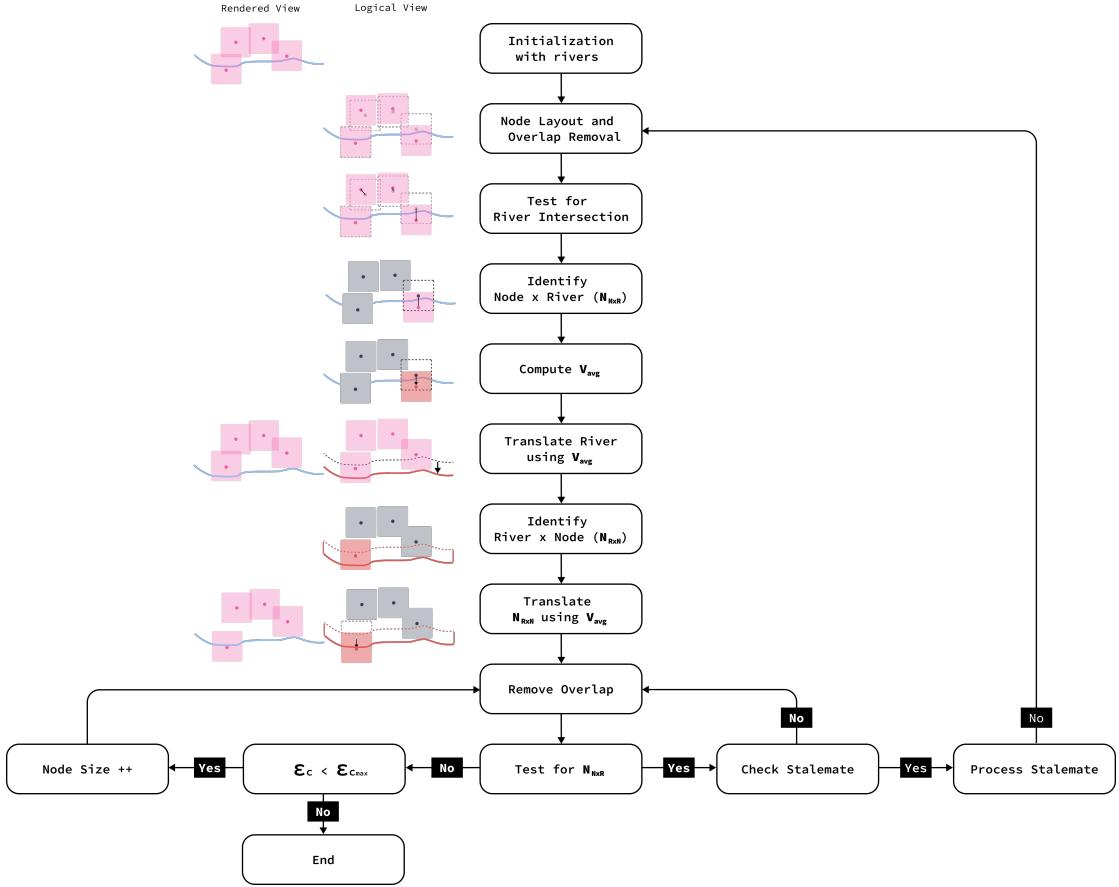


Figure 3: An overview of our hybrid layout algorithm incorporating rivers. See also Algorithm 3 in Section 4 for more detail. See Figure 5 for the logic of processing a stalemate. For illustration purposes, we show the rendered views alongside the logical views representing the actual computation.

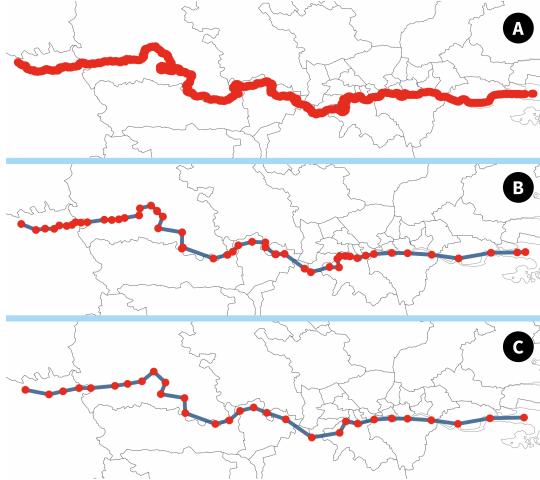


Figure 4: The resolution of rivers can be dynamically adjusted by the user. (A) shows River Thames at its original resolution with 10,170 edges. (B) shows the river at a reduced resolution of 49 edges. We further smooth the river by removing 19 vertices in dense areas, as shown in (C). The reduced resolution preserves the majority of River Thames' original shape and improves the performance of our river intersection tests.

4.5. Process Stalemates

As the FNOR always attempts to produce an optimal node layout where node distribution and translation are minimized,

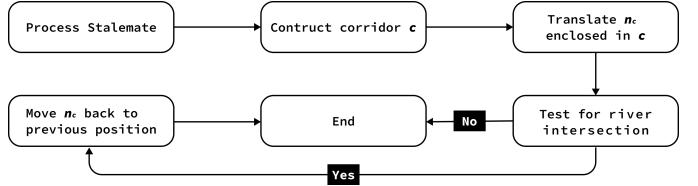


Figure 5: A flowchart illustration of stalemate processing. See Section 4.5, Figure 6, and Figure 7 for more detail.

a node's translation path can repeatedly intersect a river due to congestion, creating a stalemate situation, as shown in Figure 6. If a node is translated between two positions, n and n_t , for w iterations (a user-adjustable parameter), we introduce a heuristic solution: constructing a corridor to alleviate congestion. A corridor, c , is a rectangle with a width of c_w and a length of c_l , formed by deriving two edges $\overrightarrow{e_p^1}$ and $\overrightarrow{e_p^2}$ such that $\overrightarrow{e_p^1} \parallel \overrightarrow{e_p^2} \parallel \overrightarrow{n_i n_c}$ (See Figure 7C and D). All nodes enclosed by c are then translated by $\overrightarrow{n_t n}$ to alleviate the congestion (See Figure 7E). The procedure for constructing corridors is provided by Algorithm 6.

4.6. Terminating the Algorithm

The algorithm terminates when ϵ_c reaches ϵ_{cmax} , the error tolerance set by the user. We adopt the maximum cartographic

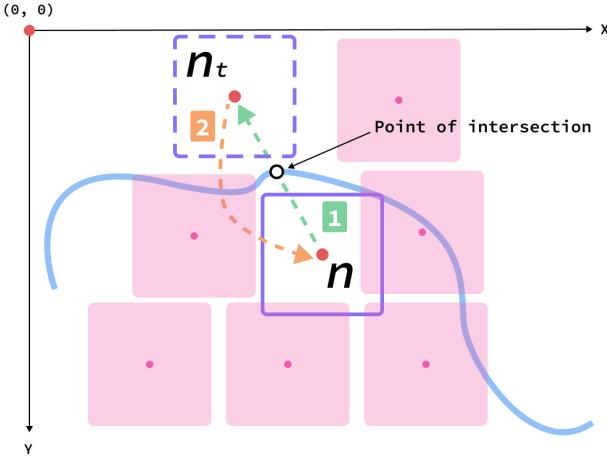


Figure 6: A stalemate: when a node’s translation path $\overrightarrow{nn_t}$ (iteration 1) intersects a river w times. The node is translated back to its previous position (iteration 2). A stalemate can occur when the area is congested and the node cannot translate to a new position without intersecting a river.

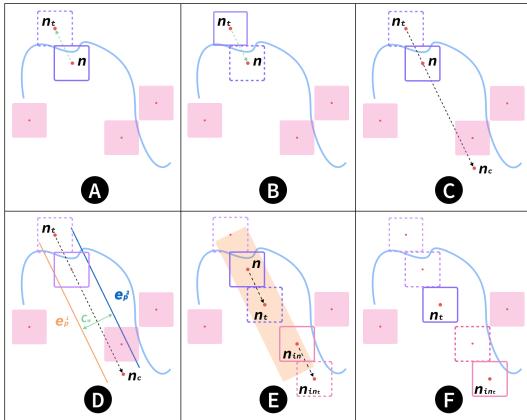


Figure 7: A stalemate occurs when a node’s translation path $\overrightarrow{nn_t}$ intersects a river for w times, as shown in (A). To address this, we derive a corridor (orange rectangle in (E)) based on n and n_t . All nodes within the corridor are translated based on $\overrightarrow{nn_t}$, such that $\overrightarrow{nn_t} = \overrightarrow{n_in n_{im}}$. For clarity in the illustration, we place nodes sparsely in this figure.

error from Alam et al. [45], namely:

$$\epsilon_{c_{max}} = \max_{n \in L} \frac{|\mathbf{n}_i - \mathbf{n}_{i_t}|}{\max\{\mathbf{n}_i, \mathbf{n}_{i_t}\}}$$

where \mathbf{n}_i and \mathbf{n}_{i_t} are the initial and translated regions in the cartogram, L represent the list of regions, and $\epsilon_{c_{max}}$ is a normalized value that we express as a percentage. For a detailed derivation of the formula, refer to the work by Alam et al. [45]. Because the algorithm processes node-river intersections, we can measure a novel kind of error, namely, topological error ϵ_t . We maintain $\epsilon_t = 0$, however, we can count how many nodes would have crossed a river if we did not test for this and simply let nodes cross rivers. We express ϵ_t in the normalized range

$\epsilon_t \in [0, 1]$, where $\epsilon_{t_{max}}$ is the case where all nodes cross a river.

When the algorithm terminates, the node layout is considered optimal where no nodes have crossed or crossed a river (denoted $\epsilon_t = 0$ and $\epsilon_t < \epsilon_{c_{max}}$). Every node remains on the same side of the river as its centroid.

4.7. User Options

Figure 8 presents an overview of the application, including user options. The user can adjust the following parameters:

Rendering Visibility: The rendering visibility of various elements, including the choropleth shapefile, rivers, nodes, and node centroids, can be toggled on and off.

Node Mapping: Both size and color of the nodes can be mapped to different EHR attributes or set to uniform.

Overlap Removal Speed: The overlap removal process can be observed in a step-by-step manner, or the algorithm can be run automatically.

Maximum Cartographic Error: The user can adjust the maximum cartographic error, $\epsilon_{c_{max}}$, which is used to terminate the algorithm.

River Translation: The behavior of rivers during the process can also be adjusted by the user: 1) Enable river crossing: this option allows nodes to cross rivers. Nodes cannot cross rivers by default; 2) Disable river translation: this option disables the translation of rivers, rivers are translated by default. Both options are useful for generating different layouts and to observe the behavior of the hybrid layout algorithm.

Corridor Length: The user can define the length of a corridor that is used to resolve stalemate situations. A longer corridor length allows more nodes to be translated when a stalemate occurs, but may result in a less accurate layout. The default corridor length is three times the max node size.

River Thickness and Resolution: Increasing the thickness of rivers may improve the recognizability of the cartogram. Similarly, increasing the resolution of rivers, at the expense of the speed of node-river intersection test, may produce a layout with higher legibility.

5. User-Centred Evaluation

We conduct a within-subject user study to evaluate the effectiveness of our approach. We chose this type of evaluation because legibility is a human-centred characteristic. Many different types of statistical error metrics have been evaluated in previous work [13], however, our focus is more user-centric in nature.

5.1. Study Hypothesis

We formulate the following hypotheses to motivate our user study:

H1: The introduction of rivers can improve the legibility and recognizability of cartograms.

H1.1: The introduction of rivers can increase the accuracy of the target location.

H1.2: The introduction of rivers can reduce the time required to locate a target region in a cartogram.

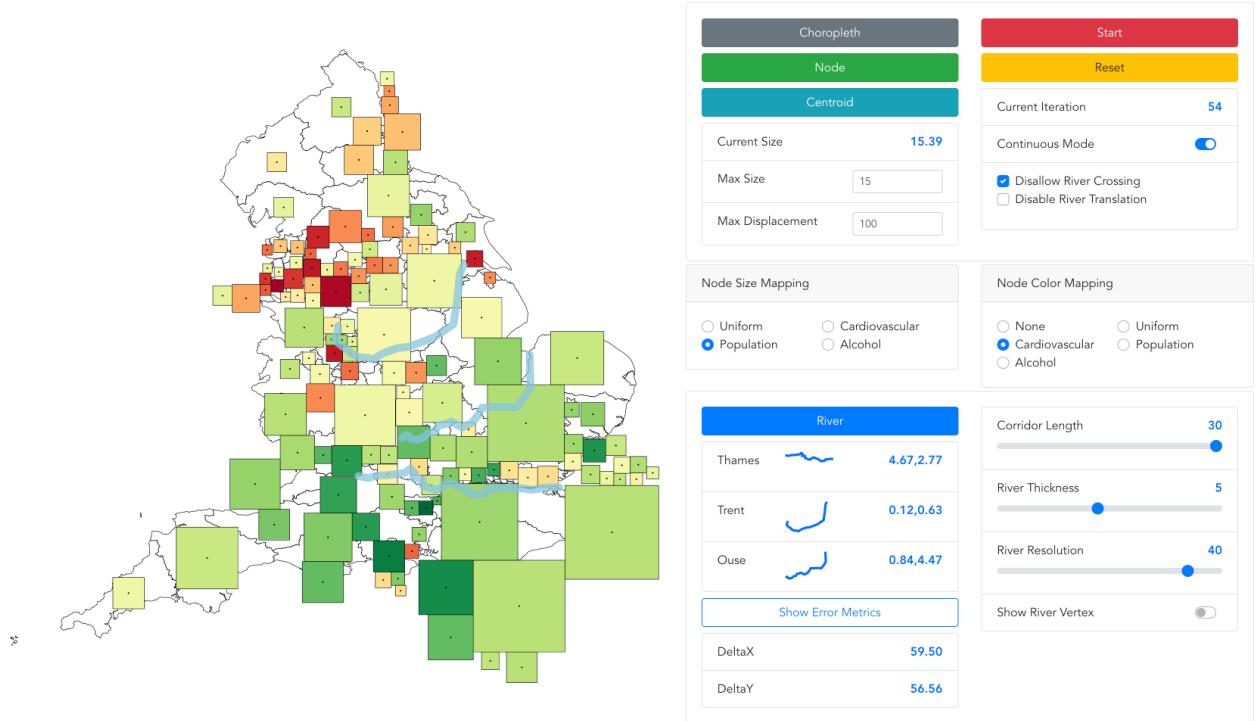


Figure 8: A screenshot of the user interface. User options are provided to adjust the terminating conditions (size and error), color mapping, visibility of nodes and rivers. Other options include the ability to control the overlap removal behavior: rivers can be static or dynamic. See Section 4.7 for more on user options. In this figure, $\epsilon_{c_{max}} = 1.875\%$, and an ϵ_t of 9.63% is eliminated.

We test these hypotheses using location-based tasks. Given a standard choropleth on the left side of the screen and the corresponding cartogram on the right side, we ask user study participants to locate the given region in the choropleth (on the left) in the cartogram (on the right). See Figure 9.

5.2. User Study Variables

In this section, we discuss the variables of our user study.

Independent Variables: The primary independent variable is the presence of rivers in the cartograms, which directly impacts the final layout of the cartogram. The identical set of rivers is either rendered or hidden collectively.

Dependent Variables: *Accuracy:* Given a CCG location in the choropleth on the left, we ask the participant to locate the corresponding node in the cartogram. Accuracy as the primary dependent variable is measured by the number of correct CCGs located and chosen by the participants. *Response time:* Another dependent variable is the time it takes the participants to complete each task.

Control Variables: *Choice of color map:* We use D3’s built-in interpolateRdYlGn color map, a color scheme of red, yellow, and green, to depict the data in our cartograms. *Communicating the target CCG to the user:* We inform the participant about the target CCG they need to find. The target CCG is shown in the form of a nonstop blinking (between its original color and black) area on the screen, each blink is given a duration of 2 seconds.

5.3. User Study Design

In this section we describe the user study participants, datasets, and the experimental procedure. The study was approved by the University of Nottingham’s Research Ethics Committee (Ref: 2021-2022-001).

Participants: We recruited 24 participants.

- Gender: 10 females and 14 males
- Age Group: 18-24 (16), 25-29 (8)
- Education: 1 PhD, 11 Master’s, 8 Bachelor’s, 4 Others

Datasets: We used the following EHR datasets for our evaluation:

- Population
- Under 75 mortality from cardiovascular disease
- Emergency admissions for alcohol-related liver disease
- Alcohol-specific admission and readmission

135 CCGs are rendered on the screen as a choropleth on the left. We then generated another view on the right using cartograms. The color of both visual designs is mapped to a data dimension. No cartograms were used more than once per participant. See Figure 9 for an example.



Figure 9: A sample location task for participants. The left shows the choropleth map, and the right shows the corresponding cartogram. Both images show the three longest rivers in England, and color is mapped to CCG population. The target CCG blinks on the choropleth (shown in black), and participants are asked to identify this CCG on the cartogram. In this figure, $\epsilon_{tmax} = 1.875\%$, and an ϵ_t of 6.67% is eliminated.

5.4. Procedure

Due to pandemic restrictions, we designed the user study to be carried out online. Participants are expected to use their retail hardware for the tasks. However, devices with small screens are not recommended because of the large size of the cartograms that are rendered. The within-subject user study includes four parts:

P1: The participants are asked to listen to instructions and training provided as both text and videos. The instructions are designed to help participants understand the concepts used in the tasks. An instructional video was provided¹.

P2: The participants are given three practice tasks to familiarize themselves with the user study design. A sample task² is shown in Figure 9 and included in the online materials³. The practice tasks are identical to the actual tasks, accompanied by instructions provided before the actual tasks commence. The results from these practice tasks are not included in the final analysis. These tasks also serve as a quality test to see whether users are taking the study seriously. A demonstration of three sample tasks is also included in the instructional video¹.

P3: The participants are asked to complete 16 location tasks that involve 4 target CCGs. See the online materials for exact locations of target CCGs³. Response and reaction time are recorded. These 4 CCGs are selected to avoid extreme cases (thus bias the result), in terms of size, color, and location.

P4: The participants are asked to complete a questionnaire³ that consists of 5-Point Likert Scale questions and open-ended questions.

5.5. User Study Results

In this section, we delve into the analysis of the preliminary user study conducted. Our study involved the recruitment

¹Instructional video: <https://www.youtube.com/playlist?list=PLL7sHvxLtD75fMtrUQrAddjt3wffFkcWz>

²A sample task is accessible at <https://ghr.wangqiru.com/#/P1>.

³The materials for P1-P4 are available on the Open Science Foundation website at <https://osf.io/q39w7>.

of 24 participants, each tasked with completing 16 location-related tasks. As a result, we collected a total of 384 responses. However, we excluded 24 responses from our analysis as they took over 60 seconds to complete. We hypothesize that these prolonged response times indicate potential distractions experienced by participants during the study.

Accuracy: Figure 10 shows the accuracy measured by the number of correct CCGs chosen by the participants. The chart shows that the introduction of rivers has improved the accuracy of two CCGs. The accuracy for the two groups differs significantly according to the two-sample unequal variance *t*-test performed, $t(262.98) = 3.76$, $p = 0.0002$. This supports our hypothesis H1.1 that the introduction of rivers increases target location accuracy.

Response Time: Figure 11 shows the average time to locate a CCG measured for four CCGs. The chart indicates that the introduction of rivers has slightly reduced the time needed to locate CCGs. The two groups show no statistically significant differences according to the two-sample unequal variance *t*-test performed, $t(356.03) = 0.87$, $p = 0.38$. However, the average time to locate a given CCG is reduced for 3 out of the 4 CCGs users in the study.

Participant Feedback: Figure 12 shows the results of the following Likert Scale questions:

- (A) 87.5% of participants agree that including rivers is useful.
- (B) 83.3% of participants agree that rivers increase the legibility of a cartogram.
- (C) 95.8% of participants agree that including rivers makes cartograms easier to understand.
- (D) 62.5% of participants agree that including rivers makes CCGs easier to locate.
- (E) 87.5% of participants agree that including rivers adds value to the standard cartogram.

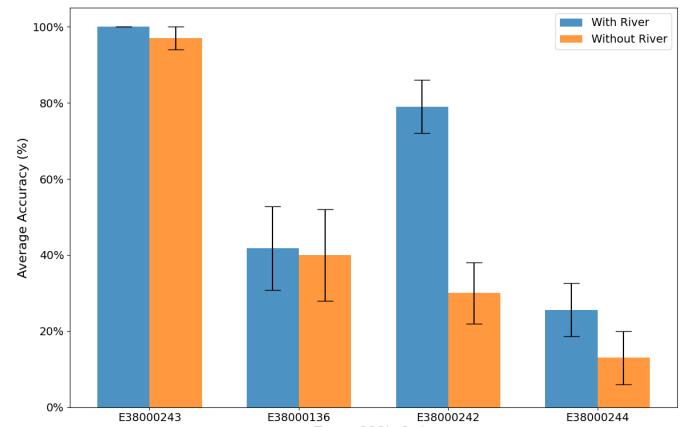


Figure 10: In the bar chart, the y-axis shows the average accuracy rate of locating CCGs with and without rivers for user study participants. The x-axis shows the four target CCGs, as described in Procedure P3.

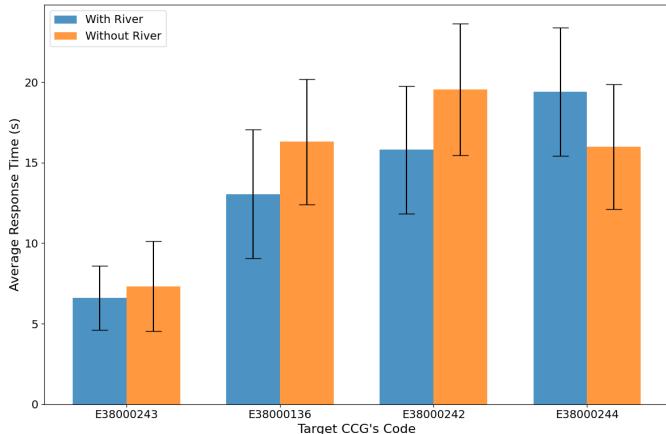


Figure 11: In the bar chart, the y-axis shows the average response time for locating CCGs with and without rivers for user study participants. The x-axis shows the four target CCGs, as described in Procedure P3.

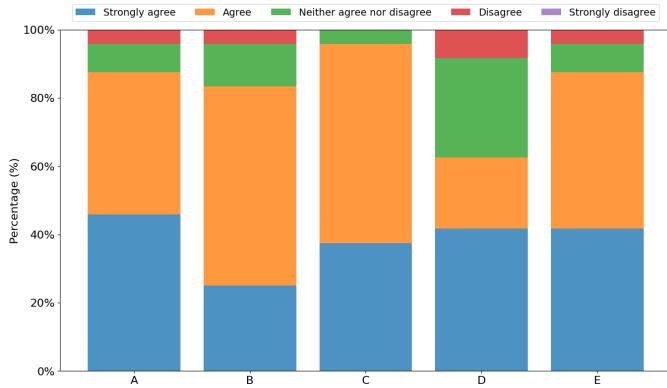


Figure 12: The stacked bar chart shows the user study participant responses of Likert Scale questions.

The qualitative results clearly support our hypothesis H1 that rivers generally increase the legibility of cartograms.

6. Limitations and Future Work

In this section, we discuss some limitations of our work and future research directions. Because this is a new concept, it opens the door for many future research directions.

6.1. Color map choice

The first limitation is the color map that we use to depict the data in our cartograms. We use D3’s built-in interpolateRdYIGn color map, a diverging color scheme of red, yellow, and green. However, we believe that the choice of color map can have a significant impact on the legibility of cartograms. In the user study, we carefully avoid extreme values where the location or color of the CCG makes it easier to locate the target. See Figure 13 for an example. We plan to explore the impact of different color maps on the legibility of cartograms in future work.

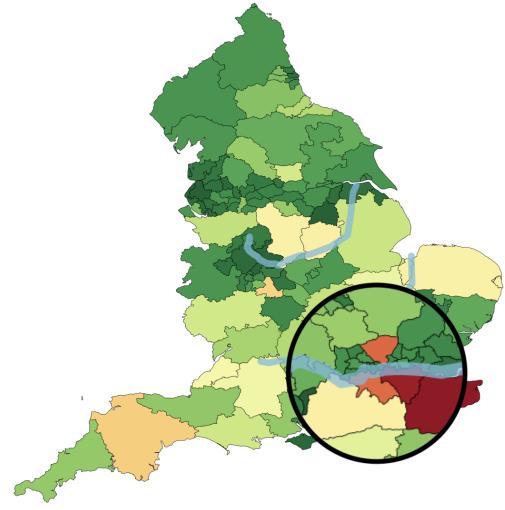


Figure 13: Due to color and relative location, we believe the CCGs in the black circle are easier to locate.

6.2. Overlap removal algorithm choice

Another limitation is the algorithm (FNOR) we use. We believe that developing a new algorithm with built-in constraint support can significantly reduce the time required to generate cartograms with rivers. Currently, the runtime of our layout algorithm is approximately 30 milliseconds for each iteration. When the quantity of nodes and features increases, generating the optimal layout demands several hundred iterations.

6.3. Generalizability

Future work also includes generalizations and extensions of the algorithm, e.g., the use of other features in the cartogram layout such as additional rivers, major highways, lakes, and coastlines, etc. We also consider whether increasing the length of the rivers as the size of the nodes increases would be a useful option. We would like to explore the case of river-river intersections (or confluence) and testing out more geographies such as the U.S. and Europe, which have more complex rivers. We also considered the idea of deforming the rivers as part of the layout algorithm, however, this idea is open to future work.

6.4. Improved User Study

Due to pandemic restrictions, the user study did not take place in a controlled environment. In the future, we aim to execute a more controlled laboratory study, ensuring a controlled environment and standardized hardware for participants.

7. Conclusions

In this paper, we present a novel hybrid layout approach to generate non-contiguous cartograms with rivers. We first propose a new algorithm to generate cartograms with rivers, and then we present a prototype to support the exploration of cartograms with rivers. We evaluate our approach with a preliminary user study, and the results support our hypotheses: the introduction of rivers improves the legibility and recognizability

of cartograms, although a deeper analysis is necessary to reach more confident conclusions. We also discuss the limitations of our work and future research directions.

8. Acknowledgements

This work is funded by the grant EP/S010238/2 from the Engineering and Physical Sciences Research Council (EPSRC). EPSRC is a British Research Council that provides government funding for grants to undertake research and postgraduate degrees in engineering and the physical sciences.

References

- [1] B. D. Dent, J. Torguson, T. W. Hodler, *Cartography: Thematic Map Design*, 6th ed ed., McGraw-Hill Higher Education, New York, 2009.
- [2] R. Inoue, A New Construction Method for Circle Cartograms, *Cartography and Geographic Information Science* 38 (2011) 146–152. doi:10.1559/15230406382146.
- [3] R. Harris, M. Charlton, C. Brunsdon, Mapping the changing residential geography of White British secondary school children in England using visually balanced cartograms and hexograms, *Journal of Maps* 14 (2018) 65–72. doi:10.1080/17445647.2018.1478753.
- [4] A. Arranz-López, J. A. Soria-Lara, A. Ariza-Álvarez, An end-user evaluation to analyze the effectiveness of cartograms for mapping relative non-motorized accessibility, *Environment and Planning B: Urban Analytics and City Science* 48 (2021) 2880–2897. doi:10.1177/2399808321991541.
- [5] F. Pappenberger, H. L. Cloke, C. A. Baugh, Cartograms for Use in Forecasting Weather-Driven Natural Hazards, *The Cartographic Journal* 56 (2019) 134–145. doi:10.1080/00087041.2018.1534358.
- [6] S. Park, G. Lee, J. O. Kim, Flood Evacuation Mapping Using a Time-Distance cartogram, *ISPRS International Journal of Geo-Information* 9 (2020) 207. doi:10.3390/ijgi9040207.
- [7] M. Galluzzi, D. Rocchini, R. Canullo, R. E. McRoberts, G. Chirici, Mapping uncertainty of ICP-Forest biodiversity data: From standard treatment of diffusion to density-equalizing cartograms, *Ecological Informatics* 48 (2018) 281–289. doi:10.1016/j.ecoinf.2018.06.005.
- [8] D. Rocchini, M. Marcantonio, G. Arhonditsis, A. L. Cacciato, H. C. Hauffe, K. S. He, Cartogramming uncertainty in species distribution models: A Bayesian approach, *Ecological Complexity* 38 (2019) 146–155. doi:10.1016/j.ecocom.2019.04.002.
- [9] A. Breitzman, Using Cartograms to Visualize Population Normalized Big-Data Sets, in: 2018 IEEE International Conference on Big Data (Big Data), IEEE, Seattle, WA, USA, 2018, pp. 3575–3580. doi:10.1109/BigData.2018.8622217.
- [10] I. Alieva, How American Media Framed 2016 Presidential Election Using Data Visualization: The Case Study of the New York Times and the Washington Post, *Journalism Practice* (2021) 1–27. doi:10.1080/17512786.2021.1930573.
- [11] P. Gao, H. Zhang, Z. Wu, J. Wang, Visualising the expansion and spread of coronavirus disease 2019 by cartograms, *Environment and Planning A: Economy and Space* 52 (2020) 698–701. doi:10.1177/0308518X20910162.
- [12] D. E. Sack, S. J. Gange, K. N. Althoff, A. C. Pettit, A. N. Kheshti, I. S. Ransby, J. J. Nelson, M. M. Turner, T. R. Sterling, P. F. Rebeiro, Visualizing the geography of HIV observational cohorts with density-adjusted cartograms, *JAIDS Journal of Acquired Immune Deficiency Syndromes Publish Ahead of Print* (2021). doi:10.1097/QAI.0000000000002903.
- [13] S. Nusrat, S. Kobourov, The State of the Art in Cartograms, *Computer Graphics Forum* 35 (2016) 619–642. doi:10.1111/cgf.12932.
- [14] T. Munzner, *Visualization Analysis and Design*, 0 ed., A K Peters/CRC Press, 2014. doi:10.1201/b17511.
- [15] S. Nickel, M. Sondag, W. Meulemans, S. G. Kobourov, J. Peltonen, M. Nollenburg, Multicriteria Optimization for Dynamic Demers Cartograms, *IEEE Transactions on Visualization and Computer Graphics* (2022) 1–1. doi:10.1109/TVCG.2022.3151227.
- [16] B. Ian, D. Steve, C. Keith, Cartogram home, 2002. URL: http://www.ncgia.ucsb.edu/projects/Cartogram_Central/index.html.
- [17] B. Warf, M. Winsberg, The Geography of Religious Diversity in the United States, *The Professional Geographer* 60 (2008) 413–424. doi:10.1080/00330120802046786.
- [18] H. Sun, Z. Li, Effectiveness of Cartogram for the Representation of Spatial Data, *The Cartographic Journal* 47 (2010) 12–21. doi:10.1179/000870409X12525737905169.
- [19] P. Cruz, Adapted dorling cartogram on wage inequality in Portugal, in: 2017 IEEE VIS Arts Program (VISAP), IEEE, Phoenix, AZ, 2017, pp. 1–5. doi:10.1109/VISAP.2017.8282365.
- [20] C. Tong, L. McNabb, R. S. Laramee, Cartograms with Topological Features, in: *Computer Graphics & Visual Computing*, 2018.
- [21] S. Nusrat, J. Alam, S. Kobourov, Recognition and Recall of Geographic Data In Cartograms, in: Proceedings of the International Conference on Advanced Visual Interfaces, ACM, Salerno Italy, 2020, pp. 1–9. doi:10.1145/3399715.3399873.
- [22] D. Dorling, Area Cartograms: Their Use and Creation, in: M. Dodge, R. Kitchin, C. Perkins (Eds.), *The Map Reader*, John Wiley & Sons, Ltd, Chichester, UK, 2011, pp. 252–260. doi:10.1002/9780470979587.ch33.
- [23] R. G. Cano, K. Buchin, T. Castermans, A. Pieterse, W. Sonke, B. Speckmann, Mosaic Drawings and Cartograms, *Computer Graphics Forum* 34 (2015) 361–370. doi:10.1111/cgf.12648.
- [24] E. Raisz, The Rectangular Statistical Cartogram, *Geographical Review* 24 (1934) 292. doi:10.2307/208794. arXiv:208794.
- [25] M. van Kreveld, B. Speckmann, On Rectangular Cartograms, in: D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, S. Albers, T. Radzik (Eds.), *Algorithms – ESA 2004*, volume 3221, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004, pp. 724–735. doi:10.1007/978-3-540-30140-0_64.
- [26] M. Bearak, L. Gamio, Everything you ever wanted to know about the U.S. foreign assistance budget, *Washington Post* (2016). URL: <https://www.washingtonpost.com/graphics/world/which-countries-get-the-most-foreign-aid/>.
- [27] L. Gamio, Election maps are telling you big lies about small things, *Washington Post* (2016). URL: <https://www.washingtonpost.com/graphics/politics/2016-election/how-election-maps-lie/>.
- [28] A. Taylor, What a real ‘Brexit Britain’ would look like, *Washington Post* (2016). URL: <https://www.washingtonpost.com/news/worldviews/wp/2016/06/29/what-a-real-brexit-britain-would-look-like/>.
- [29] G. Miller, Election Maps Can Be Misleading—Here’s a Solution, *National Geographic* (2016). URL: <https://www.nationalgeographic.com/culture/article/improved-election-map-cartograms>.
- [30] M. Stabe, The search for a better US election map, *Financial Times* (2016). URL: <https://www.ft.com/content/3685bf9e-a4cc-11e6-8b69-02899e8bd9d1>.
- [31] M. Sandberg, Cartogram: House Election Results: Democrats Take Control (The New York Times), 2018. URL: <https://datavizblog.com/2018/11/14/cartogram-house-election-results-democrats-take-control-the-new>
- [32] The Learning Network, What’s Going On in This Graph? — 2020 Presidential Election Maps, *The New York Times* (2020). URL: <https://www.nytimes.com/2020/11/19/learning/whats-going-on-in-this-graph-2020-presidential-election-maps.html>.
- [33] A. McCartney, B. Harris, M. Rojanasakul, J. Burgess, P. Murray, A. Vann, D. Pogkas, B. Benhamou, A. Kessler, A. Tribou, 2020 Presidential Election Results: Live Updates, *Bloomberg* (2020). URL: <https://www.bloomberg.com/graphics/2020-us-election-results>.
- [34] Open Geography portalx, Open Geography portalx, 2022. URL: <https://geoportal.statistics.gov.uk/datasets/d6acd30ad71f4e14b4de808e58d9bc4c>.
- [35] Q. Wang, R. Laramee, EHR STAR: The State-Of-the-Art in Interactive EHR Visualization, *Computer Graphics Forum* (2021) cgf.14424. doi:10.1111/cgf.14424.
- [36] NHS, NHS Clinical commissioning groups (CCGs), 2022.

- URL: <https://www.england.nhs.uk/commissioning/who-commissions-nhs-services/>.
- [37] OpenStreetMap, Relation: Thames (2263653), 2022. URL: <https://www.openstreetmap.org/relation/2263653>.
- [38] Overpass Turbo, Overpass turbo, 2022. URL: <https://overpass-turbo.eu/>.
- [39] QGIS, Welcome to the QGIS project!, 2022. URL: <https://qgis.org/en/site/>.
- [40] M. Bloch, Mapshaper, 2022. URL: <https://mapshaper.org/>.
- [41] TopoJSON, The TopoJSON Format Specification, 2022. URL: <https://github.com/topojson/topojson-specification>.
- [42] NHS Digital, Clinical Commissioning Group Outcomes Indicator Set (CCG OIS), 2022. URL: <https://digital.nhs.uk/data-and-information/publications/statistical/ccg-outcomes-indicator-set>.
- [43] T. Dwyer, K. Marriott, P. J. Stuckey, Fast Node Overlap Removal, in: D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, P. Healy, N. S. Nikolov (Eds.), Graph Drawing, volume 3843, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 153–164. doi:10.1007/11618058_15.
- [44] F. Chen, L. Piccinini, P. Poncelet, A. Sallaberry, Node Overlap Removal Algorithms: An Extended Comparative Study, Journal of Graph Algorithms and Applications 24 (2020) 683–706. doi:10.7155/jgaa.00532.
- [45] M. J. Alam, S. G. Kobourov, S. Veeramoni, Quantitative Measures for Cartogram Generation Techniques, Computer Graphics Forum 34 (2015) 351–360. doi:10.1111/cgf.12647.

Appendix A. Obtaining Shapefile

```
relation(2263653);>>;
// River Great Ouse: 2798097
// River Trent: 2863468
out skel;
```

Listing 1: The query that downloads the shapefile of River Thames from OpenStreetMap via the Overpass Turbo API.

Appendix B. Pre-processing Shapefiles

Shapefiles from different sources are likely to be incompatible. In our case, the NHS CCG shapefile is incompatible with the river shapefiles. The major reason for the incompatibility is the coordinate reference system (CRS). The CRS of the CCG shapefile is EPSG:27700 (OSGB36 - British National Grid). The CRS of the river shapefiles is EPSG:4326 (WGS84 - World Geodetic System). Here, we provide some pre-processing steps using QGIS (version: 3.26.0-Buenos Aires) [39] to handle the incompatibility and reduce shapefile size to improve performance.

Appendix B.1. Import Shapefiles into QGIS

We first load all three river shapefiles into QGIS Figure B.14, followed by the CCG shapefile Figure B.15.

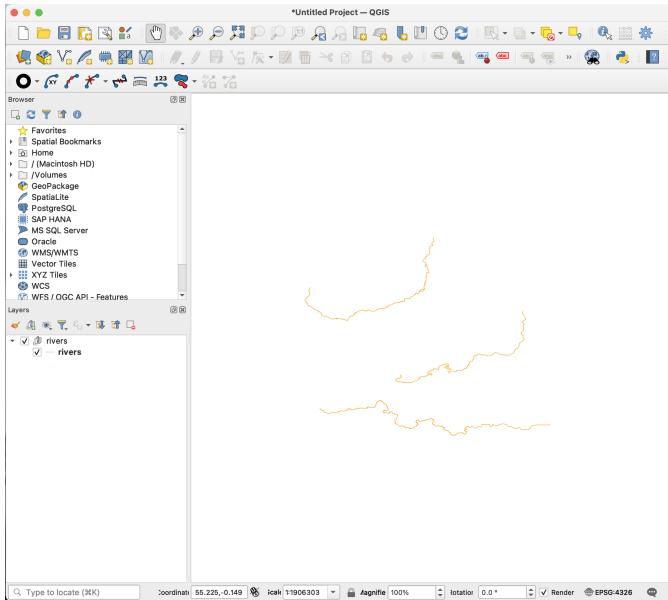


Figure B.14: QGIS interface, with River Trent, River Great Ouse, and River Thames (from top to bottom) imported.

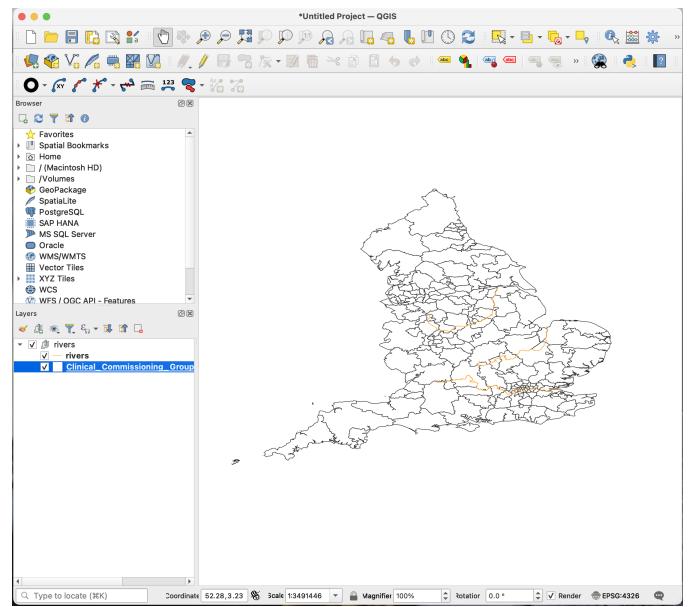


Figure B.15: QGIS interface, with all NHS CCGs imported.

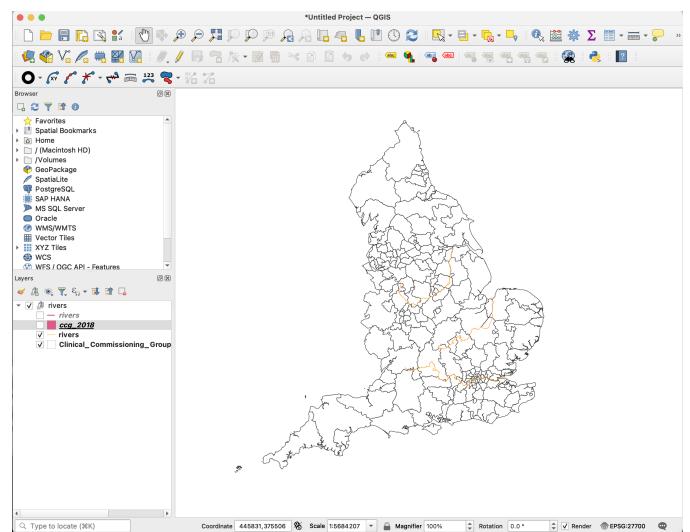


Figure B.16: QGIS interface, showing the unified CRS (OSGB36) for both layers.

Appendix B.2. Export Shapefiles in GeoJSON and Unify the Coordinate Reference System (CRS)

We then use QGIS to unify the CRS, and export both layers in GeoJSON format. See Figure B.17 and Figure B.18. The unified layer is shown in Figure B.16.

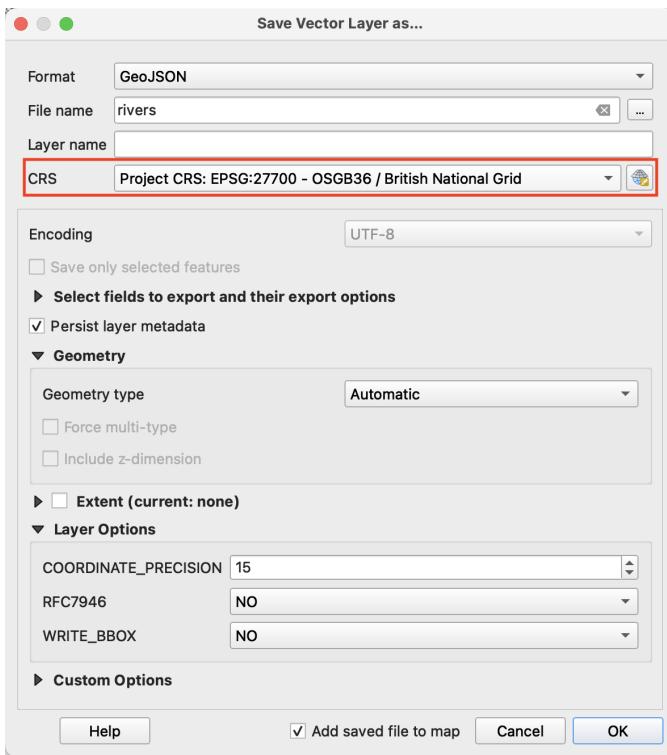


Figure B.17: QGIS interface, exporting all rivers using the OSGB36 CRS in GeoJSON.

Appendix B.3. Merge Shapefiles and Reduce File Size

We then merge two layers into one layer, and export it in the TopoJSON format using Mapshaper [40]. Mapshaper also supports the simplification of GeoJSON shapefiles. See Figure B.19.

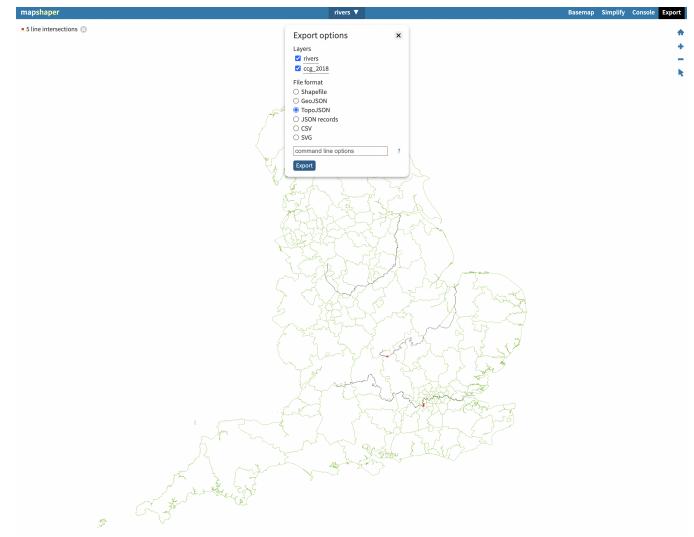


Figure B.19: Mapshaper interface, merging all rivers with NHS CCGs into one layer, and export the merged layer in TopoJSON.

Appendix B.4. Pre-processing Result

Table B.3 shows the preprocessing result. The reduction in file size is significant and greatly reduces the initialization time of our implementation.

Shapefile	Original	GeoJSON	TopoJSON
Rivers	2.0 MB (GeoJSON)	1.4 MB	-
NHS CCGs	46.6 MB (.shp, Esri vector shapefile)	140.2 MB	-
Merged	-	-	16.3 MB

Table B.3: The file size is reduced by 88.5% from the original size.

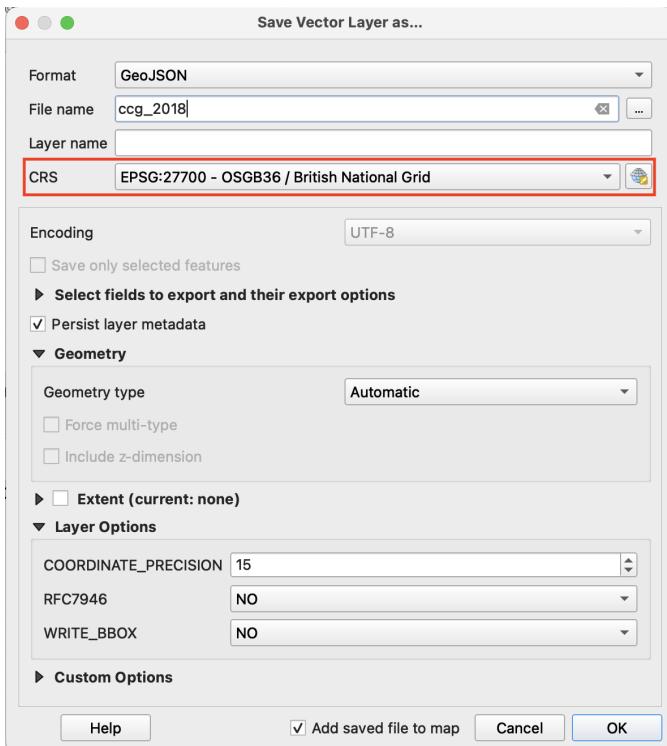


Figure B.18: QGIS interface, exporting all NHS CCGs using the OSGB36 CRS in GeoJSON.

Appendix C. Procedure: DeriveParallelEdge

Algorithm 1 Procedure to derive an edge, \bar{e}_p , that is parallel to \bar{e} with a distance of d .

Input:

\bar{e} \leftarrow the edge used to derive the parallel edge \bar{e}_p
 $d \leftarrow$ the shortest distance between \bar{e} and \bar{e}_p

Output:

An edge, \bar{e}_p , that is parallel to \bar{e} with a distance of d .

Local variables:

$\Delta x, \Delta y \leftarrow$ the differences in x, y for $\bar{e}.start$ and $\bar{e}.end$
 $scale \leftarrow$ the scale of $\frac{d}{\sqrt{\Delta x^2 + \Delta y^2}}$

```

1: procedure DERIVEPARALLELEDGE( $\bar{e}, d$ )
2:    $\Delta x \leftarrow \bar{e}.start.x - \bar{e}.end.x$ 
3:    $\Delta y \leftarrow \bar{e}.start.y - \bar{e}.end.y$ 
4:    $scale \leftarrow \frac{d}{\sqrt{\Delta x^2 + \Delta y^2}}$ 
5:    $\bar{e}_p.start.x \leftarrow scale \cdot (-\Delta y) + \bar{e}.start.x$ 
6:    $\bar{e}_p.start.y \leftarrow scale \cdot \Delta x + \bar{e}.start.y$ 
7:    $\bar{e}_p.end.x \leftarrow scale \cdot (-\Delta y) + \bar{e}.end.x$ 
8:    $\bar{e}_p.end.y \leftarrow scale \cdot \Delta x + \bar{e}.end.y$ 
9:   return  $\bar{e}_p$ 
10: end procedure

```

Appendix D. Procedure: TranslateNode

Algorithm 2 Procedure to translate node positions.

Input:

$L \leftarrow$ a list of n representing regions

Output:

$\epsilon_t \leftarrow$ the number of nodes crossing the river in the input

Global variables:

$w \leftarrow$ the maximum number of iterations indicating a stalemate

Local variables:

$n \leftarrow$ a node is an object with the following properties:
 $n.x, n.y$, or $n(x, y) \leftarrow$ the x and y coordinates of n
 $n.cross \leftarrow$ the number of times that n crosses a river
 $n_p \leftarrow$ the previous position of n
 $n_t \leftarrow$ the translated position of n

```

1: procedure TRANSLATENODE( $L$ )
2:    $\epsilon_t \leftarrow 0$ 
3:   for each  $n \in L$  do
4:     if  $n(x, y) \neq n_t(x, y)$  then
5:        $n(x, y) \leftarrow n_t(x, y)$ 
6:       if TESTINTERSECTION( $\overline{nn_t}$ ) = True then
7:          $n.cross + +$ 
8:          $\epsilon_t + +$ 
9:         if  $n.cross < w$  then
10:            $\triangleright$  Translate back to previous position
11:            $n(x, y) \leftarrow n_p(x, y)$ 
12:         else
13:           PROCESSSTALEMATE( $n, n_t$ )
14:            $n.cross \leftarrow 0$   $\triangleright$  Reset counter
15:         end if
16:         end if
17:       end if
18:     end for
19:   return  $\epsilon_t$ 
20: end procedure

```

Appendix E. Procedure: UpdateLayout

Algorithm 3 Procedure to adjust river positions, remove node overlap and prevent nodes from crossing rivers. See Section 4.

Global variables:

$L \leftarrow$ a list of n representing regions with the following properties:

$L.cross \leftarrow$ the number of n in L that crosses a river

$s \leftarrow$ the initial size of all nodes

$\epsilon_c \leftarrow$ the average cartographic error of all nodes

$\epsilon_{max} \leftarrow$ the maximum cartographic error of all nodes

$w \leftarrow$ the maximum number of iterations indicating a stalemate

Local variables:

$n \leftarrow$ a node is an object with the following properties:

$n.x, n.y,$ or $n(x, y) \leftarrow$ the x and y coordinates of n

$n.cross \leftarrow$ the number of times that n crosses a river

$n_p \leftarrow$ the previous position of n

$n_t \leftarrow$ the translated position of n

```

1: procedure UPDATERAYOUT
2:    $s \leftarrow 1$ 
3:   while  $\epsilon_c < \epsilon_{max}$  do
4:      $L.cross \leftarrow 1$             $\triangleright$  Trigger the while loop
5:     while  $L.cross > 0$  do
6:        $L.cross \leftarrow 0$ 
7:        $L \leftarrow$  RemoveOverlap( $L$ )
8:       TRANSLATERIVER( $L$ )
9:        $L.cross \leftarrow$  TRANSLATENODE( $L$ )
10:    end while
11:     $s + +$ 
12:  end while
13: end procedure

```

Appendix F. Procedure: TranslateRiver

Algorithm 4 Procedure to translate rivers.

Input:

$L \leftarrow$ a list of n representing regions

Local variables:

$R \leftarrow$ a list of r representing river features

$n \leftarrow$ a node is an object with the following properties:

$n.x, n.y,$ or $n(x, y) \leftarrow$ the x and y coordinates of n

$n_p \leftarrow$ the previous position of n

$n_t \leftarrow$ the translated position of n

$\epsilon_t \leftarrow$ the number of nodes intersecting r

```

1: procedure TRANSLATERIVER( $L$ )
2:   for each  $r \in R$  do
3:      $\epsilon_i \leftarrow 0$ 
4:      $\vec{v}_r \leftarrow (0, 0)$             $\triangleright$  Hold the sum of vectors  $\overrightarrow{nn}_t$ 
5:     for each  $n \in L$  do
6:       if  $n(x, y) \neq n_t(x, y)$  then
7:         if TESTINTERSECTION( $\overrightarrow{nn}_t, r$ ) = True then
8:            $\vec{v}_r \leftarrow \vec{v}_r + \overrightarrow{nn}_t$ 
9:            $\epsilon_t + +$ 
10:      end if
11:    end if
12:   end for
13:   Translate river  $r$  by the average vector  $\frac{\vec{v}_r}{\epsilon_i}$ 
14: end for
15: end procedure

```

Appendix G. Procedure: DerivePoint

Algorithm 5 Procedure to derive a point based an edge and a distance.

Input:

$\bar{e} \leftarrow$ the edge used to derive the new point

$d \leftarrow$ the distance between n_c and $\bar{e}.start$

Output:

A point, n_c , that is distance d away from $\bar{e}.start$.

Local variables:

$\Delta x, \Delta y \leftarrow$ the differences in x, y for $\bar{e}.start$ and $\bar{e}.end$

```

1: procedure DERIVEPOINT( $\bar{e}, d$ )
2:    $\Delta x \leftarrow \bar{e}.start.x - \bar{e}.end.x$ 
3:    $\Delta y \leftarrow \bar{e}.start.y - \bar{e}.end.y$ 
4:    $n_c.x \leftarrow \frac{\Delta x}{\sqrt{\Delta x^2 + \Delta y^2}} \cdot d$ 
5:    $n_c.y \leftarrow \frac{\Delta y}{\sqrt{\Delta x^2 + \Delta y^2}} \cdot d$ 
6:   return  $n_c$ 
7: end procedure

```

Appendix H. Procedure: ProcessStalemate

Algorithm 6 Procedure to derive a corridor to resolve stalemate. We use an SVG canvas, where the point of origin (0,0) is located at the top left corner, with the x-axis extending to the right and the y-axis extending downwards (See Figure 6).

Input:

$n \leftarrow$ the node used to derive the corridor

Global variables:

$c_l \leftarrow$ the length of a corridor

$c_w \leftarrow$ the width of a corridor

Local variables:

$c \leftarrow$ the corridor

$\mathbf{n}_c \leftarrow$ the point extending $\overrightarrow{\mathbf{n}_t \mathbf{n}}$ such that $|\mathbf{n}_t \mathbf{n}_c| = c_l$

$\overline{e_{p^1}}, \overline{e_{p^2}} \leftarrow$ the edges parallel to $\overrightarrow{\mathbf{n}_t \mathbf{n}_c}$

$\text{corridor} \leftarrow$ a rectangle formed by $\overline{e_{p^1}}$ and $\overline{e_{p^2}}$

```

1: procedure PROCESSSTALEMATE( $n$ )
2:    $\mathbf{n}(x, y) \leftarrow \mathbf{n}_p(x, y)$ 
3:    $\mathbf{n}_c \leftarrow \text{DERIVEPOINT}(\overrightarrow{\mathbf{n}_t \mathbf{n}}, c_l)$ 
4:    $\overline{e_{p^1}} \leftarrow \text{DERIVEPARALLELEDGE}(\overrightarrow{\mathbf{n}_t \mathbf{n}_c}, \frac{c_w}{2})$ 
5:    $\overline{e_{p^2}} \leftarrow \text{DERIVEPARALLELEDGE}(\overrightarrow{\mathbf{n}_t \mathbf{n}_c}, -\frac{c_w}{2})$ 
6:    $c \leftarrow \begin{bmatrix} \overline{e_{p^1}}.\text{start} & \overline{e_{p^1}}.\text{end} \\ \overline{e_{p^2}}.\text{start} & \overline{e_{p^2}}.\text{end} \end{bmatrix}$ 
7:   for each  $\mathbf{n}_{in}$  inside  $c$  do
8:      $\overrightarrow{\mathbf{n}_{in} \mathbf{n}_{in_t}} = \overrightarrow{\mathbf{n}_t \mathbf{n}}$ 
9:      $\mathbf{n}_{in}(x, y) \leftarrow \mathbf{n}_{in_t}(x, y)$ 
10:   end for
11: end procedure

```

Appendix I. Procedure: TestIntersection

Algorithm 7 Procedure to test if a node's translation path, \overline{nn}_t intersects a river.

Input:

$\overline{nn}_t \leftarrow$ the node's translation path

$r \leftarrow$ a river feature

Output:

Returns *True* if the node crosses a river.

Local variables:

$b_{\bar{n}}, b_{\bar{e}} \leftarrow$ the bounding boxes for \overline{nn}_t and \bar{e}

$\bar{e} \leftarrow$ an edge of r

```

1: procedure TESTINTERSECTION( $\overline{nn}_t, r$ )
2:   for each  $\bar{e} \in r$  do
3:      $b_{\bar{n}} \leftarrow \text{GetBoundingBox}(\overline{nn}_t)$ 
4:      $b_{\bar{e}} \leftarrow \text{GetBoundingBox}(\bar{e})$ 
5:     if  $b_{\bar{n}} \text{ intersect } b_{\bar{e}} = \text{True}$  then
6:       return  $\overline{nn}_t$  intersect  $\bar{e}$ 
7:     end if
8:   end for
9:   return False
10: end procedure

```
