Combined Algorithm for Voronoi Diagram Construction as it Applies to Dynamic Ride Sharing

Anton Butenko and Jorge Marx Gómez
Department of Computing Science
Carl von Ossietzky University of Oldenburg
Oldenburg, Germany
anton.butenko@uol.de, jorge.marx.gomez@uni-oldenburg.de

***Abstract*—Standard Voronoi diagram decomposes a plane into cells with a common closest site. This structure is widely used in computational geometry in application to the nearest neighbor problem. Using Euclidean metric is the most straightforward solution, however, in urban environment it may lead to insufficient accuracy that is crucial in such applications as dynamic ride sharing. Deviations in determining the nearest meeting point are especially significant under the presence of obstacles: water reservoirs, railway tracks, highways, industrial zones as well as hilly terrain. Here, we propose a combined approach for city Voronoi diagram construction in general metric space. A transportation network is modelled as weighted graph, so that the route consists of a foot-walking part and shortest path in graph. Presented algorithm constructs continuous Voronoi diagram for a plane using the individual graph Voronoi cells as generator objects. Evaluation for the specific city topography shows that the described algorithm provides more accurate results in comparasion with the standart Voronoi diagram.**

***Index Terms*—Voronoi diagram; dynamic ride sharing.**

I. Introduction

Ride sharing applications are aimed at connecting drivers and passengers in an optimal way. What this optimal way means depends a lot on the specific mobility solution philosophy and its target audience. Nevertheless, most of them face such optimization problem as the nearest neighbor search: identifying the point from a set of points which is the closest to a given point according to some measure. The mobility application Instaride [3] developed for the spontaneous shared trips is driven by an instant matching algorithm. It connects drivers and passengers in real time based on the user's mobile device positioning (satellite navigation data, triangulation in mobile network) [2]. In order to minimize the driver's efforts and his route detour, the finite set of preselected fixed points is used for passengers' pick-up and drop-off (named meeting points, in general). Preselection of the meeting points is determined by the environmental conditions and is based on criteria such as parking opportunity, presence of pedestrian zones and easily recognizable landmarks. Such an approach leads to the problem of finding the nearest meeting point for users (both drivers and passengers) based on their real-time positions. The paper structure is the following. The Introduction explains the problem's origin. In Section II, we describe the concept of the presented approach and introduce the terms and notation. Sections III and IV describe two parts of the algorithm: discrete and continuous. In Section V, the algorithm steps are given in detail. Section VI presents the algorithm efficiency evaluation for the specific city topography. Section VII concludes our work.

II. Voronoi diagram in a generalized metric space

One of the most effective ways to solve problems related to the nearest neighbor search is to use the Voronoi diagram. We introduce the following notation here: L_ρ is a metric space with the corresponding function $\rho : L \times L \to \mathbb{R}^+$ that satisfies metric axioms. Then, $O_r(x) = \{z : \rho(x,z) < r\}$ is the open metric ball with radius $r \in \mathbb{R}^+$, $S_r(x) = \overline{O}_r(x) \setminus O_r(x)$ is the metric sphere and $\Lambda(x,y) = \{z : \rho(x,z) = \rho(y,z)\}$ is the bisector of x and y. It splits L_ρ into the half-spaces $D(x,y) = \{z : \rho(x,z) < \rho(y,z)\}$ and, lying on the other bisector side $D(y,x) = \{z : \rho(y,z) = \rho(x,z)\}$. For a given finite set of seeds $S = \{s_1, ..., s_k\} \in L_\rho$, the Voronoi cell related to s_i is expressed as

$$VR(s_i, S) = \bigcap_{i \neq j} D(s_i, s_j) \tag{1}$$

and the Voronoi diagram of S:

$$V(S) = \bigcup_{i \neq j} \overline{VR}(s_i, S) \cap \overline{VR}(s_j, S). \tag{2}$$

Being the most straightforward solution, a Voronoi diagram based on the Euclidean distance provides tolerable approximation in the urban environment if the points are located quite far apart within the uniform transportation network. In other cases, the results are significantly worse: for short distances, for a sparse roads network, in areas with irregular topography, under the presence of one way roads, or in application to suburbs stretched along the roads forming axon-like structures. Natural obstacles such as rivers, lakes, vegetation zones, ravines and mountains as well as urban (railways, highway, industrial zone, pipelines) play a particularly important role in complicating the route of movement between two points. The use of other metric functions may improve the accuracy; however, another problem arises: in some cases, the bisector dimension may be more than 1 (this is true even for the Manhattan distance $\rho(x,y) = \sum |x_1 - y_1| + |x_2 - y_2|$) applicable to the regular rectangular streets network.

2022, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

In a number of works, the graph represents the streets network. The discrete network Voronoi diagram is then constructed while the metric used is the link between nodes (e.g., Yomono [6]). However, such models do not allow shortcuts, which are often used by pedestrians to shorten the routes. Aichholzer et al. [4] consider a plane with Manhattan distance and isothetic transportation network. There are also several works that use the generalized concept of Voronoi region (*needle*) proposed by Bae and Chwa [7].

The approach presented in this work is aimed at being applicable for the non-orthogonal street structure with curvilinear street segments. At the same time, as the ride sharing is spontaneous, we strive to avoid excessive model complexity; only walking to/from meeting points is assumed for the passenger. In addition, being flexible to the possibility of using the available network bandwidth data, the model should also work with the minimum information of this kind. Thus, we believe, the task of developing an optimal method for constructing a Voronoi diagram for a similar class of problems is to find a balance between complexity, accuracy and flexibility in using available data, as the latter may vary a lot in different regions.

The main idea of the approach presented below is to construct a discrete Voronoi diagram on a graph and then transform the obtained cells into the seeds or generator objects for the continuous Voronoi diagram on the plane. The latter represents the partition of the plane with a transportation network into proximity regions for the set of the given meeting points. The algorithm overview is presented below while individual steps are discussed in detail in sections III-IV.

1. Geospatial data preprocessing. The necessary information is: land use, coordinates of the roads.
2. Voronoi diagram construction on the graph representing transportation network.
3. Cell of the constructed discrete diagram with their coordinates are used as the seeds for continuous Voronoi diagram on a plane.

As a result, continuous combined Voronoi diagram for the meeting points is constructed. This algorithm had been tested for Oldenburg city centre [1] and then applied for pedestrians on the area with radius 7 km around the city centre containing 79 meeting points (Fig. 1).

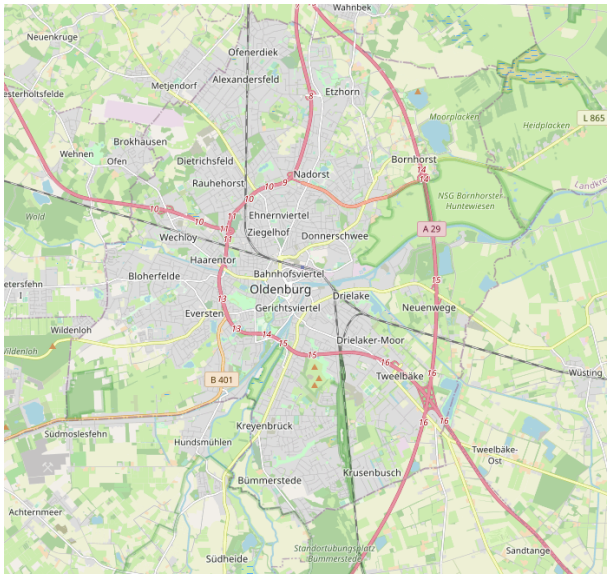

Fig. 1. Oldenburg with the suburbs (OpenStreetMap [5]).

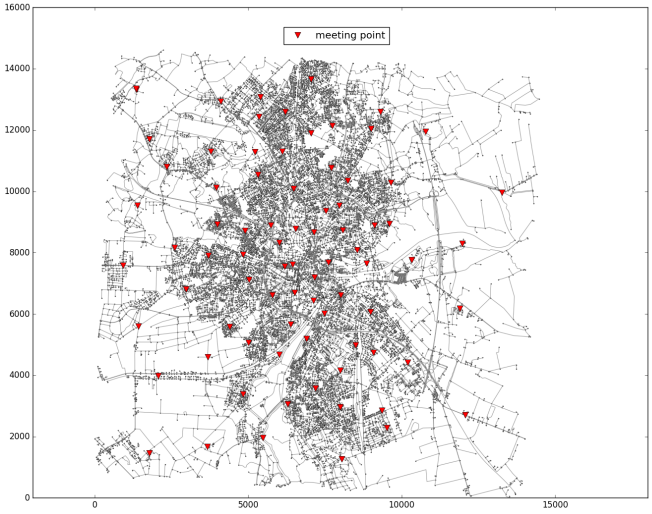

Fig. 2. Graph representing transportation network with the meeting points.

III. Voronoi diagram on the graph

We consider the area of interest as a rectangular domain $\Omega \subset \mathbb{R}^2$ containing the city transportation network, providing fixed routes. This network is modelled as a weighted graph $G(V, E)$, where $E = \{e_i\}$ is the set of edges, representing roads and streets and $V = \{v_k\}$ are the graph vertices, corresponding to the intersections and the deadlocks. Non-negative edge weights $w(e_i)$ determine some proximity measure between the vertices connected by the edge e_i.

Depending on data availability, it can be, e.g., edge length or edge travel time. The latter depends on the segment's capacity, inclination, or traffic. Setting $\rho_G(v_i, v_j)$ in an ordinary way as the weights sum of the shortest path between v_i and v_j , one can consider V_{ρ_G} as a metric space. Without additional constraints, it is true for the undirected graph as ρ_G always satisfies the symmetry axiom. It is not so in the directed graph case, nevertheless inward and outward Voronoi diagrams with corresponding asymmetric metric function can be considered instead. In application to the present nearest neighbor search inward diagram is a general approach that represents both types of movement: driving by car and walking. Taking into account that there are no one-way pedestrian roads and omitting the height difference for simplicity, one can assume that the undirected graph provides good representation for walking on foot in the mild regions.

Hence, we can build a Voronoi diagram on the graph $G(V, E)$ with respect to the meeting points $S = \{s_1, .., s_k\} \subset V$ (Fig. 2).

2022, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

The Voronoi diagram brakes up the set of vertices into the direct sum of the Voronoi cells $V = V_1 \oplus ... \oplus V_k$, where $V_i = VR(s_i, S)$. Let $E_i(s_i)$ be a set of edges connecting vertices within V_i. Then $E = E_1 \oplus ... \oplus E_k + E_0$, where E_0 is the set of "border" edges whose vertices belong to the different cells.

The following steps describe a computational algorithm for constructing a Voronoi diagram on a graph. The city transportation network representation as a graph $G(V, E)$ is obtained from the OpenStreetMap (OSM) project geodata [5]. The project provides free editable geographic database of the world. In this work we use Python package Osmnx to download, model and project geospatial OSM data. The rest of the code is also written in Python using such packages as NumPy, Shapely, Matplotlib, Networkx, GeoPandas and others.

At the first step geospatial data of this region is downloaded and projected to Gauss-Krüger projection in which all further computations take place. Thus, current data structure appears as a weighted graph with the certain geometrical coordinates for nodes and edges. Second, locations of the meeting points are added to the set of graph vertices (Fig. 2). Since graph order for the individual town lets allows it, brute-force can be used for the Voronoi diagram construction: $\forall v \in V$ find the distance on the graph $\rho_G(v, s_i), i = \overline{1, k}$ using the Dijkstra algorithm. If s_j satisfies $\rho_G(v, s_j) = \min\limits_i \rho_G(v, s_i)$, then $v \in VR(s_j)$. This computaion can be easily and effectively parallelized as long as there is no need for data transfer between the threads. Set V is split up into disjoint subsets by the processor cores number. Then nodes of each subset are divided into the groups according to their proximity to a certain seed. Finally, the results are combined together. Finding terms V_k for direct sum decomposition of V allows to determine corresponding graph edges subsets E_k belonging to which clearly indicates the nearest seed – meeting point for each $e \in E \setminus E_0$.

IV. Planar Voronoi diagram

Constructed according to the previous section, the Voronoi diagram on the graph does not indicate the nearest meeting point for the surface points lying outside the graph edges. As graph $V(G, E)$ can be considered not only as a topological structure but set of geometrical objects: points and lines with the certain coordinates, each subset E_m corresponds to the lines set E'_m on $\mathbb{R}^2$. It should be noted that in general two seeds may intersect (Fig. 4). Normally it happens under presence of the multi-level roads interchanges, tunnels, crossroads with the prohibition for movement in the certain direction but, in general, may have a connection with the roads congestion and bandwidth. Although such cases represent a small proportion of the total number of cases, the need to process them significantly complicates the procedure for bisector construction (section IV-C).

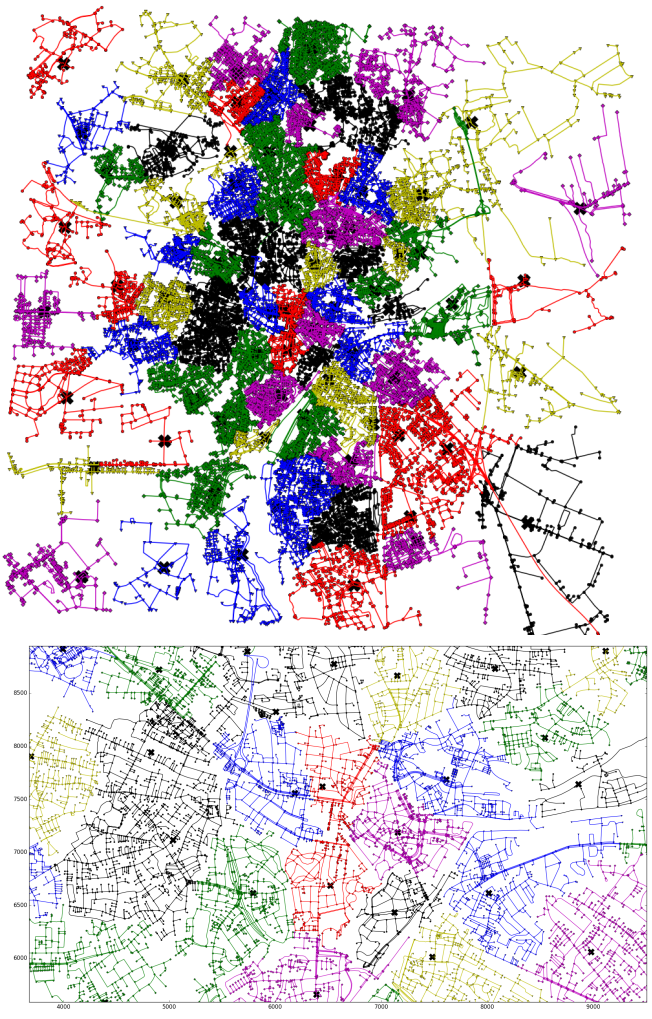

Fig. 3. Voronoi diagram on the graph.

A. Planar metric function

As long as there is no exact information about travel routes outside the transportation lines, it is natural to assume movement along a straight line in the direction of the nearest transportation network segment. However, this simplification does not take into consideration the presence of natural and man-made obstacles: buildings, fences, water reservoirs, ravines, vegetation and industrial zones, farmland as well as private and restricted access areas. In suburbs and rural surroundings such objects can occupy a large area, therefore bypassing them significantly complicates the route. On the one hand, it is possible here to consider a geometrical problem of building an optimal curvilinear route between a given point and a transportation network that does not intersect impassable regions. The length of such a route is then used as a metric function.

On the other hand, the problem of identifying impassable regions from generally available geospatial data can be more difficult than it seems. Although some obstacles can confidently be considered as impassible, for others it is hard to

2022, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

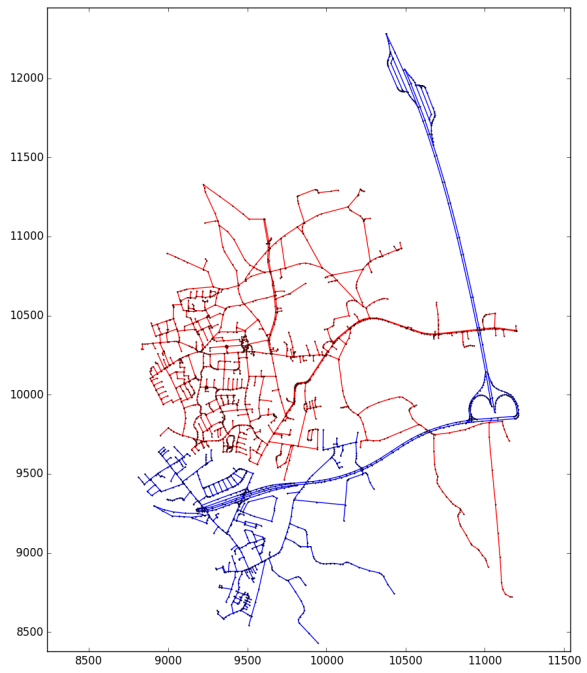

Fig. 4. Geometrically overlapping cells of Voronoi diagram on the graph.

determine their real degree of obstruction. This applies to a lesser extent to movement by car, but is relevant enough for pedestrians who tend to take shortcuts. For example, taking a shorter route by moving through a vegetation zone may depend on vegetation type, density, soil type, time of year, weather, time of day (due to the illumination factor). Thereby, not only spatial, but short- and long- term time variation of site passability occurs. Even the water reservoirs can freeze in winter and become passable. Additional socio-behavioral aspects play a role in relation to the zones forming artificial obstacles. For them obstruction may depend on such factors: if they are actively used or abandoned; if there security guards and/or CCTV; the kind of fence around the perimeter; legal consequences of a violation. The same applies to the crossing the railways and highways outside the permitted spots.

It should be noted that there is likely a connection between shortcut usage and benefits of route reduction. In contrast, a high local crime rate can drastically reduce pedestrians' willingness to walk outside of the streets. Moreover, tendency to follow formal prohibitions varies in different cultures and regions [9]: while in some cases a prohibition sign is enough, in others even concrete fence is useless. It seems that up-to-date information regarding the passibility of shortened or alternative routes should come via some pedestrians' feedback system. Satellite imagery can help with the determining of vegetation properties and recognition of footpaths. Nevertheless, leaving this approach for the future stage of work, we currently use the Euclidean distance as a metric function.

B. Search for the equidistant points

Considering $\{E'_1, ..., E'_k\}$ as seeds in (2) and Euclidean metric ρ_2 as ρ, Voronoi diagram $V(E')$ can be constructed. Obviously, $\rho_2(M, E'_m)$ is the distance between $M \in \mathbb{R}^2$ and the nearest to M point of E'_m.

The first step is to find the metric sphere $S_r(E'_m)$ for E'_m with the given radius r. The metric sphere analytically obtained for the straight line segment consists of two couples of straight line segments and circular arcs. As far as even curvilinear roads are represented in E'_m as polygonal chains, $S_r(E'_m)$ is expressed as the individual spheres union's perimeter. By the definition, for two seeds and any point $M \in \Omega : M \in S_r(E'_m) \cap S_r(E'_n) \Rightarrow M \in \Lambda(E'_m, E'_n)$. Therefore, finding sufficient number of such equidistant points as equal radius spheres intersection, allows to determine with some precision the bisector within the domain through further interpolation. Let B'_r denote the set $S_r(E'_m) \cap S_r(E'_n)$. Giving to the radius r variation with some step: $r_{k+1} = r_k + \Delta r$ $(k = 0, 1, ...)$ we compute all coresponding metric spheres intersections B'_r. Here $r_k \in [r_{min}, r_{max}]$, where $r_{min} = \frac{1}{2}\rho_2(E'_m, E'_n)$ and r_{max} is the minimum radius r_k that satisfies the condition $B'_{r_k} \not\subset \Omega$. Choice for the Δr depends on two aspects. First, the set of obtained equidistant points should be adequate for the proper bisector line representation. Second, the excessive precision should be avoided to reduce computational complexity at this algorithm stage. For this reason, the variable radius increment step is chosen: $\Delta r(k) = \Delta r_k$:

$$\Delta r_k = \begin{cases} f \cdot \Delta r_{k-1} & \text{if } E'_m \cup E'_n \subset \bigcup\limits_{l=m,n} O_r(E'_l), \\ \Delta r_0 & \text{otherwise} \end{cases} \quad (3)$$

The radius increment step remains constant unless both seeds are located within the open balls union, hereafter it grows geometrically. Fig. 6 (top) illustrates how it affects computed points distribution. In the computations below $\Delta r_0 = 6$ meters and $f = 1.25$. As a result, for each pair E'_m, E'_n we obtain a set of equidistant points as combination:

$$B'(E'_m, E'_n) = \bigcup_{\forall r_k} B'_{r_k}(E'_m, E'_n) \quad (4)$$

C. Bisector construction

A goal of the current step is to construct a continuous bisector from the set of equidistant points $B'_{mn} = B'(E'_m, E'_n) = \{Q_i\}_{i=0}^{N_B}$, $Q_i \in \mathbb{R}^2$. Bisector $\Lambda = \Lambda_{mn}$ constructed from $B' = B'_{mn}$ should satisfy the following conditions:

1. Λ is a finite set of simple curves without self-intersections.
2. Λ contains maximum number of points from B'.
3. Each curve in Λ intersects Ω in two points making the domain partition possible.
4. Λ does not intersect with E'_m and E'_n.

The problem of such line set construction is to separate points into groups (if necessary) and arrange them in each group in a correct order for interpolation.

2022, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

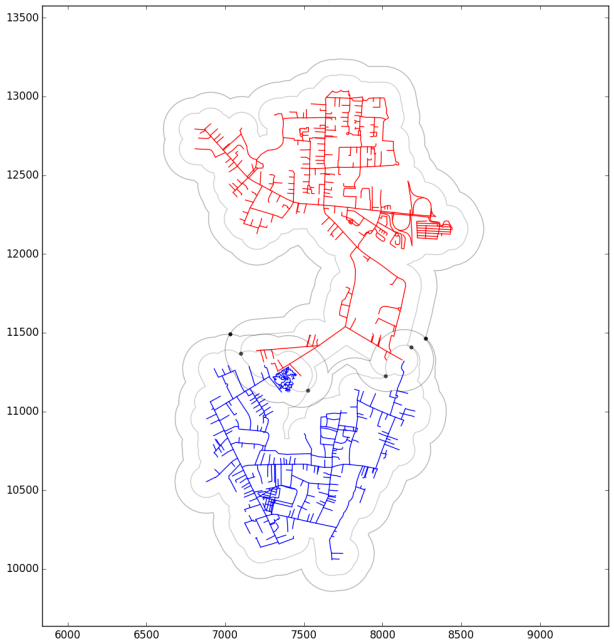

Fig. 5. Equidistant points as equal radius metric spheres intersection.

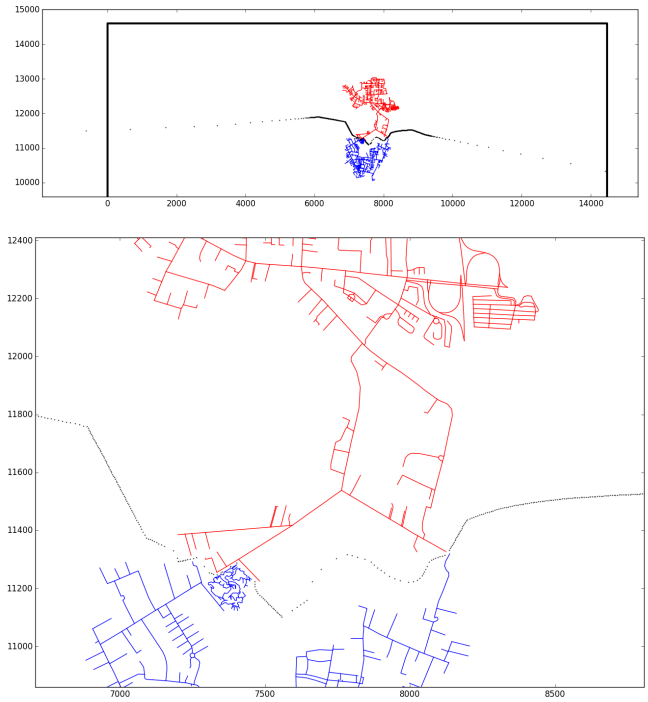

Fig. 6. Computed equidistant points.

Assuming $E'_m \bigcap E'_n = \emptyset$, $\Lambda = \Lambda_{mn}$ will consist of one line L that can be obtained with the following procedure. Let denote L_ω as a tuple of points.

1. Select arbitrary the initial point $Q_0 \in B' : Q_0 \in \Omega$.
   Assign $L_\omega = (Q_0)$; $B' = B' \setminus \{Q_0\}$.
2. Find $Q_1 \in B' : Q_1 \in \Omega$ that is the nearest to Q_0 point in B'.
   Set $L_\omega = (Q_0, Q_1)$, $B' = B' \setminus \{Q_1\}$, $n = 2$.
3. For $L_w = (Q_{j_0}, ..., Q_{j_{n-1}})$ find $Q^\alpha_n, Q^\beta_n \in B'$ such that:
   a) $\rho_2(Q^\alpha_n, Q_{j_0}) = \min\limits_{Q \in B'} \rho_2(Q, Q_{j_0})$.
   If $\rho_2(Q^\alpha_n, Q_{j_0}) < \rho_2(Q^\alpha_n, Q_{j_{n-1}})$ then place Q^α_n as the first element in L_ω and set $B' = B' \setminus \{Q^\alpha_n\}$.
   b) $\rho_2(Q^\beta_n, Q_{j_{n-1}}) = \min\limits_{Q \in B'} \rho_2(Q, Q_{j_{n-1}})$.
   If $\rho_2(Q^\beta_n, Q_{j_{n-1}}) < \rho_2(Q^\beta_n, Q_{j_0})$ then place Q^β_n as the last element in L_ω and set $B' = B' \setminus \{Q^\beta_n\}$.
4. Assign $n = n + 1$; repeat step [3.] until B' is not empty
5. Compute L as the linear interpolation of L_ω.

In other words, the process of points arrangement is the sequential increment of the polygonal chain from the two ends. Testing shows that this approach, which is based on simple proximity, provides sufficient accuracy in the vast majority of cases. However, for some closely located seeds with irregular outlines containing combinations of convex and concave elongated segments it may lead to: skipping some of B' points. Also obtained with interpolation line L can: a) contain loops; b) intersect with the seeds. Thus, this resulting L requires examination and, if necessary, must be rebuilt. In exeptional cases in order to enhance the algorithm robustness, a simplified bisector can be constructed. A possible option in such a case is an analytically obtained straight line – bisector of the seeds centroids.

D. Overlapping seeds processing

In this section case $E'_m \bigcap E'_n \neq \emptyset$ will be covered (Fig. 7, 8). The above described procedure for the bisector construction does not work correctly under this condition. In the test performed for Oldenburg this was observed 9 times among 3081 pairs. The approach is the following: the procedure [1] – [5] from the previous section is performed recursively with the additional constraints for $\Lambda = \{L_j\}_{j=1}^p$:

$$L_j \bigcap_{j \neq i} L_i = \emptyset \text{ and } L_j \bigcap E'_l = \emptyset. \quad (5)$$

Here choice $l = m$ or $l = n$ is voluntary. The process of bisector construction for the intersected seeds is presented below. Numbers in square brackets refer to the algorithm steps for the individual line from section IV-C.

I. Assign a value to l; Set $j = 0$.
II. Set $B' = B'_j$.
   1. Perform step [1.].
   2. Find Q_1 as in [2.]. Set $B' = B' \setminus \{Q_1\}$.
      If Q_0Q_1 intersects E'_l then repeat the current step. Otherwise set $L_\omega = (Q_0, Q_1)$.
   3. Let $\xi \in \{\alpha, \beta\}$. If $Q_{j_0}Q^\xi_n$ does not intersect E'_l then perform for Q^ξ_n steps [3.a] or [3.b].
   4. Perform step [4.].
   5. Perform step [5.].
III. Add L into Λ.
   Set $B'_{j+1} = B'_j \setminus L_\omega$, $j = j + 1$.

2022, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

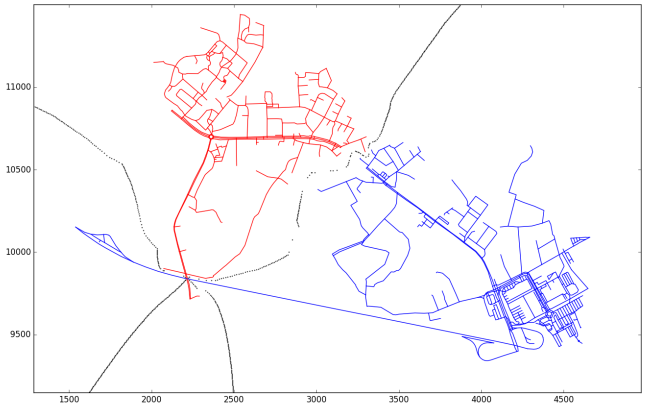

Fig. 7. Equidistant points for the overlapping seeds: single intersection.

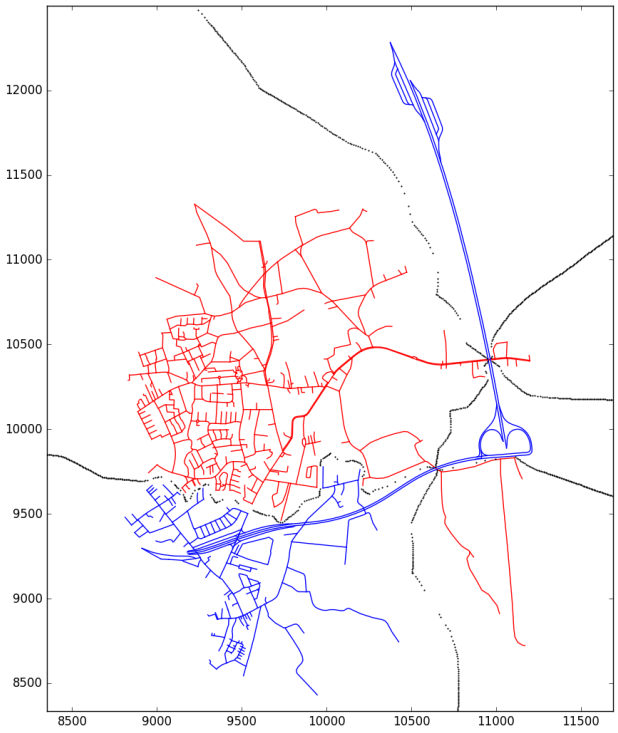

Fig. 8. Equidistant points for the overlapping seeds: multiple intersections.

IV. Repeat steps II, III until $card(B'_j \bigcap \Omega) < K^*$.

Using $K^* = 0$ is possible although it reduces algorithm robustness. In certain cases, a few equidistant points may be left unused due to the algorithm simplifications. As a result of steps I-III, we obtain lines set Λ.

It is worth to mention that computations on the step of bisesector construction as well as on the step of searching for the equidistant points allows effective parallelization: one thread is allocated for each seeds pair.

E. Voronoi cells construction

For the certain seed E'_m each computed bisector Λ_{mn} splits the domain in two parts: $\Omega = \Omega_n^+ \oplus \Omega_n^-$: $E'_m \subset \Omega_n^+$, $E'_m \not\subset \Omega_n^-$. If the bisector consists of a single line then it cuts the domain in two polygons. Otherwise multiple bisectors divide the domain into one simply connected region and one multiply connected region consisted of two or more subregions (Fig. 9). According to the Voronoi cell definition:

$$VR(E'_m, E') = \bigcap_{i=\overline{1,k}} \Omega_i^+. \quad (6)$$

Making a reverse substitution $E'_m \rightarrow E_m \rightarrow s_m$ we get $VR(s_m, S)$ as the cells of the combined continuous Voronoi diagram based on metric functions ρ_2 and ρ_G. In practice, due to the limited accuracy for bisector computation $\Omega = \bigcup VR(s_m, S) \bigcup R$. While cells overlay is not possible, the voids $R = \{R_j\}$ between cells may occur. Simple procedure is used here to dispose of this areas of uncertainty: each void is merged with the cell that has the largest common border with it.

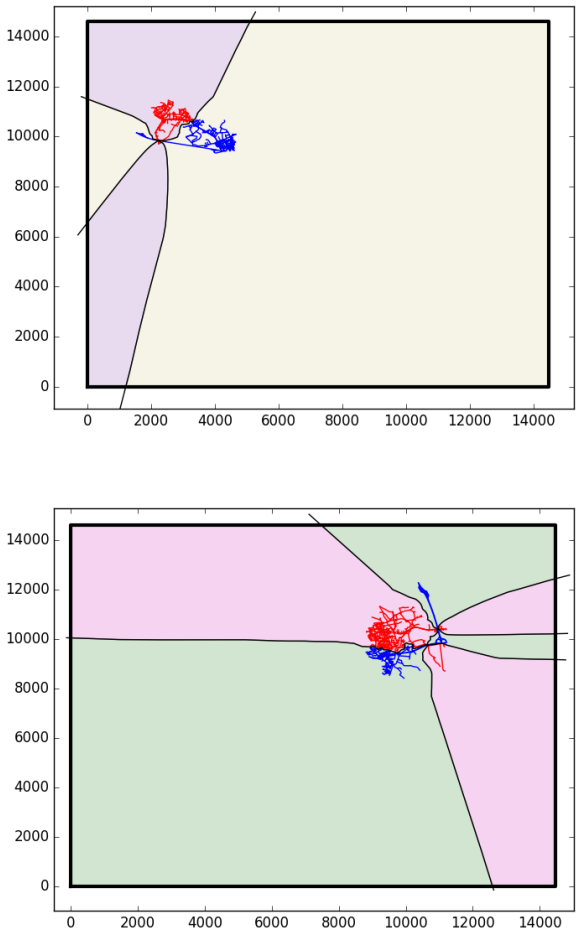

Fig. 9. Domain partition. Overlapping seeds case.

2022, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

V. Evaluation

As mentioned above, test computations were performed for the area with radius 7 km around Oldenburg city centre. The selected region consists of urban, suburban and rural areas. It also includes a variety of natural and man-made obstacles: highways, railways, water reservoirs, industrial and vegetation zones. For a given region with 79 selected meeting points, we construct analytically the standard Voronoi diagram based on the Euclidean distance (diagram I) and the combined one in a way described above (diagram II). After the cells construction through half-spaces intersection, 151 void region remained within the domain. Their total area ≈ 0.1 km^2 that is 0.05% of domain area. The area of the two largest voids is approx. 99% of total void area while 131 are smaller than 1 m^2. After merging the voids with the computed cells the final diagram is obtained (Fig. 11).

For comparison of Voronoi diagrams of types I and II, the following value is used as measure of difference:

$$\Delta S = \frac{1}{S(\Omega)} \sum_i S(C_i^1 \setminus C_i^2) = \frac{1}{S(\Omega)} \sum_i S(C_i^2 \setminus C_i^1), \quad (7)$$

where C_i^1 and C_i^2 are the cells for the same seed in diagrams I and II correspondingly. For the considered example $\Delta S \approx 18\%$. One can expect the bigger difference for higher number of meeting points and, consequently, smaller cells. Also, for 3501 random locations uniformly distributed within the domain, we determine the nearest meeting point in three ways: a) from diagram I; b) from diagram II; c) by computing the routes to all the meeting points with the Openrouteservice engine [10] and detecting the meeting point corresponding to the minimum route length. The results are the following. The nearest meeting points obtained from diagrams I and II are not equal in 18% what is consistent with ΔS value. Meeting points are different from the obtained with Openrouteservice for 17% (diagram I) and 10% (diagram II).

VI. Conclusion

There are several steps to be performed next in the context of this work. First, the potential of using additional bandwidth data must be analyzed. Second, impassable region processing must be implemented in planar Voronoi diagram construction. Third, the presented approach must be tested for the different regions and other methods of travelling, e.g., cars and bicycles. Nevertheless, at this stage, one can conclude that, despite the number of simplifications, the described algorithm provides more accurate results in comparison with a standard Voronoi diagram. At the same time, the processing of complex topography features requires further study since they are probably the main reason for the remaining imprecision. These include multi-level road crossings, tunnels, elongated geometric objects and natural obstacles.

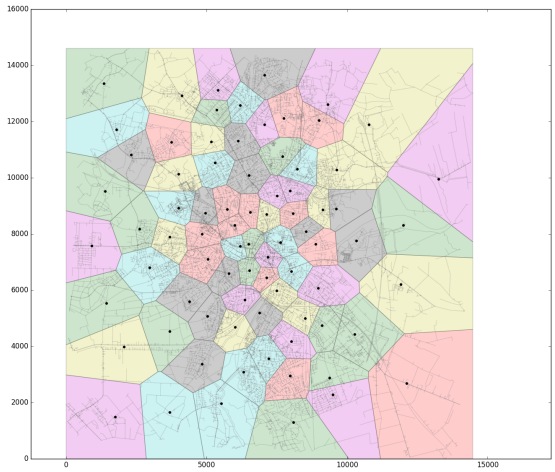

Fig. 10. Classic Voronoi diagram.

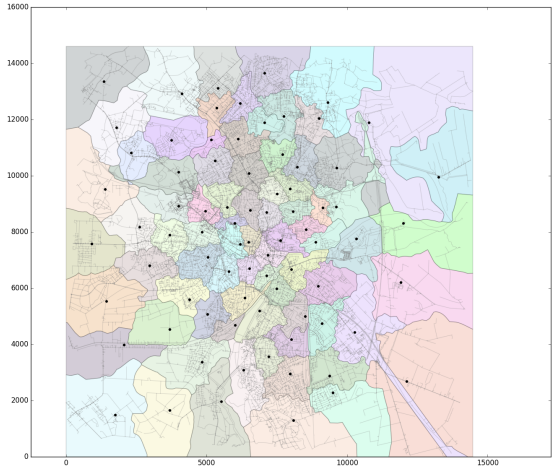

Fig. 11. Combined Voronoi diagram.

References

[1] A. Butenko and J. Marx Gómez, “Combined algorithm for Voronoi diagram construction in application to dynamic ride sharing,“ MOBILITY 2022: The Twelfth International Conference on Mobile Services, Resources, and Users, pp. 5-8, Porto, Portugal, 2022.

[2] M. Eilers et al., “An instant matching algorithm in the context of ride-hailing applications, using isochrones and social scoring”, In:, (Hrsg.), Informatic 2021.

[3] Instaride, https://instaride.webflow.io, retrieved: November 2022.

[4] O. Aichholzer, F. Aurenhammer, and B. Palop, “Quickest Paths, Straight Skeletons, and the City Voronoi Diagram”, Discrete and Computational Geometry, 31. pp. 17-35, 2004,doi:10.1007/s00454-003-2947-0. Gesellschaft für Informatik, Bonn, pp. 103-114, 2021, doi:10.18420/informatik2021-007.

[5] https://www.openstreetmap.org, retrieved: November 2022.

[6] Yomono H. Yomono, “The Voronoi diagram on a network”, Technical report, Nippon Systems Co, Tokyo, 1991.

2022, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[7] S. W. Bae and K.-Y. Chwa, "Voronoi Diagrams with a Transportation Network on the Euclidean Plane," Int. J. Comput. Geometry and Appl., vol. 16, pp. 101-112, 2004, doi:10.1007/978-3-540-30551-4_11.
[8] https://github.com/gboeing/osmnx, retrieved: November 2022.
[9] G. Bierbrauer, "Reactions to violation of normative standards: a cross-cultural analysis of shame and guilt", International Journal of Psychology, vol. 27, pp. 181-193, 1992, doi: 10.1080/00207599208246874.
[10] https://openrouteservice.org, retrieved: November 2022.

2022, © Copyright by authors, Published under agreement with IARIA - www.iaria.org