Annotating OpenStreetMap data with elevation data

Bachelor's Thesis

Albert-Ludwigs-Universität Freiburg

Urs Spiegelhalter
Department of Computer Science
Chair of Algorithms and Data Structures
March 16, 2022

Problem: Situation

- OpenStreetMap (OSM) contains map data of the whole planet
- Map features are stored in the objects
  - Nodes: Single locations as (Longitude, Latitude)
  - Ways: List of nodes
  - Relations: List of nodes, ways, other relations

- All objects can have tags as *key=value* strings where any additional information can be stored
- The tag *ele=** is used to denote the elevation above sea level in meters of an object.

Problem: Situation

Type	Number of objects with *ele*=*
All	0.09%
Node	1.20%
Way	0.63%
Relation	1.45%

Table 1: Tag *ele*=* usage in OSM objects. Data provided by taginfo [1].

⇒ Elevation data in OSM very lacking

Problem: OpenStreetMap data

Number of nodes	Number of ways	Number of relations
7,468,758,706	832,784,309	9,611,640

Table 2: Total number of objects in OSM. © OpenStreetMap contributors

⇒ Elevation data for billions of objects have to be added

Problem: Elevation data

- We use elevation data from the NASA Digital Elevation Model (NASADEM)
- NASADEM has near-global coverage and a horizontal resolution of 1 arc-second, which is approximately 30 meters

Number of files	Total size compressed	Total size uncompressed
14,520	109 GB	376 GB

Table 3: NASADEM elevation dataset size.

⇒ Combination of the OSM and elevation data sizes result in large amounts of data

Problem: Elevation data errors

- Errors can be obversed when looking at the elevation data of map features in OSM
- Obvious errors can occur due to the horizontal resolution of NASADEM of approx. 30 meters not being accurate enough
  - Roads suddenly dropping or rising by multiple meters
  - Rivers flowing uphill
- Elevation data that is not captured by NASADEM
  - Tunnels
  - Bridges

Problem: Elevation data errors

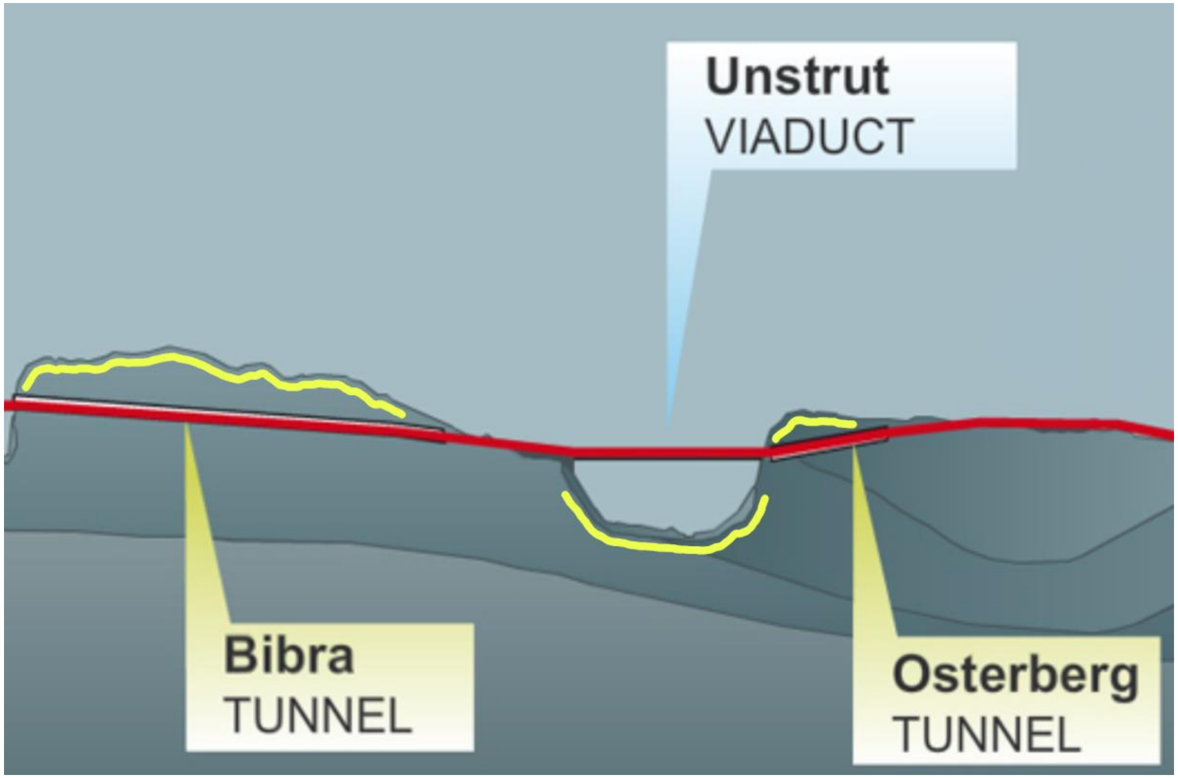

Figure 1: Geologic cross-section of a railroad [2]. The red line represents the railroad (elevation). Yellow lines indicate the incorrect NASADEM elevation data when applied to the railroad.

Problem definitions

1. Annotate OSM data with elevation data by adding a tag *ele*=* to each node.
2. Perform corrections on elevation data in OSM data. The corrections are based on linear map features like roads, rivers. By corrections affected nodes get their *ele*=* tag updated.

Adding elevation data: NASADEM

- The elevation data from the NASADEM dataset are provided by 14,520 tiles of size 1 degree longitude by 1 degree latitude, each tile is stored in a compressed file

Figure 2: NASADEM elevation data tiles over southern Europe.

Adding elevation data: NASADEM

- Each tile has a horizontal resolution of 1 arc-second, which is approximately 30 meters
- This corresponds to 3601 x 3601 cells in each tile

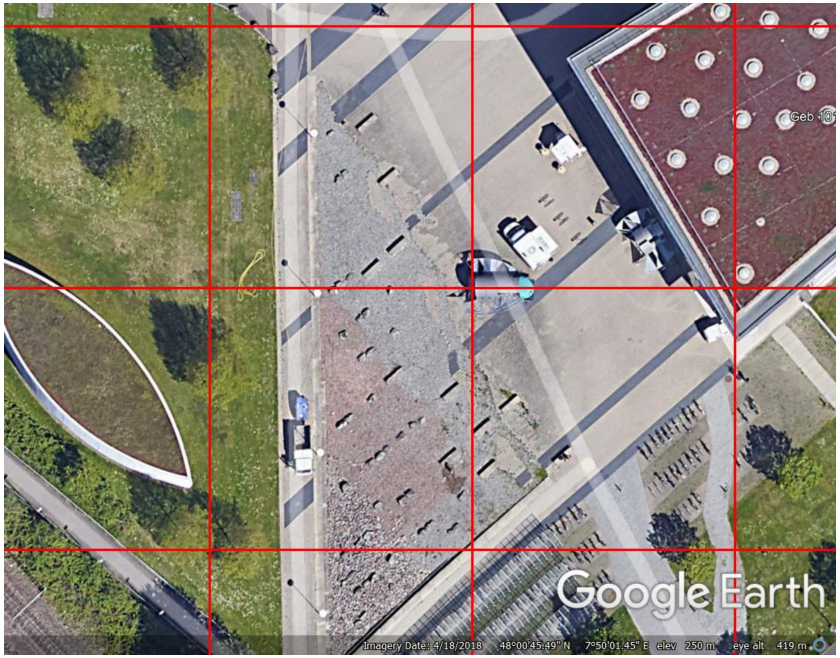

Figure 3: Individual cells of an elevation data tile. Maps data: Google, ©2020 Image Landsat / Copernicus

Adding elevation data: Geographic partitions

- We add a tag *ele*=* to each node in OSM
- To get the elevation for a single node, the complete elevation data tile, where the node is located in, has to be loaded into main-memory
- A single uncompressed elevation data tile has a size of ≈ 26 MB
- The nodes in the input OSM data do not have any geographical clustering

$\Rightarrow$ To retrieve the elevation of each node from NASADEM in the original order is not feasible

Adding elevation data: Geographic partitions

- We perform multiple passes over all nodes
- In a single pass, only nodes that are located in a geographic partition get processed
- A geographic partition is a rectangular on the map defined by its bottom-left and upper-right coordinates
- This introduces a geographic clustering of the nodes over all passes
- We can directly control the main-memory consumtion of elevation data tiles by changing the size of the geographic partitions

Adding elevation data: Geographic partitions

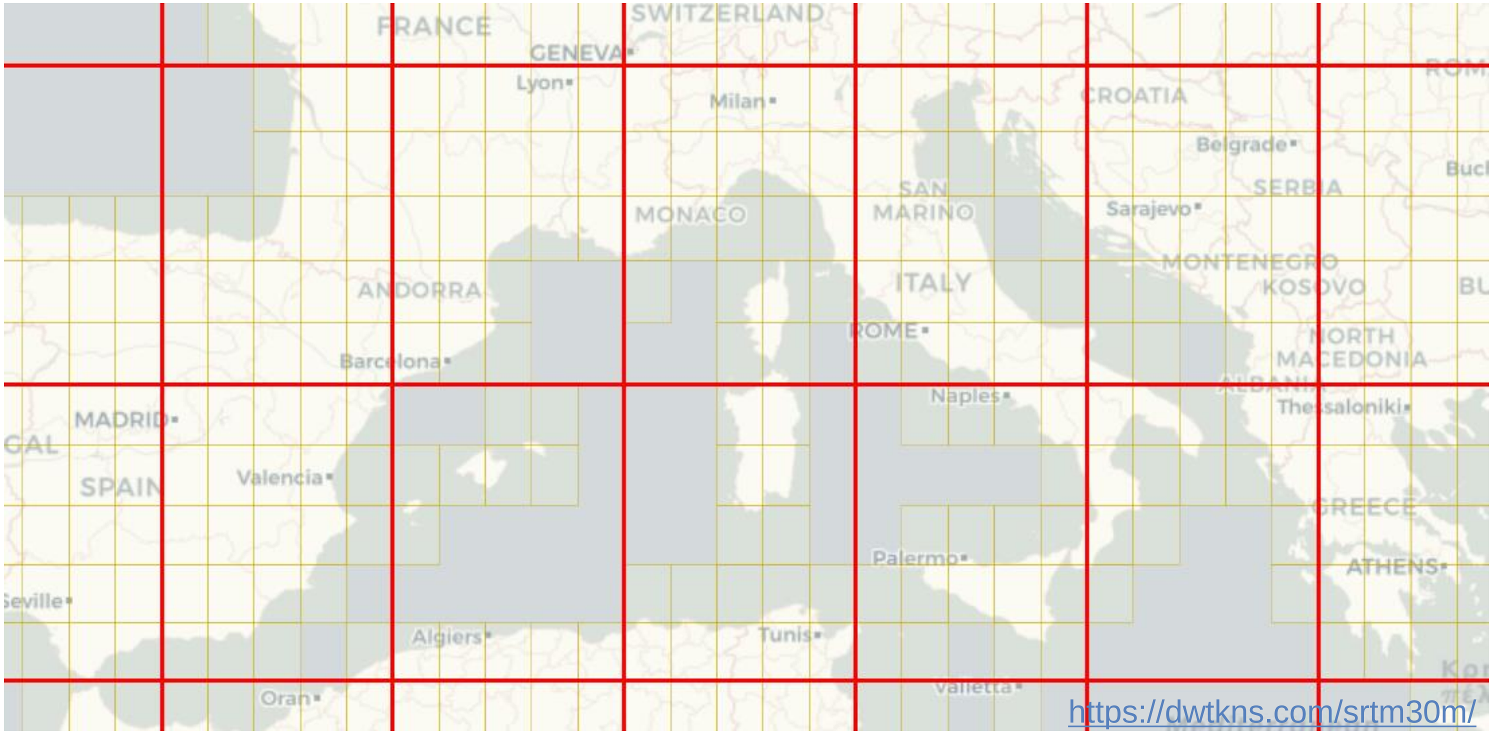

Figure 4: Geographic partitions of size 5. Only elevation data tiles inside a geographic partition get loaded into main-memory at once.

Adding elevation data: Retrieving elevation data

- The elevation of a node is stored in the cell where the node's location lies in
- We additionally use interpolation with the surrounding cells
- We use inverse distance weighting as the interpolation algorithm
- The importance of a data point decreases as distance increases
- The center of the cells represent the known data points

Adding elevation data: Retrieving elevation data

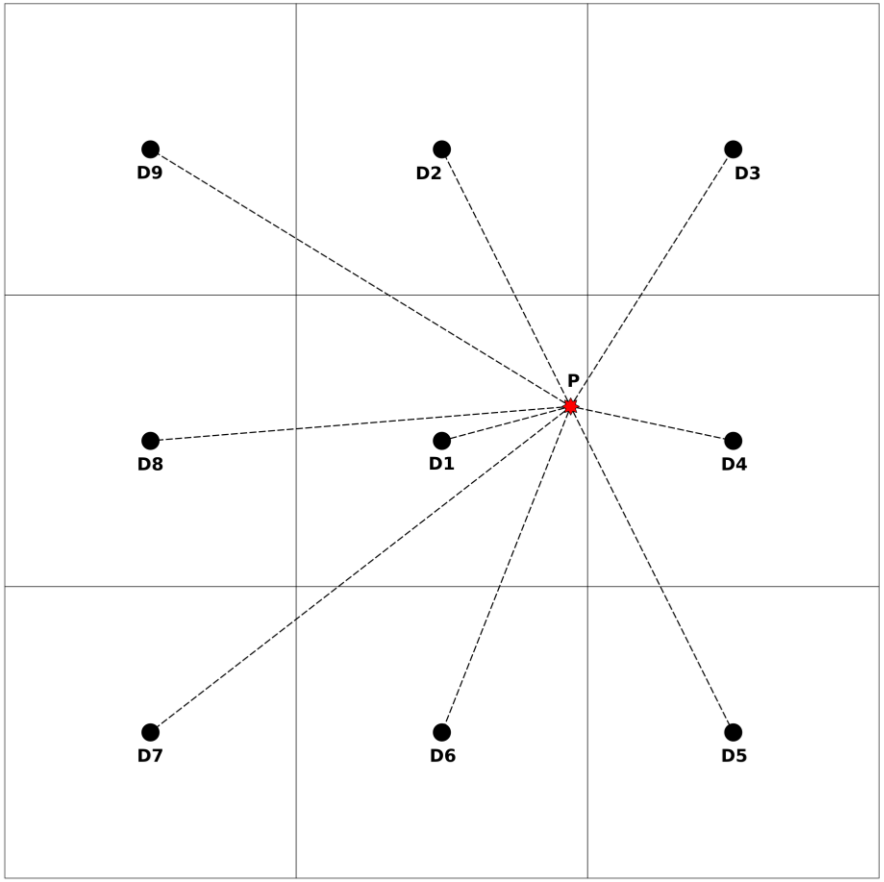

Figure 5: Inverse distance weighting for a point P with surrounding cells. D1 – D9 represent the known data points (centers of the cells).

Correcting elevation data: Routes

- Routes are linear map features in OSM that can be traveled by car, train, bike, foot, or ship
- Short-term fluctuations in elevation along routes should not occur
- Rivers should not flow uphill
- Routes going through tunnels/bridges should not have the elevation of the mountain/valley they cross

Correcting elevation data: Routes data

- Routes are stored in OSM ways and relations, specific tags are used like *type=route*, *waterway=river*, *highway=footway*
- At the lowest level, a route is a linestring of nodes
- The data needed for a route in a way:
  - The data of the nodes of the way
- The data needed for a route in a relation:
  - The data of the ways of the relation
  - The data of the nodes of the ways

⇒ Not feasible to have all data in main-memory at once

Correcting elevation data: Routes ranges

- Routes range: Specify how many routes and their data are loaded into main-memory at once and subsequently corrected

Relation 1	*type=route*
Relation 2	
Relation 3	*type=route*
Relation 4	*type=route*
Relation 5	
Relation 6	*type=route*
Relation 7	
Relation 8	*type=route*

- Routes ranges of size 2:
  - First iteration: Relation 1 & 3
  - Second iteration: Relation 4 & 6
  - Third iteration: Relation 8

Correcting elevation data: Smoothing routes

- To remove short-term fluctuations in elevation along routes, we apply a smoothing algorithm [3] to the linestrings of nodes

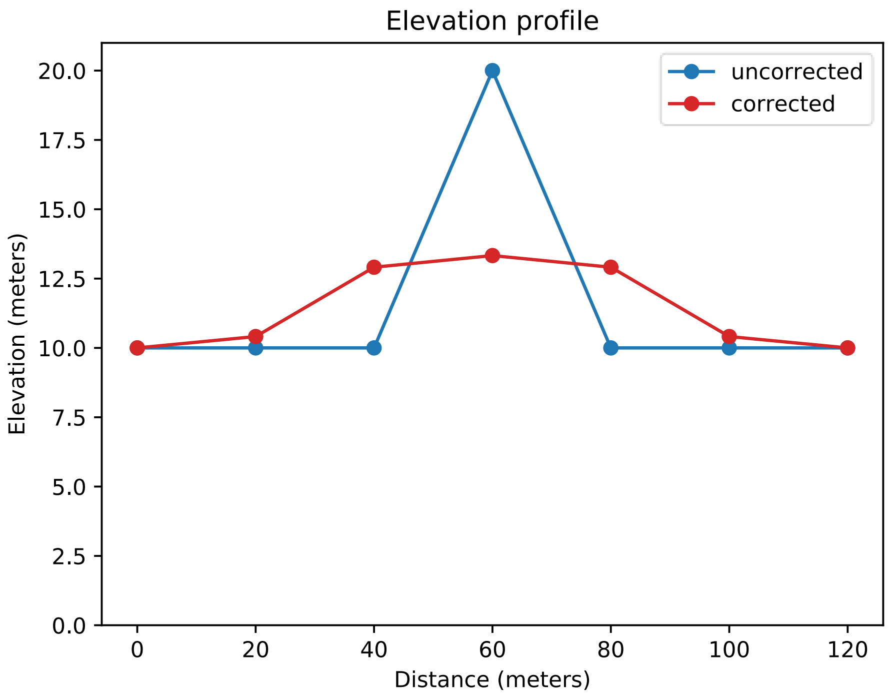

Figure 6: Smoothening of a short-term fluctuation in the elevation profile of a linestring of nodes.

Correcting elevation data: Smoothing routes

- The elevation of each node along the linestrings gets averaged over the integral of the 30 meters before and after the node

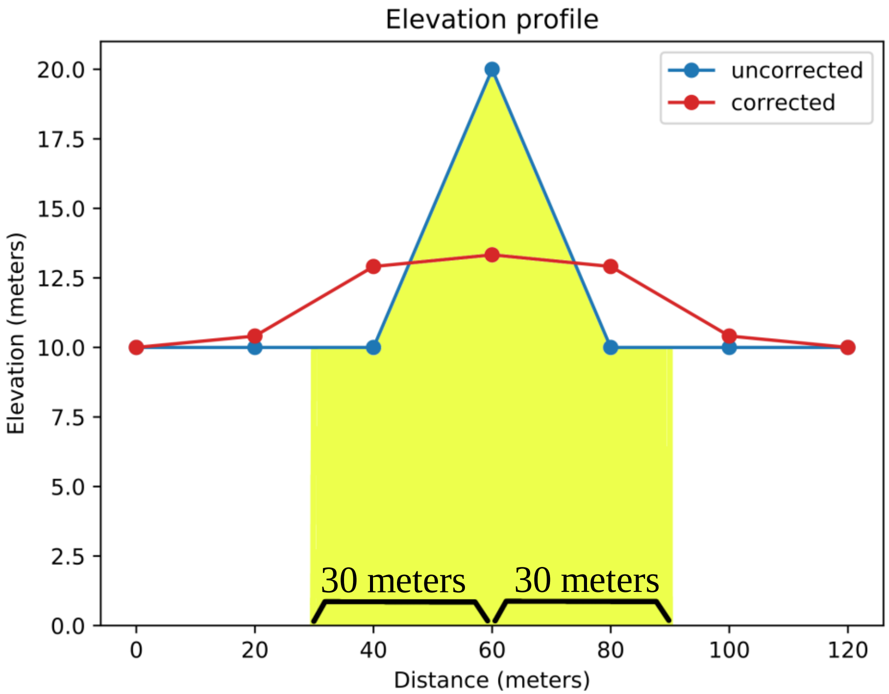

Figure 7: The area (yellow) for the average of the blue node at (60, 20). The average (red node at (60, 13.33)) results from dividing the yellow area by 60.

Correcting elevation data: Tunnels/bridges

- We span a plane between the start and the end of the tunnel/bridge
- To get the correct elevation of a node along a tunnel/bridge, we sample the z-coordinate from the plane using the x- and y-coordinates (longitude and latitude) of the node
- The plane allows for non-straight tunnels/bridges

Correcting elevation data: Tunnels/bridges

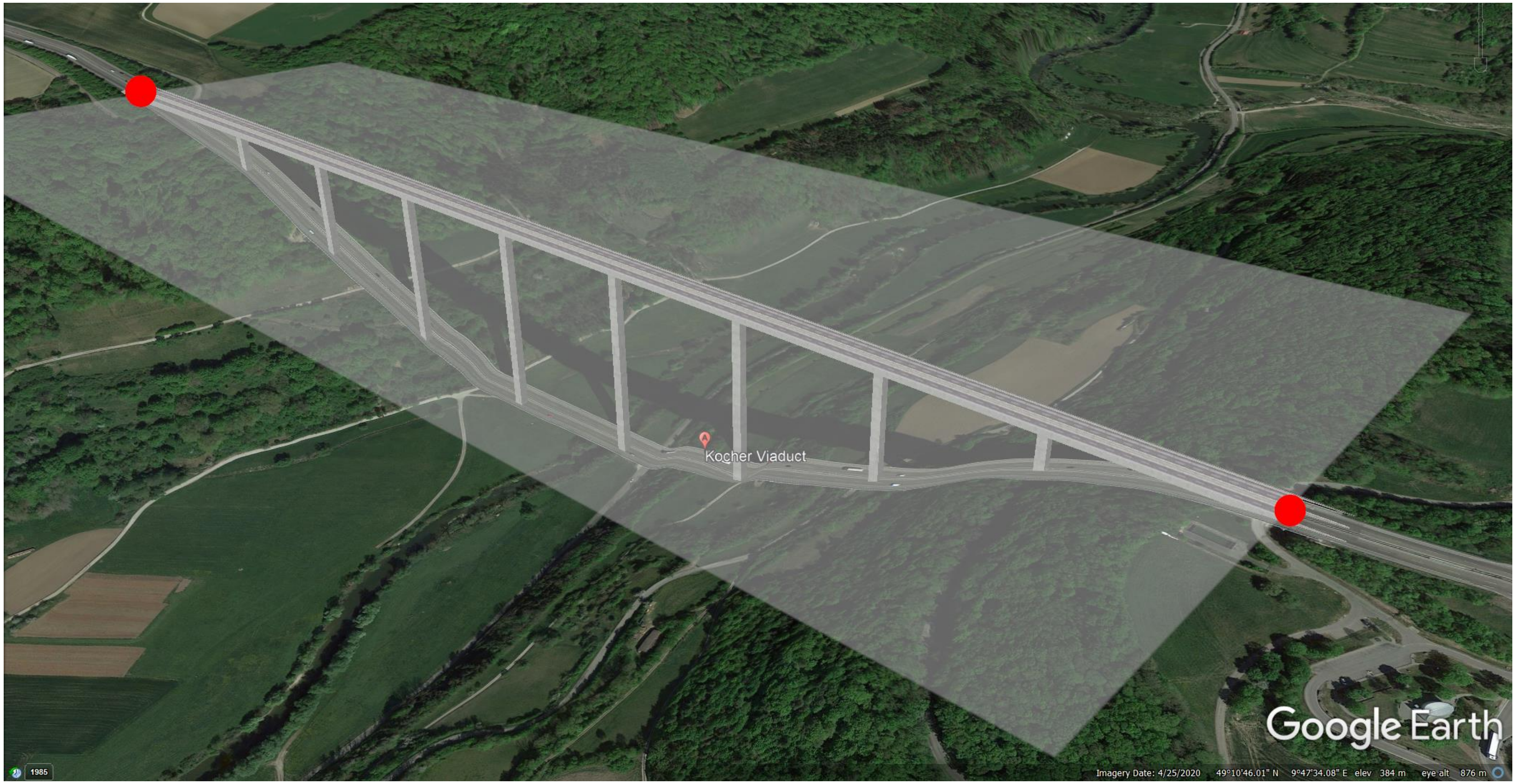

Figure 8: The plane spanning between the start and end (red dots) of a bridge. Maps data: Google, ©2020 Image Landsat / Copernicus

Correcting elevation data: Rivers

- To make sure that rivers do not flow uphill, we simply iterate over the linestrings of nodes of the rivers
- The linestrings of rivers are ordered in the direction of the flow
- If a subsequent node has a higher elevation, we set that elevation to the previous node's elevation

Evaluation: Smoothing route

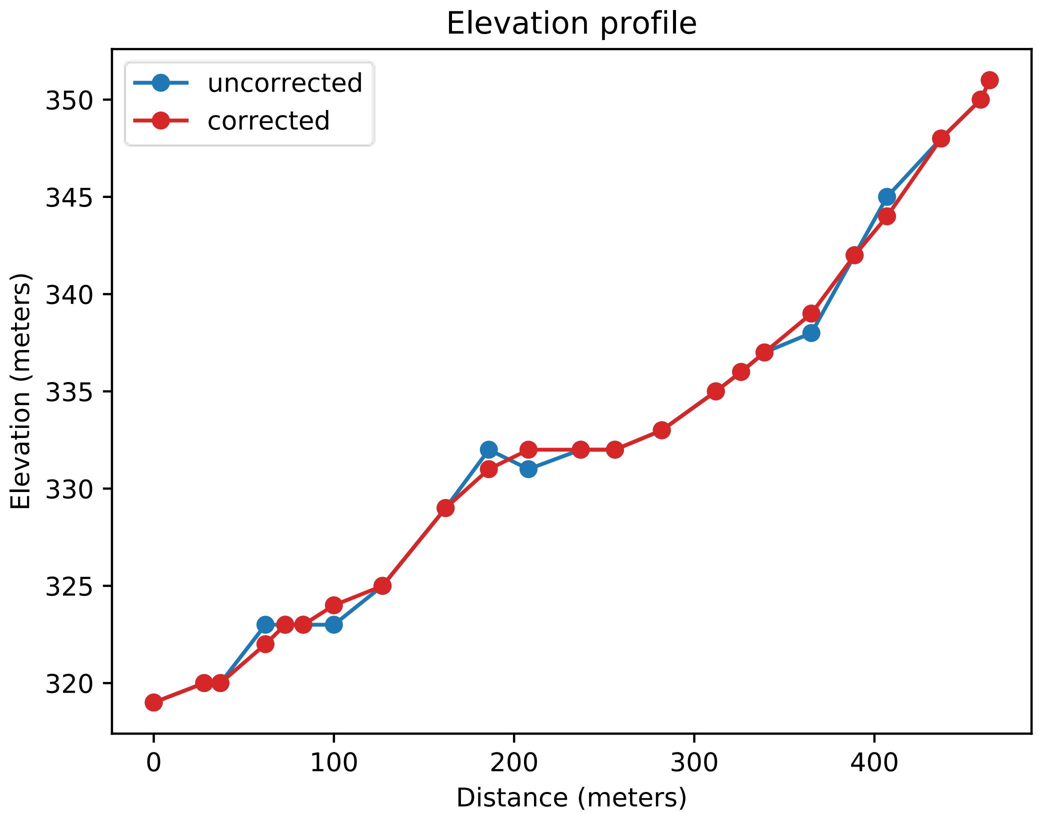

Figure 9: Uncorrected and corrected elevation profile of ascending road (part of Hexental road near Freiburg). Plot generated from data of the OSM ways *19794411*, *30354354*, *319722236*, *143661523*.

Evaluation: Correcting bridge

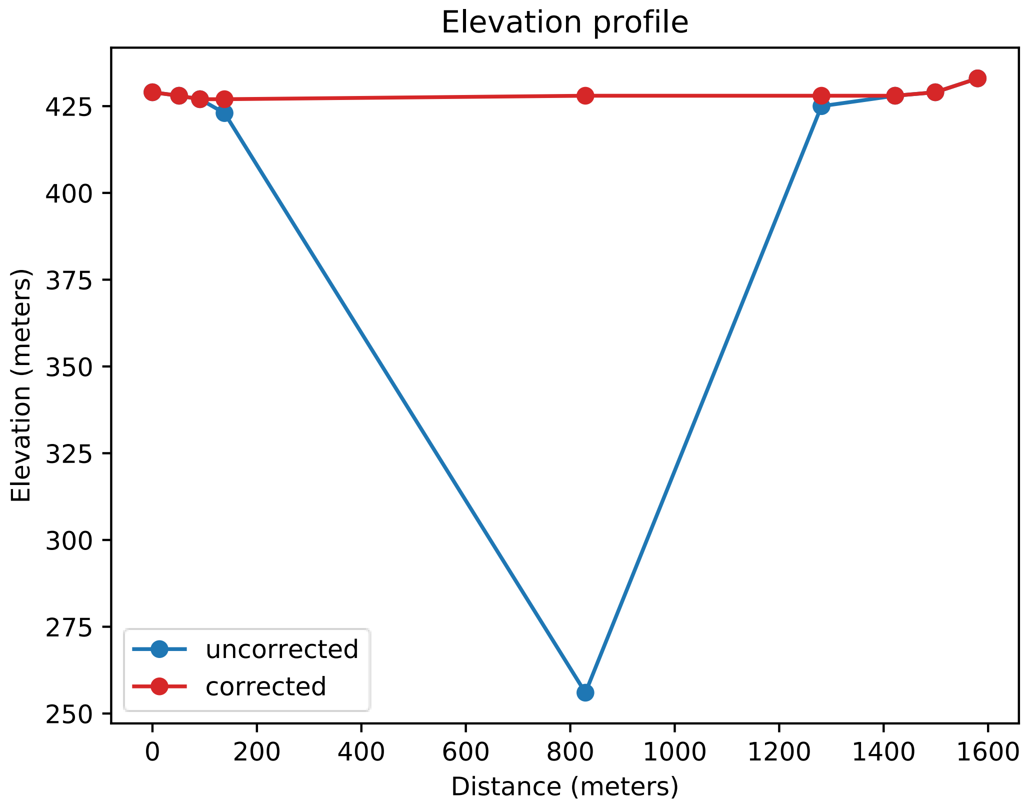

Figure 10: Uncorrected and corrected elevation profile of road crossing a bridge over a valley (Kocher viaduct). Plot generated from data of the OSM ways *403909403*, *320517373*, *320517370*, *24625636*, *24625669*.

Evaluation: Correcting tunnel

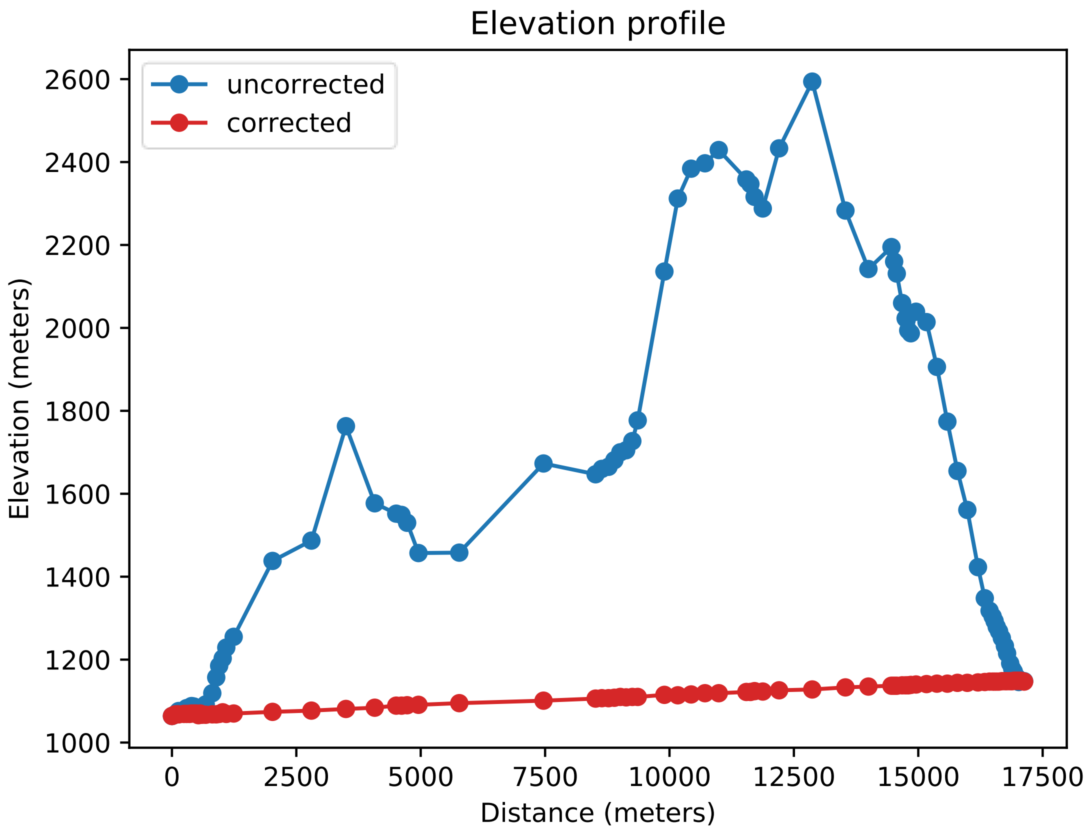

Figure 11: Uncorrected and corrected elevation profile of road going through a tunnel (Gotthard road tunnel). Plot generated from data of the OSM ways *304476718*, *4214708*, *49124512*, *431339440*, *29736069*.

Evaluation: Runtime adding elevation data

	germany	planet
runtime tool	2 minutes	102 minutes
runtime writing output OSM	4 minutes	61 minutes
maximum used main-memory	10 GB	94 GB

Table 4: Runtimes of different input OSM sizes when adding elevation data. Run on machine with *AMD Ryzen 3700X 8-Core/16-Threads and 128 GB Ram*. Data was read from and written to *2TB NVME Samsung 970 Evo+*.

Evaluation: Runtime correcting elevation data

	germany	planet
runtime tool	4 minutes	111 minutes
runtime writing output OSM	4 minutes	83 minutes
maximum used main-memory	9 GB	95 GB

Table 5: Runtimes of different input OSM sizes when correcting elevation data. Run on machine with *AMD Ryzen 3700X 8-Core/16-Threads and 128 GB Ram*. Data was read from and written to *2TB NVME Samsung 970 Evo+*.

Conclusion

- We provide a tool to annotate the complete OSM data with elevation data
- We provide a second tool that performs the following corrections on elevation data in OSM:
  - Smooth short-term fluctuations in elevation along routes
  - Correct the elevation of tunnels and bridges
  - Make sure that rivers do not flow uphill
- Source code (GPLv3):

  https://github.com/us58/osmelevation

References

[1] TagInfo. *key=ele − Taginfo*, [Online; accessed 27-January-2022]. 2022. URL: https://taginfo.openstreetmap.org/keys/?key=ele.

[2] German Unity Transport Project 8. *The new line from Erfurt to Leipzig/Halle*, [Online; accessed 09-March-2022]. 2022. URL: https://www.vde8.de/Overview------_site.site..ls_dir._nav.373_likecms.html.

[3] Andreas Eckner. Algorithms for unevenly spaced time series: Moving averages and other rolling operators. *Working Paper*, 2019. http://www.eckner.com/papers/Algorithms%20for%20Unevenly%20Spaced%20Time%20Series.pdf.

Inverse distance weighting interpolation

- For a finite number N of known data points D_i, $d[P, D_i]$ denotes the distance of a point P to a known data point D_i
- z_i denotes the value of a known data point D_i
- The interpolated value of P can be calculated with the function

$$f(P) = \begin{cases} \frac{\sum_{i=1}^{N} d[P, D_i]^{-u} z_i}{\sum_{i=1}^{N} d[P, D_i]^{-u}} & \text{if } d[P, D_i] \neq 0 \text{ for all } D_i \\ z_i & \text{if } d[P, D_i] = 0 \text{ for some } D_i \end{cases}$$

- Greater values of $u > 0$ increase the influence of known data points closest to P (we use $u = 2$)

Andreas Shepard. A two-dimensional interpolation function for irregularly-spaced data. *Proceedings of the 1968 ACM National Conference, page 517–524*, 1968.

Spanning plane between two coordinates

(1) startVector = (plane start lon, plane start lat, plane start elevation)

(2) endVector = (plane end lon, plane end lat, plane end elevation)

(3) directionVector1 = endVector – startVector

(4) directionVector2 = (-1 * directionVector1.y, directionVector1.x, 0)

(5) planeNormalVector = directionVector1 × directionVector2

(6) d = -1 * (planeNormalVector · startVector)

Then,

$$z = \frac{-\mathrm{d} - \mathrm{planeNormalVector.x} * \mathrm{x} - \mathrm{planeNormalVector.y} * \mathrm{y}}{\mathrm{planeNormalVector.z}}$$

yields the elevation *z* for given x- and y-coordinates on the plane.