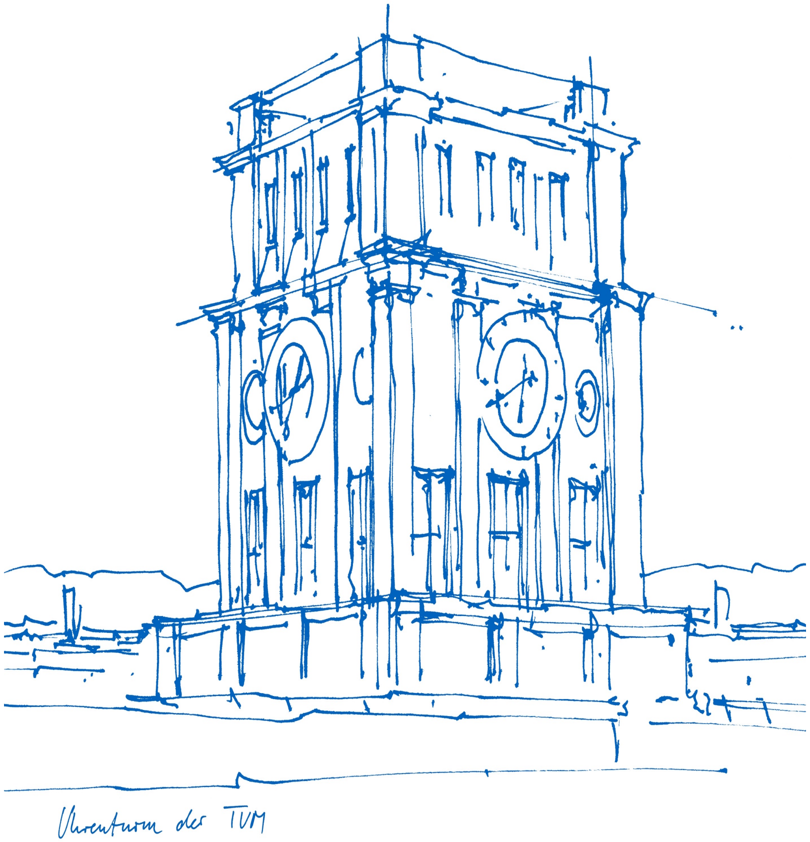
Geoinformation

Prof. Dr.-Ing. Liqiu Meng

Chair of Cartography and Visual Analytics

WS 24/25



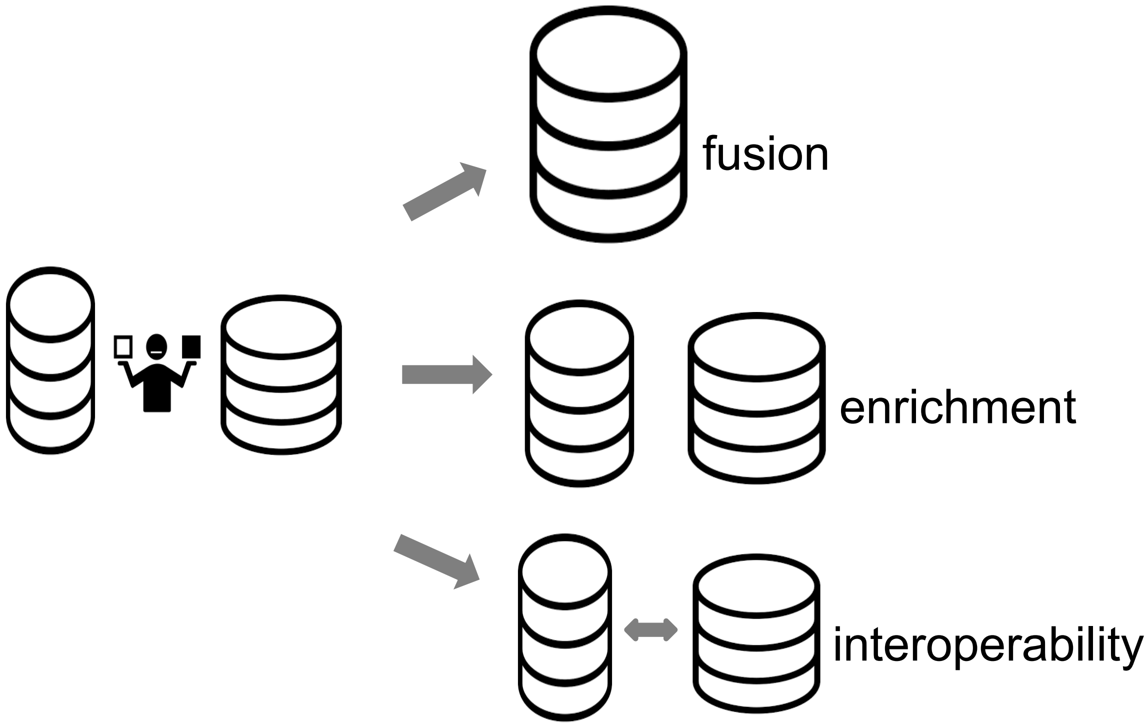
Contents of Lectures

Cartographic techniques

I Spatial data integration

II Map generalization algorithms

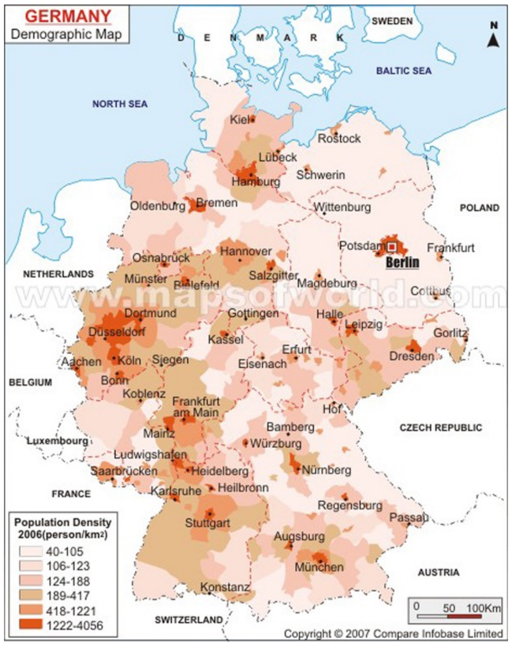
I. Spatial data integration

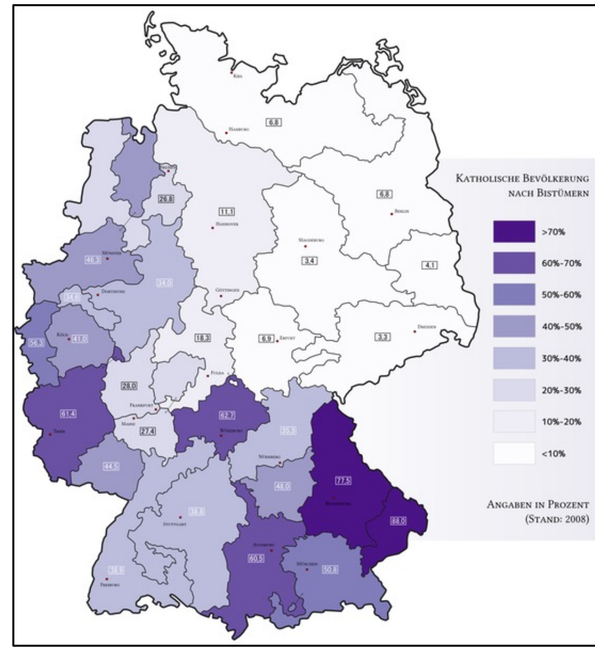


Differences in
- themes
- coordinate systems
- spatiotemporal scales
- data suppliers

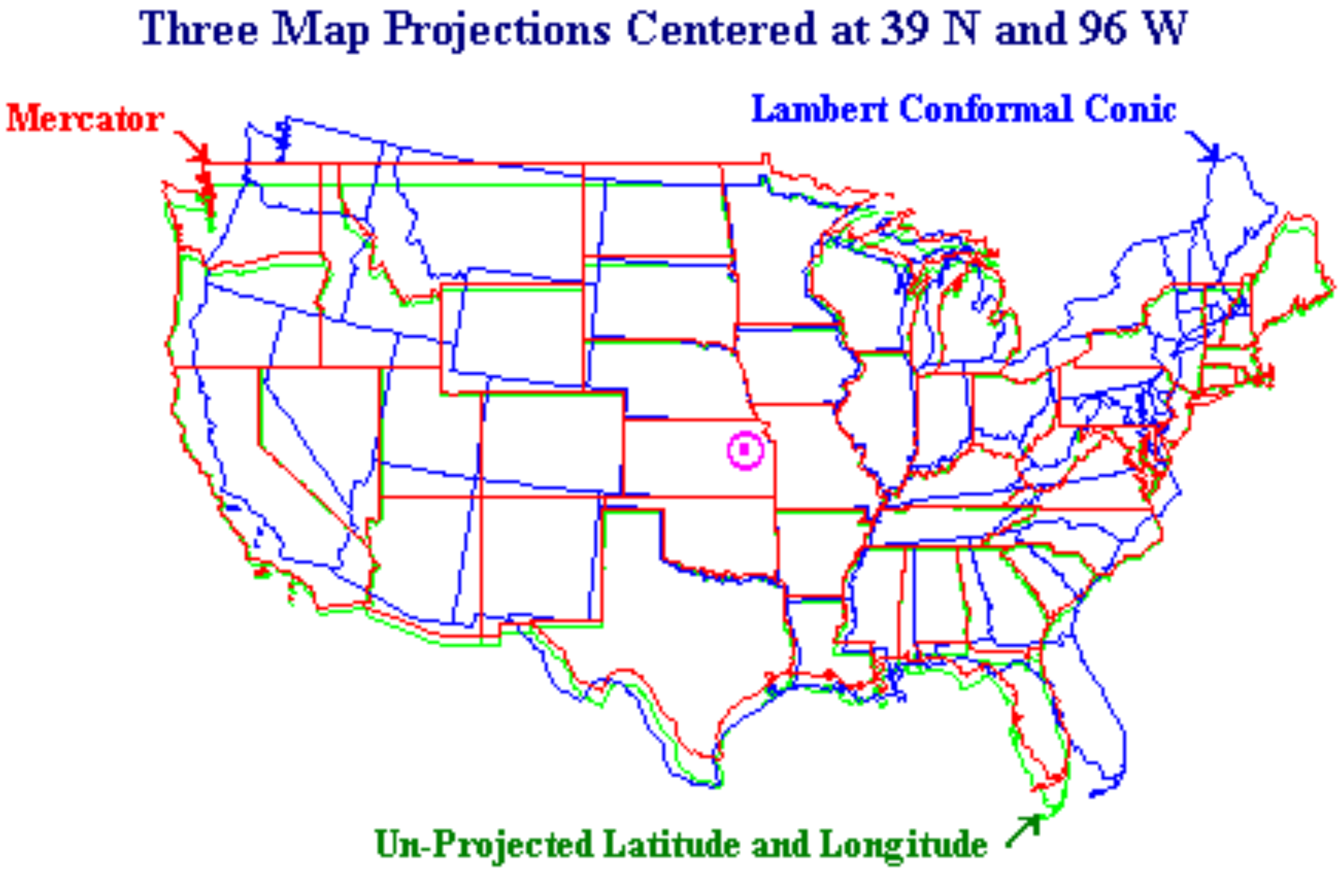
- Bring many data sources describing the same region together (fusion)
- Enhance the richness and quality of the individual datasets (enrichment)
- Link different datasets (interoperability)

Different themes

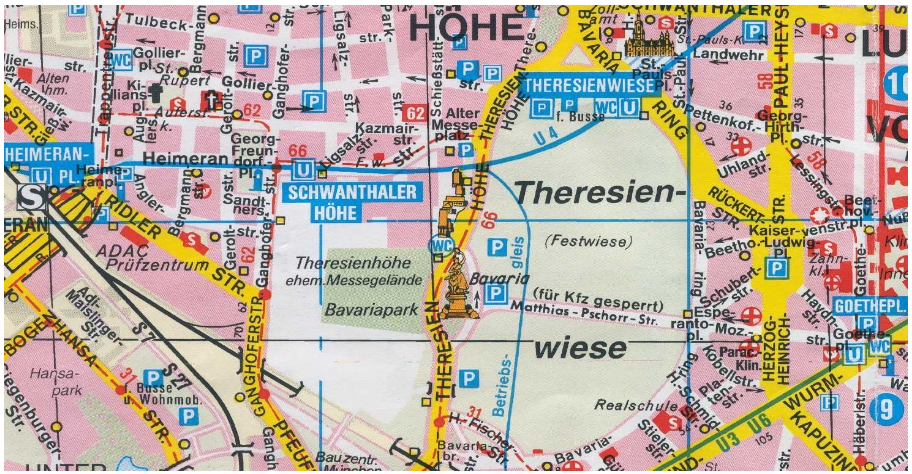


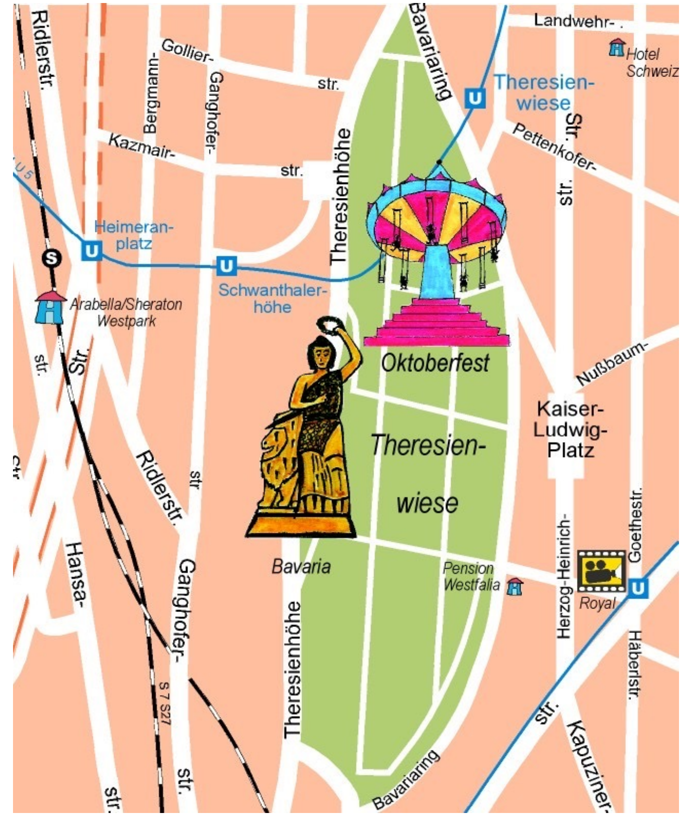


Different map projections

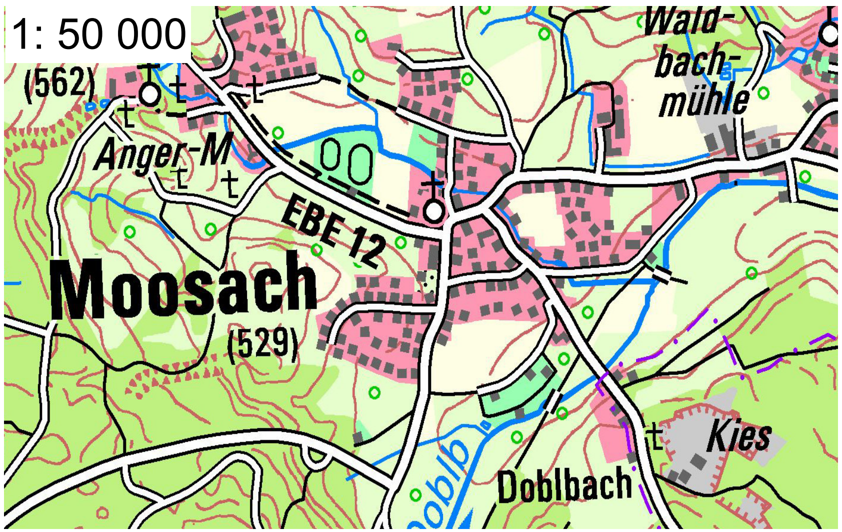


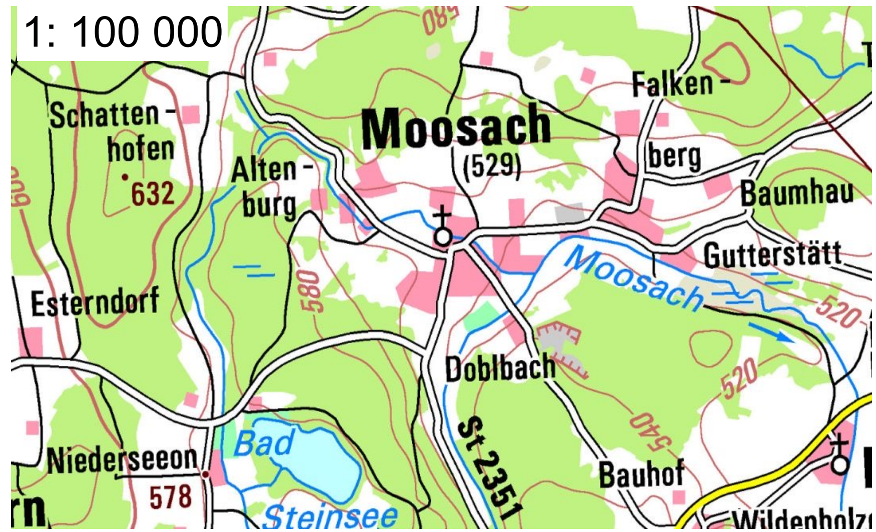
Different geometries driven by purposes





Different map scales





Working principle of spatial data integration

The number of objects
The number of attributes

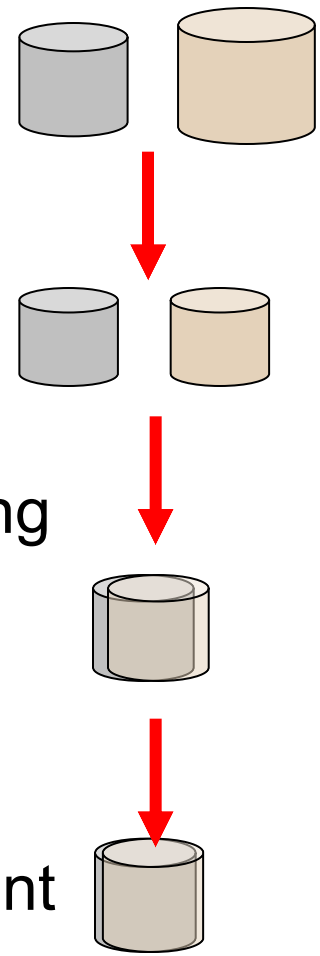
Data preprocessing

Geometry
Topology
Semantic attributes

Automatic linking

Link file
Quality measures
Types of mistakes

Interactive improvement



Results of integration

A case study of data integration

Task

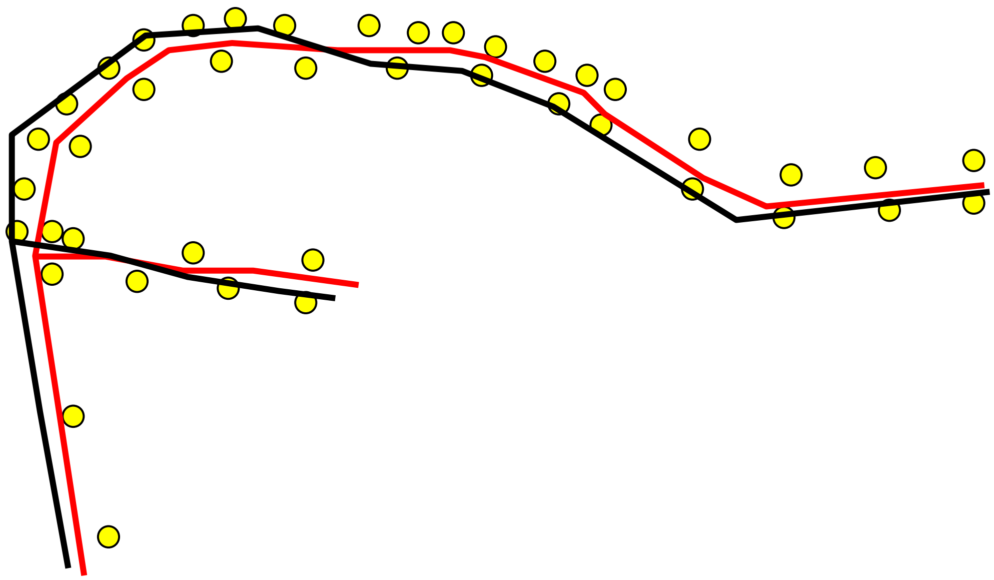
Integration of house number with streets from Basis-DLM

Problem

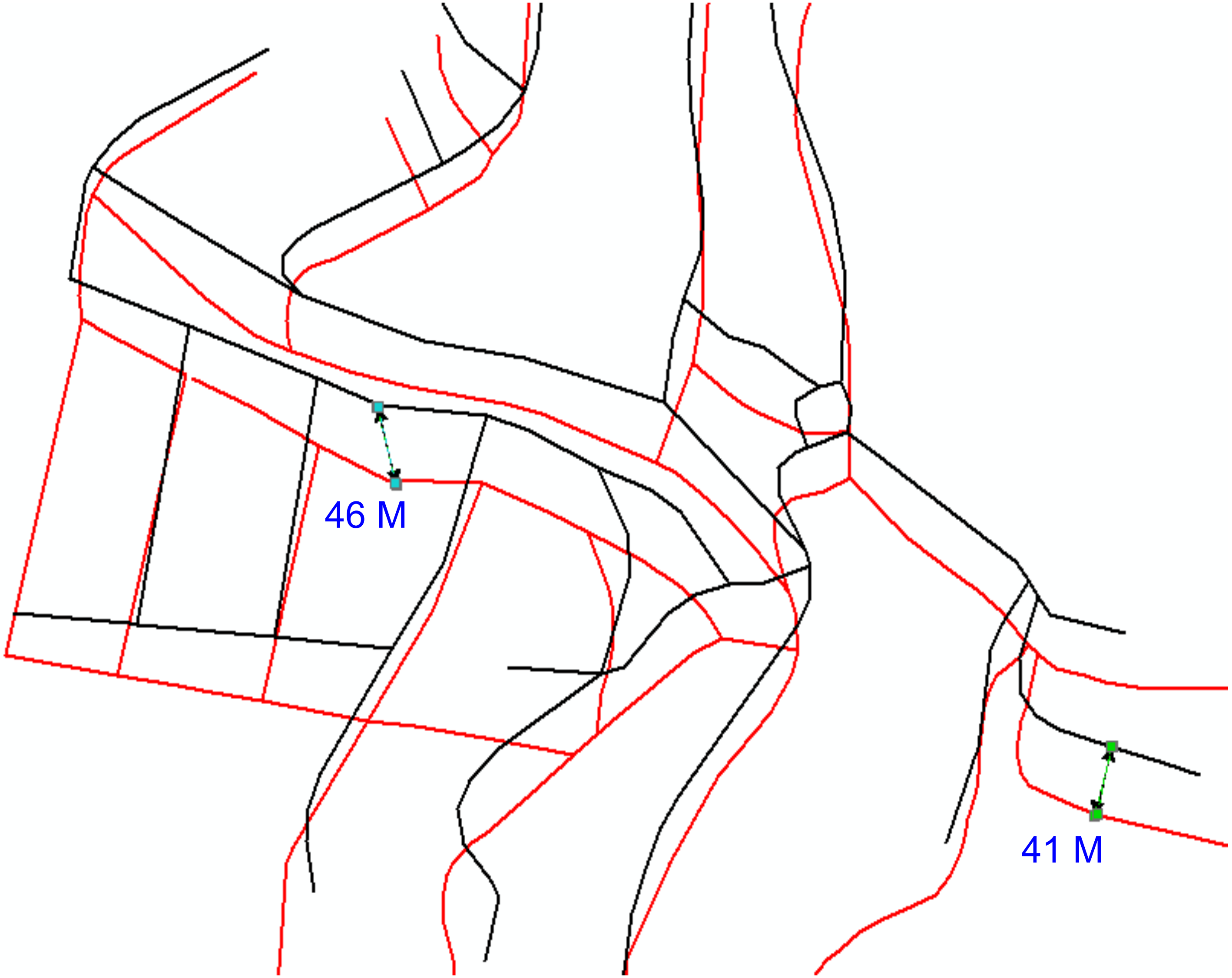
House number were captured based on TomTom street data

Solution

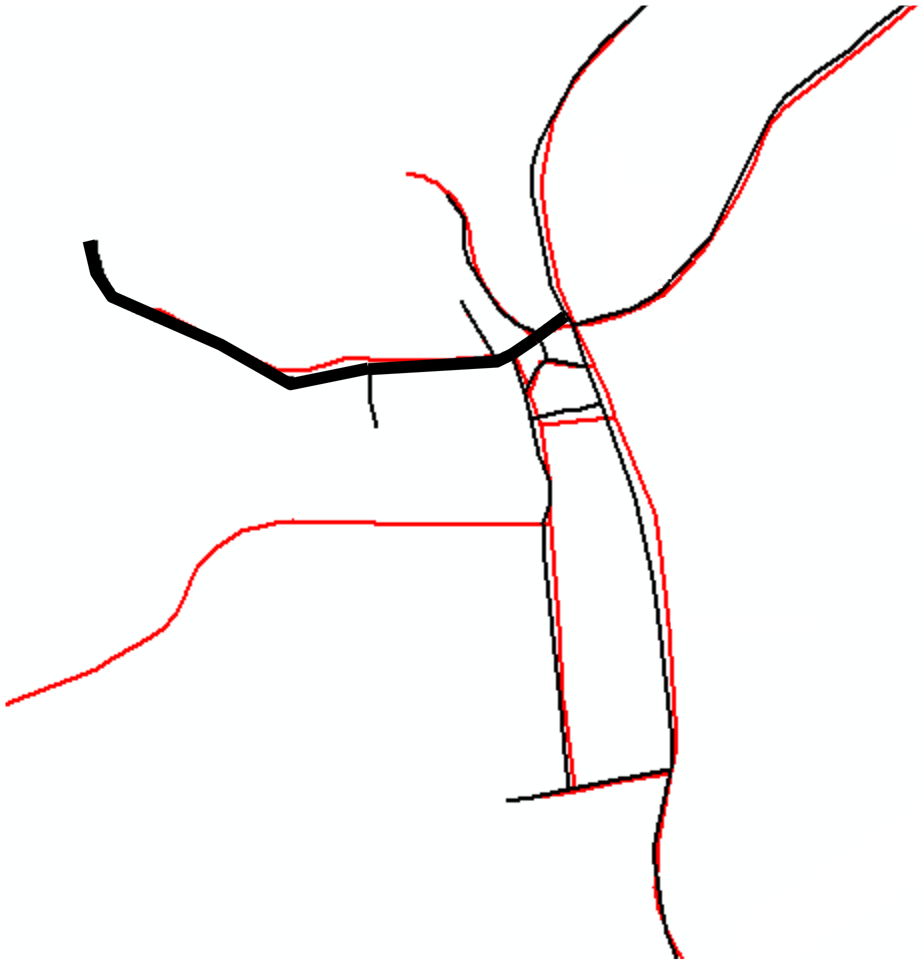
Matching between TomTom and Basis-DLM



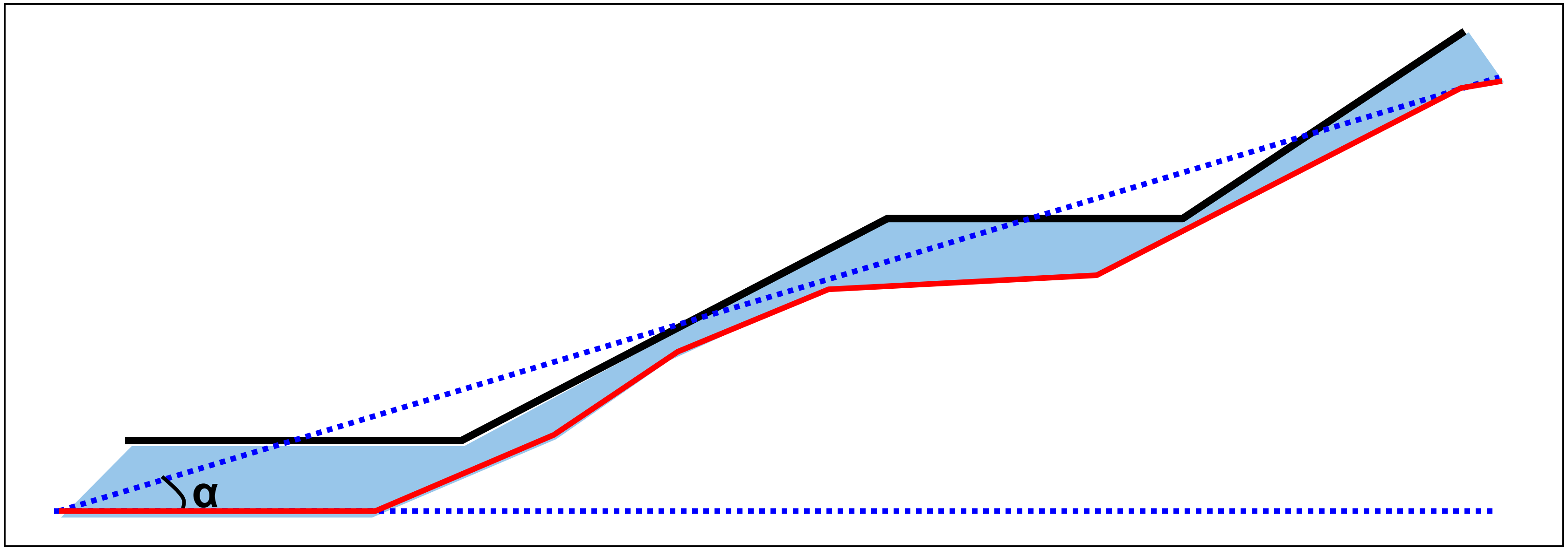
Reference data: Basis-DLM (black)
Candidate data: TomTom (red)



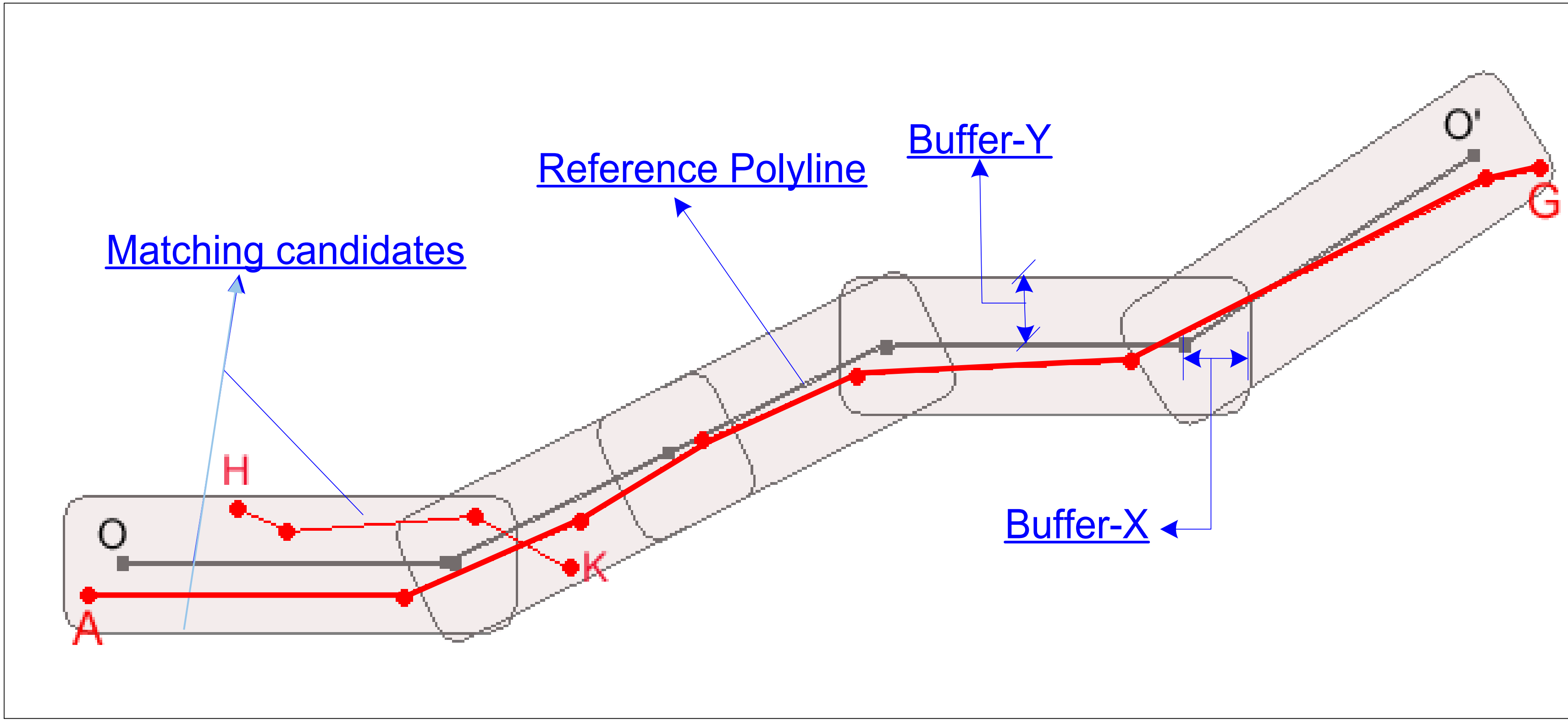
Analysis of geometric discrepancy



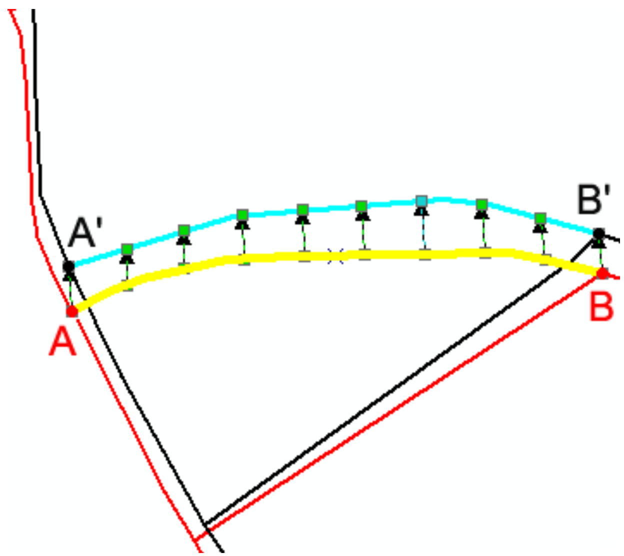
- Chain the segments sharing some common attribute values into long strokes,
- Compare the similarity of corresponding strokes,
- Grow to further strokes in the network

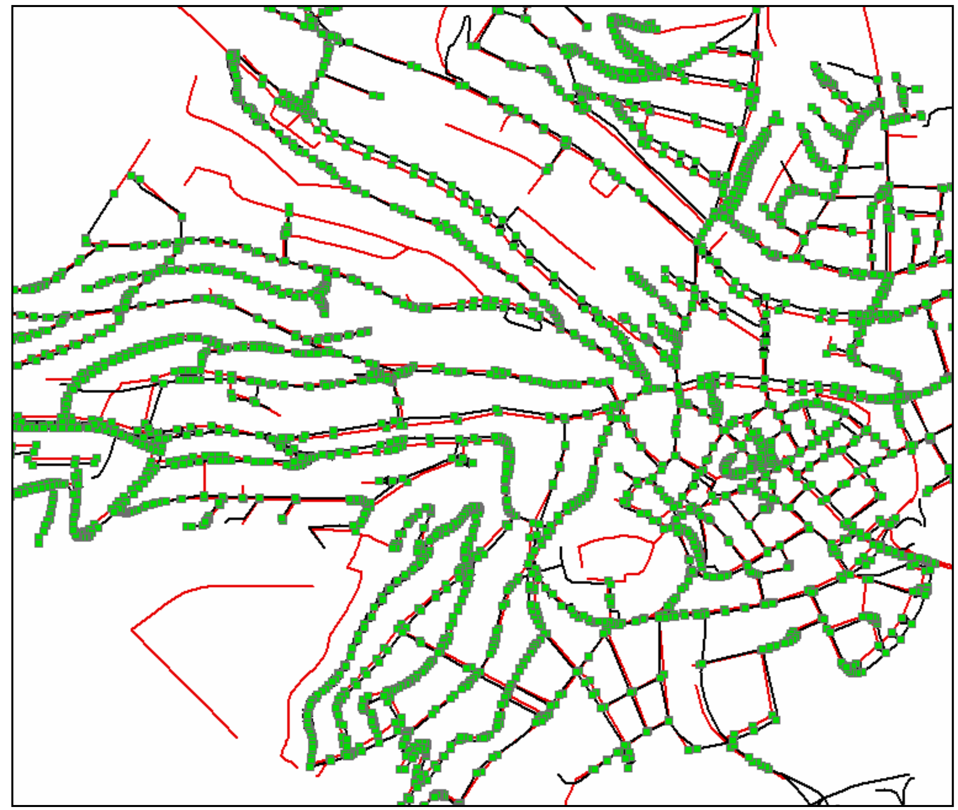


Computing the similarity of each candidate according to a weighted criterion (length, angle, area etc.)

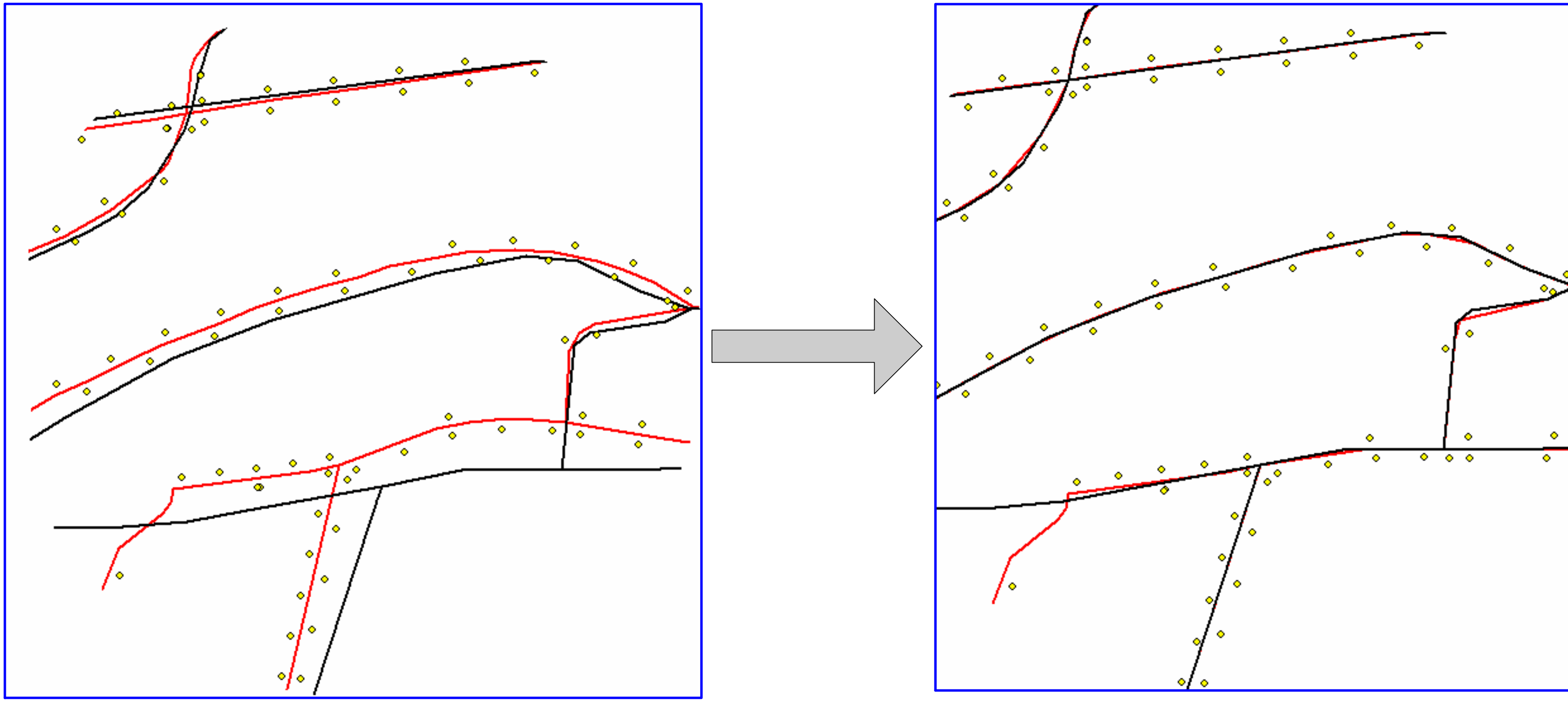


- Constructing buffers around segments and identifying matching candidates from the other database
- Selection of candidates with the largest similarity (≥ threshold)

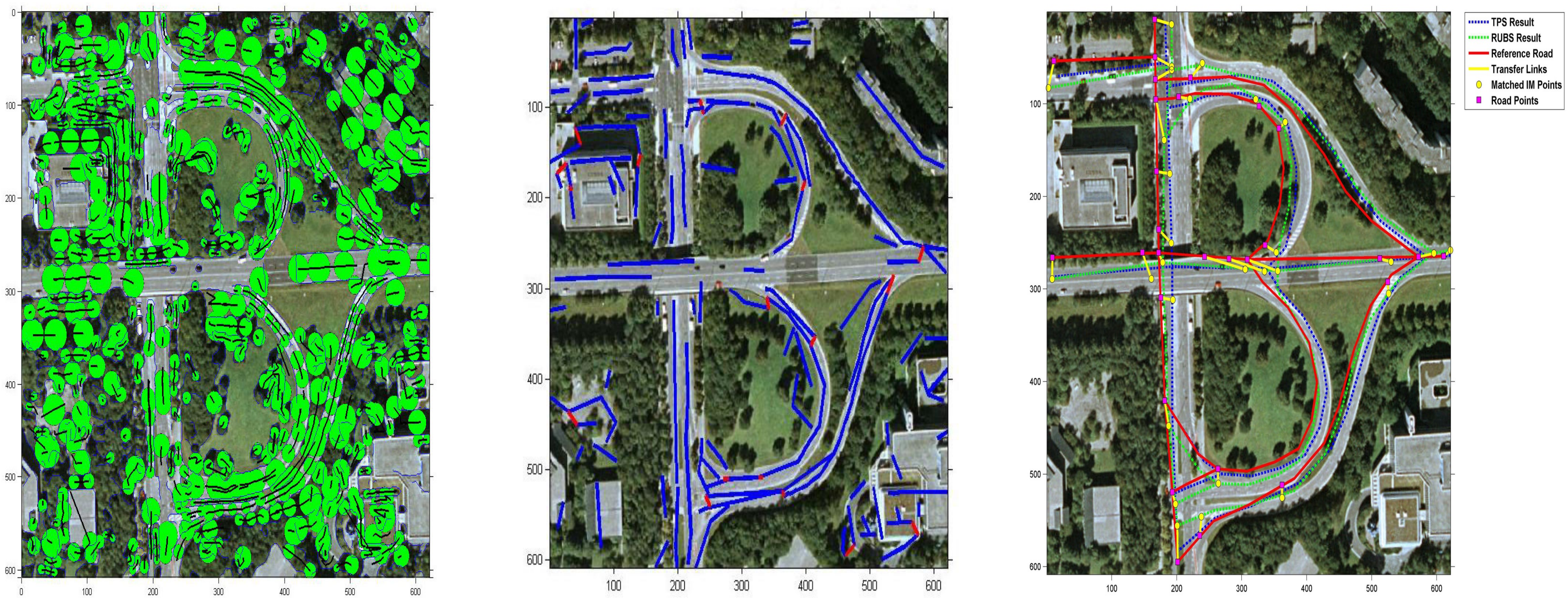




Establishment of links between line segments



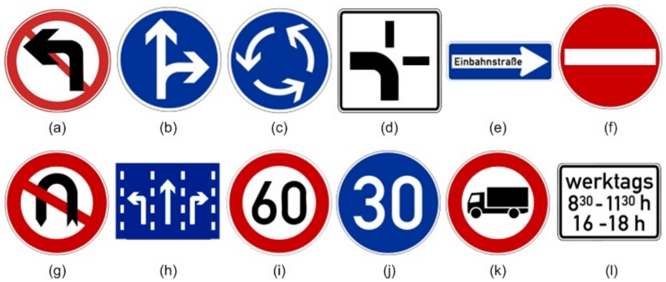
Project house numbers and other POIs to the reference database based on the rubber-sheeting principle

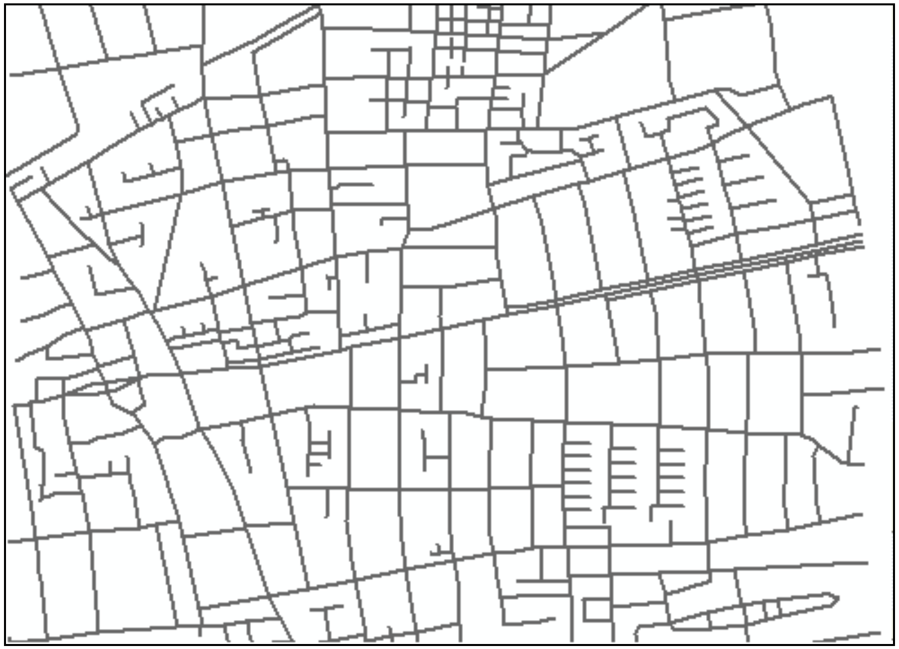


Matching between raster and vector data

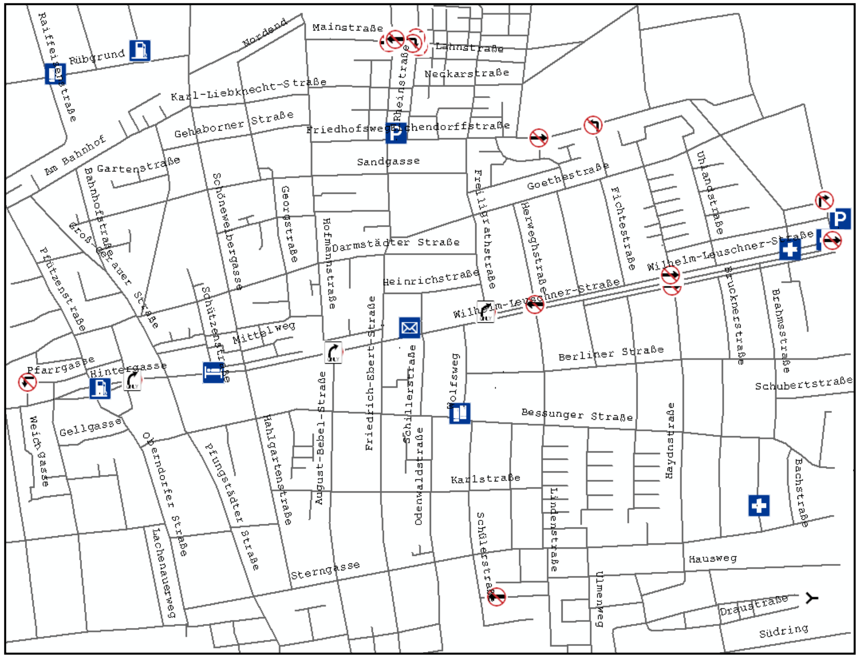
- Identify middle axes belonging to road segments in the raster image
- Chain them into longer strokes
- Align the strokes with vector roads

Enrichment with navigation / landmark information



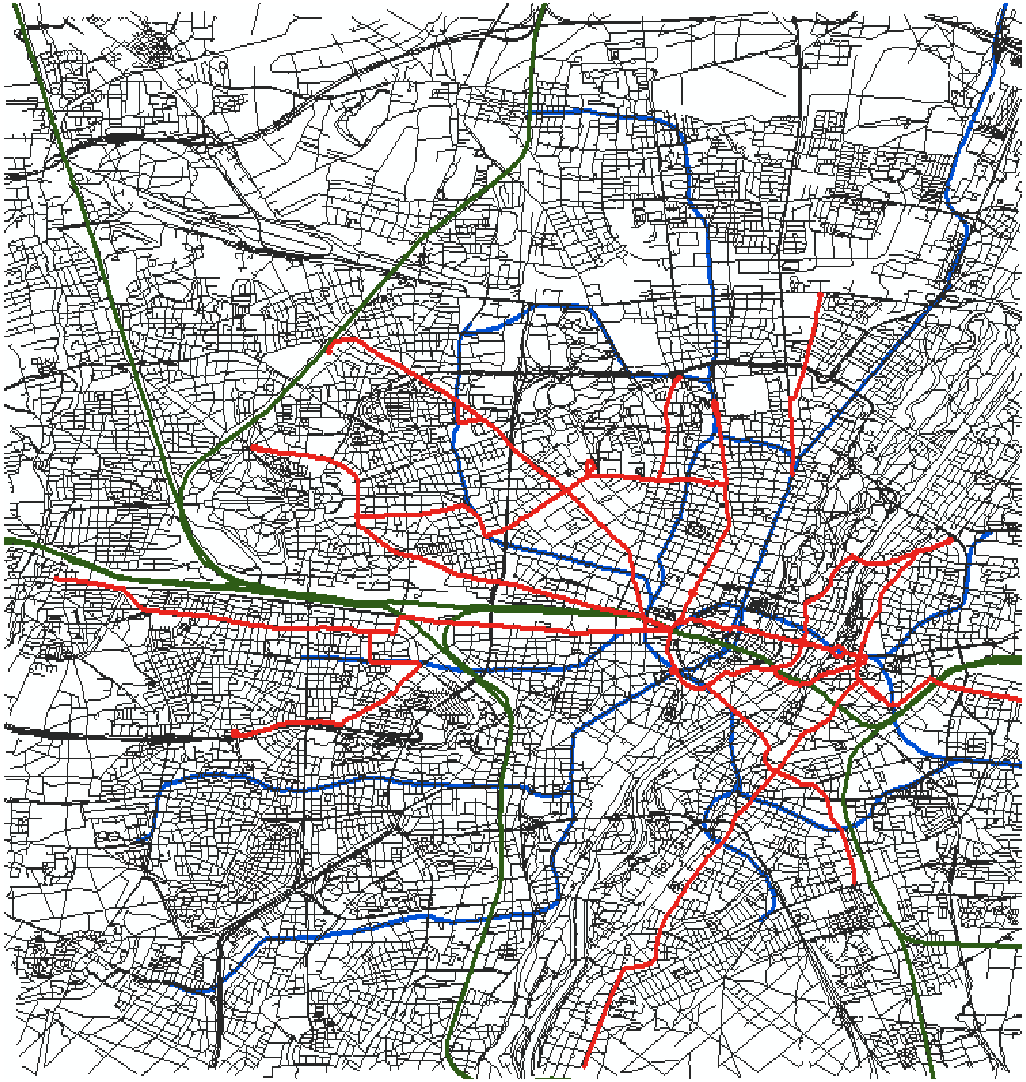


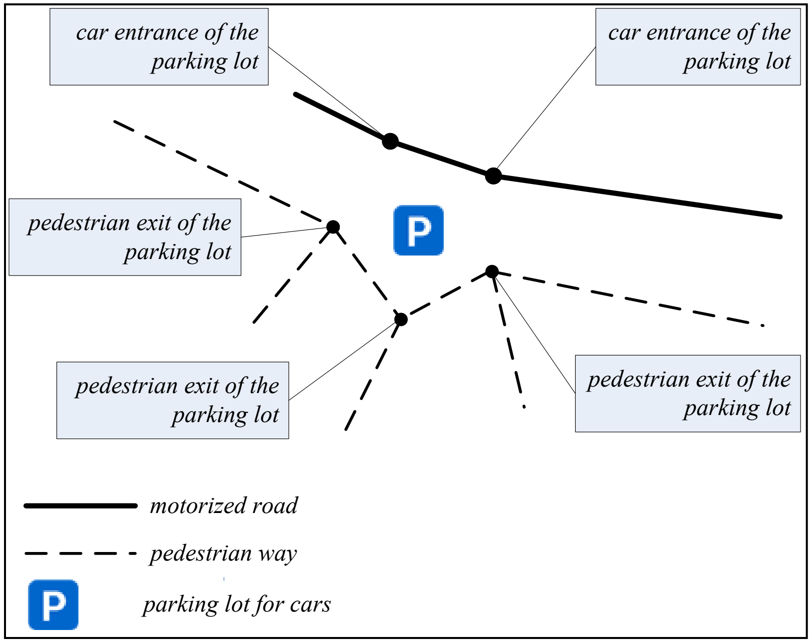
streets from digital landscape model



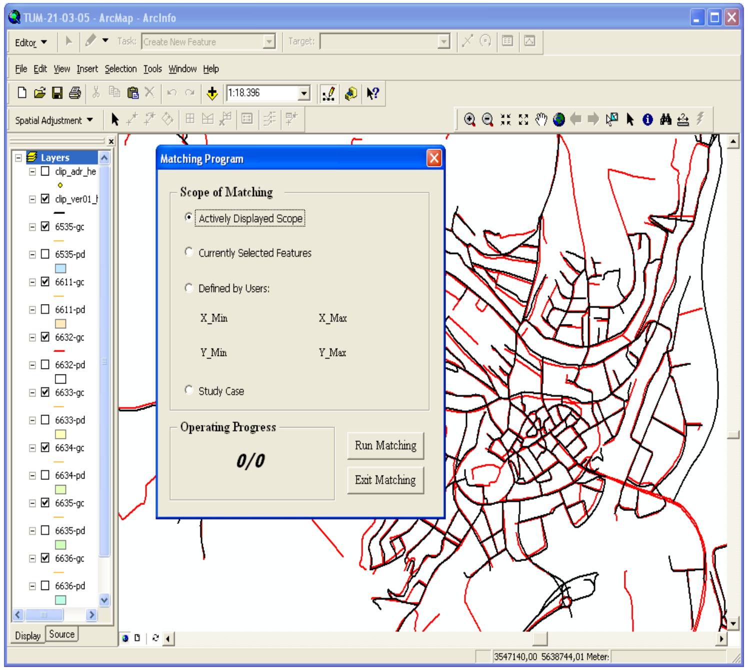
after integration

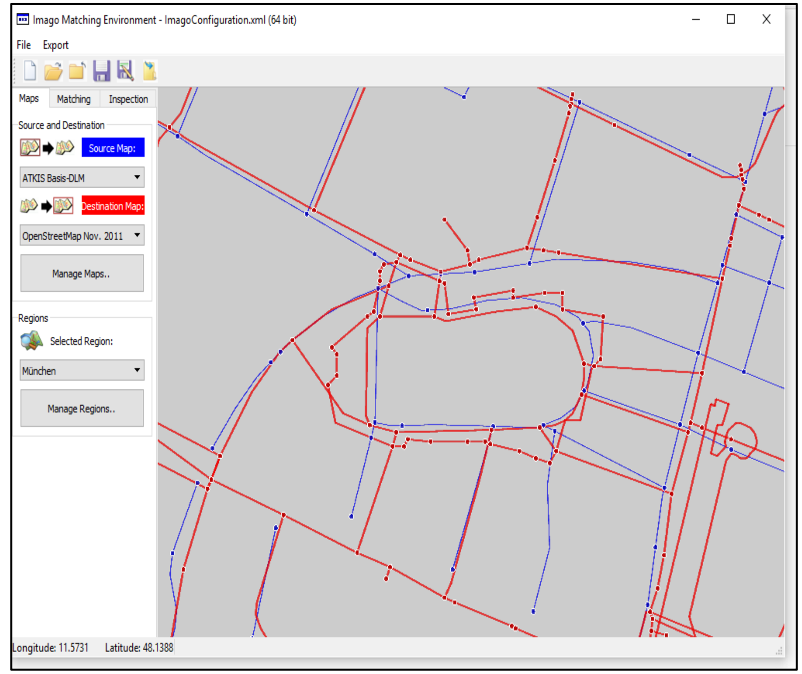
Cell-based docking of pedestrian roads (grey) to motor ways (brown)





Fusion of motorways, pedestrian roads and public transport network in Munich (Tram - red, S-Bahn – green, U-Bahn – blue)



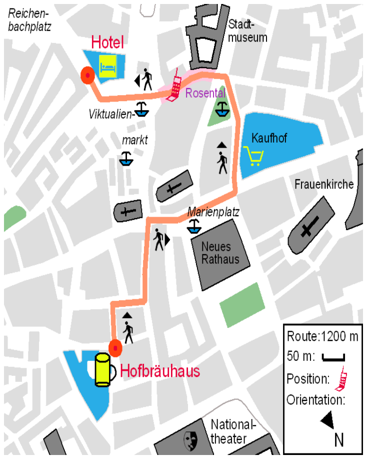


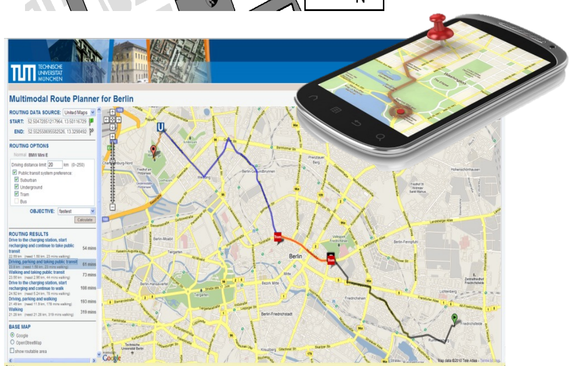
Algorithms
- Delimited stroke
- Buffer growing
- Iterative Hierarchical Conflation
- Sparse matching
- Conditional Random Field
- Switch-point docking
- …

Quality indicators
- Connectivity
- Computing speed
- Robustness
- Repeatability
- Transferability
- Transparency
- …

Applications of data integration

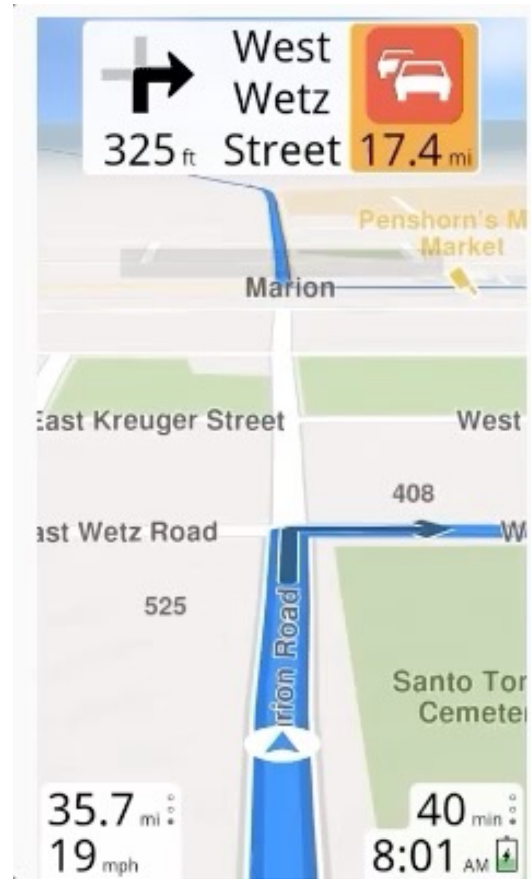
Landmark-based navigation





Multimodal navigation

Augmented reality



Adaptive text placement

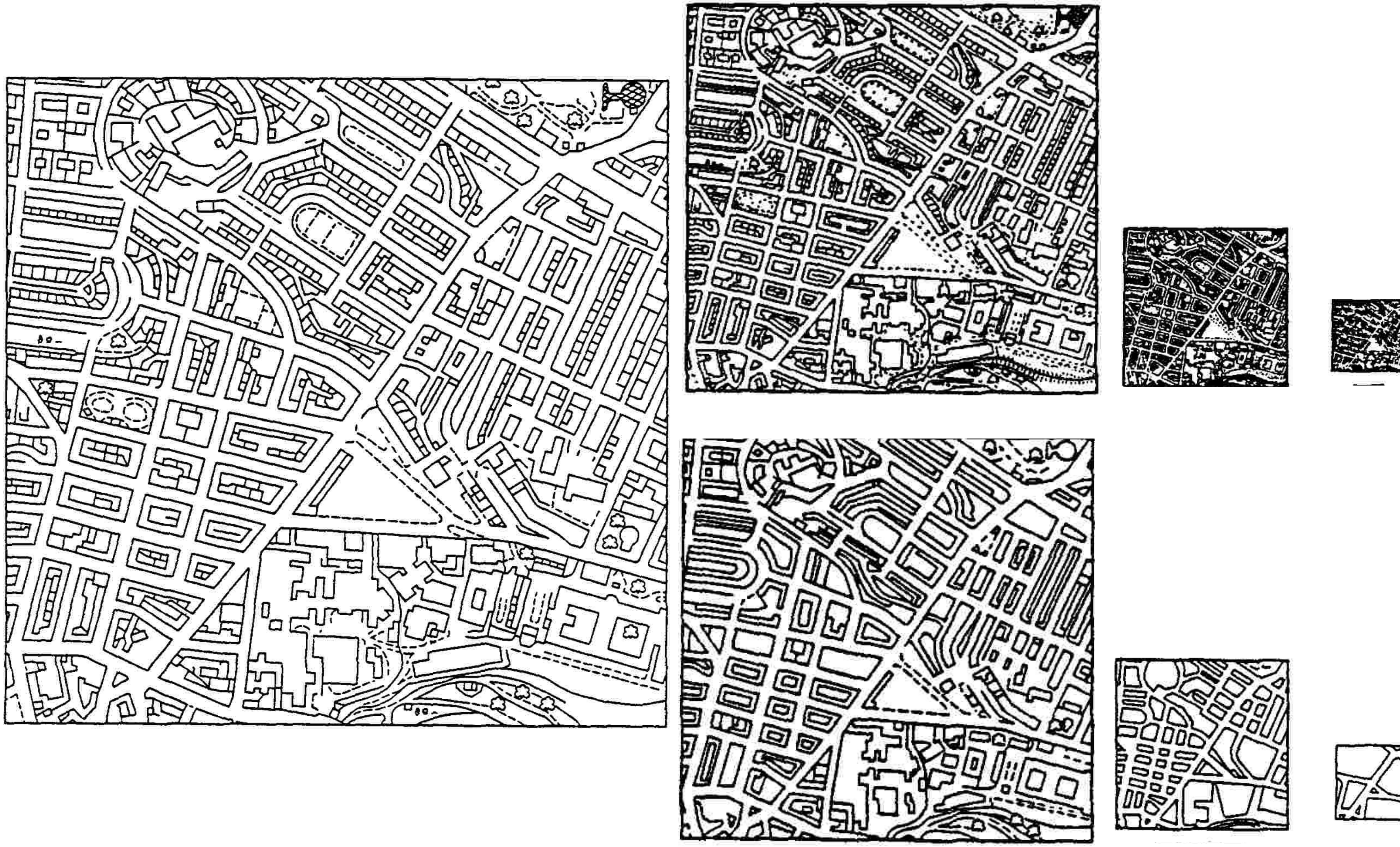
II. Map generalization algorithms

Goals

- Reduce the data amount
- Enhance the abstraction level
- Harmonize the level of details of different data sources
- Improve the legibility of the map

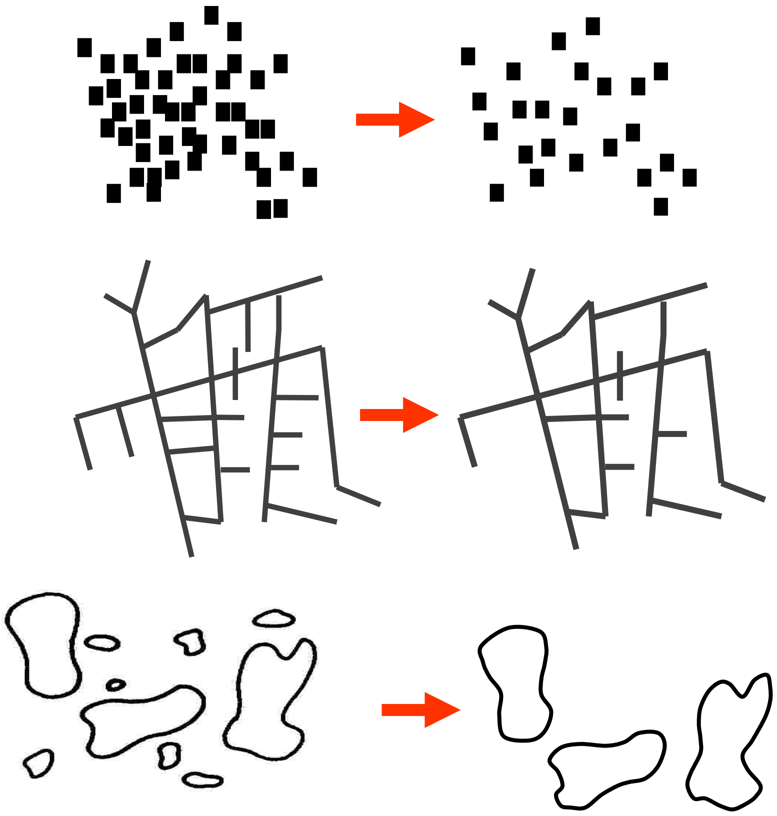
Generalization leads to

- continuous simplification and
- abrupt changes



Automatic generalization operations

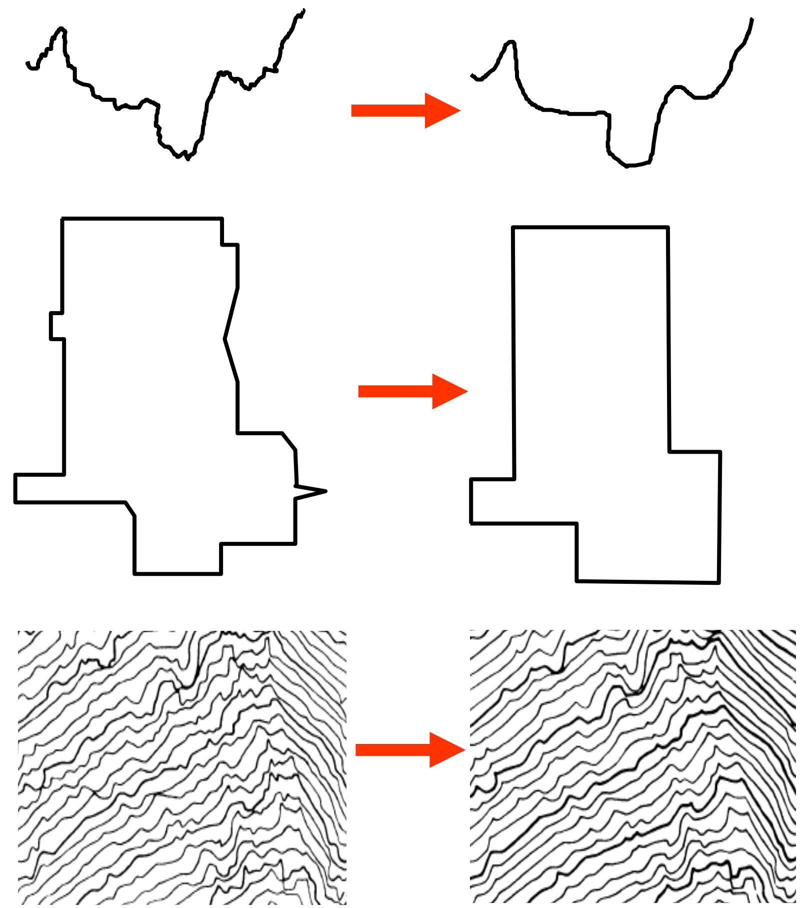
Selection / Elimination



- It works on groups of discrete objects.
- Important objects are preserved, unimportant objects are eliminated.
- It results in a reduced number of objects and distribution density.

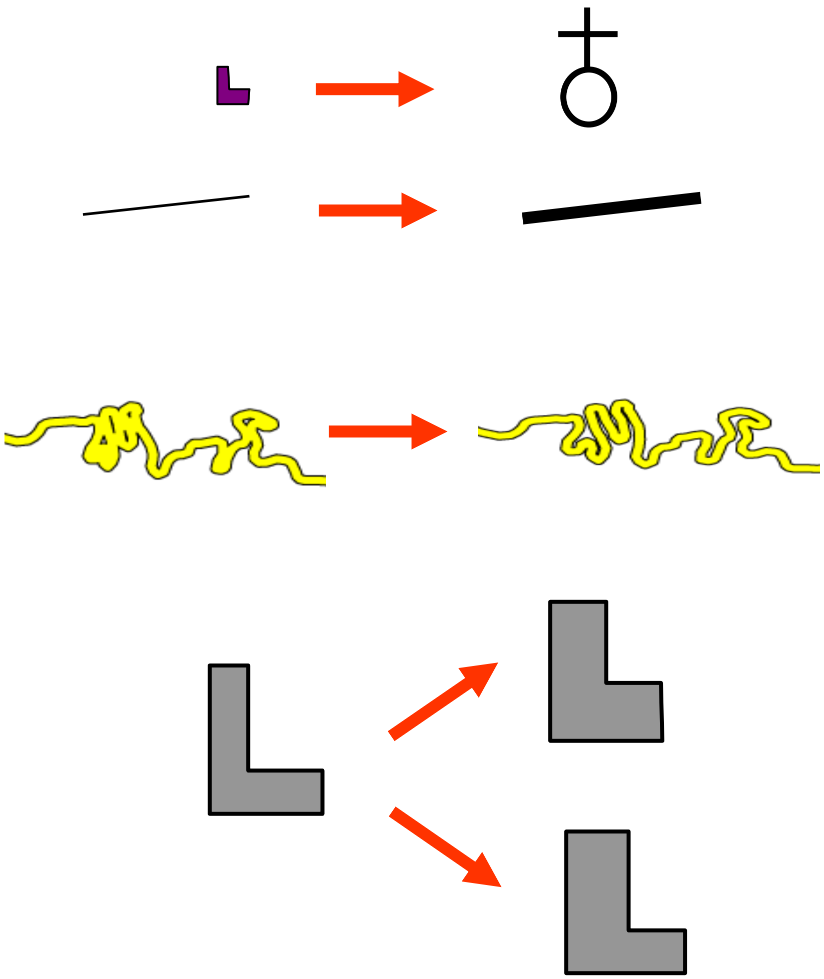
Graphic simplification

- It works on individual line and polygon objects (or object groups).
- Important details are preserved, unimportant details are eliminated.
- It results in a reduced graphic complexity of the concerned object.



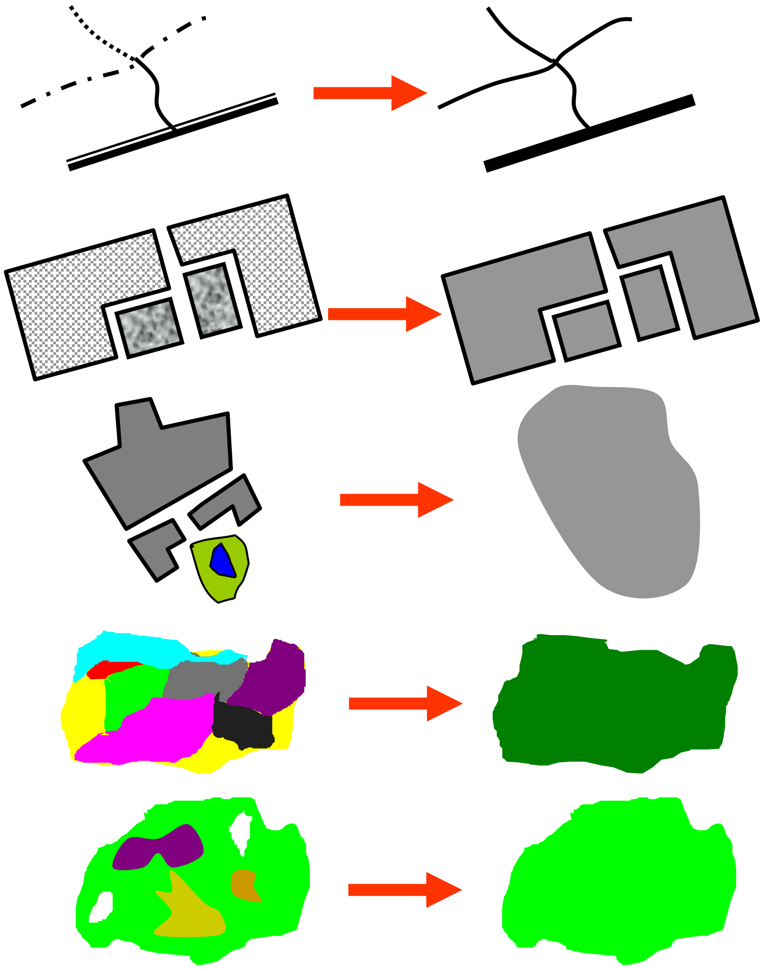
Graphic exaggeration

- It works on any kind of individual objects.
- Important objects (or object parts) are enlarged.
- It leads to an enhanced clarity and legibility of important contents.



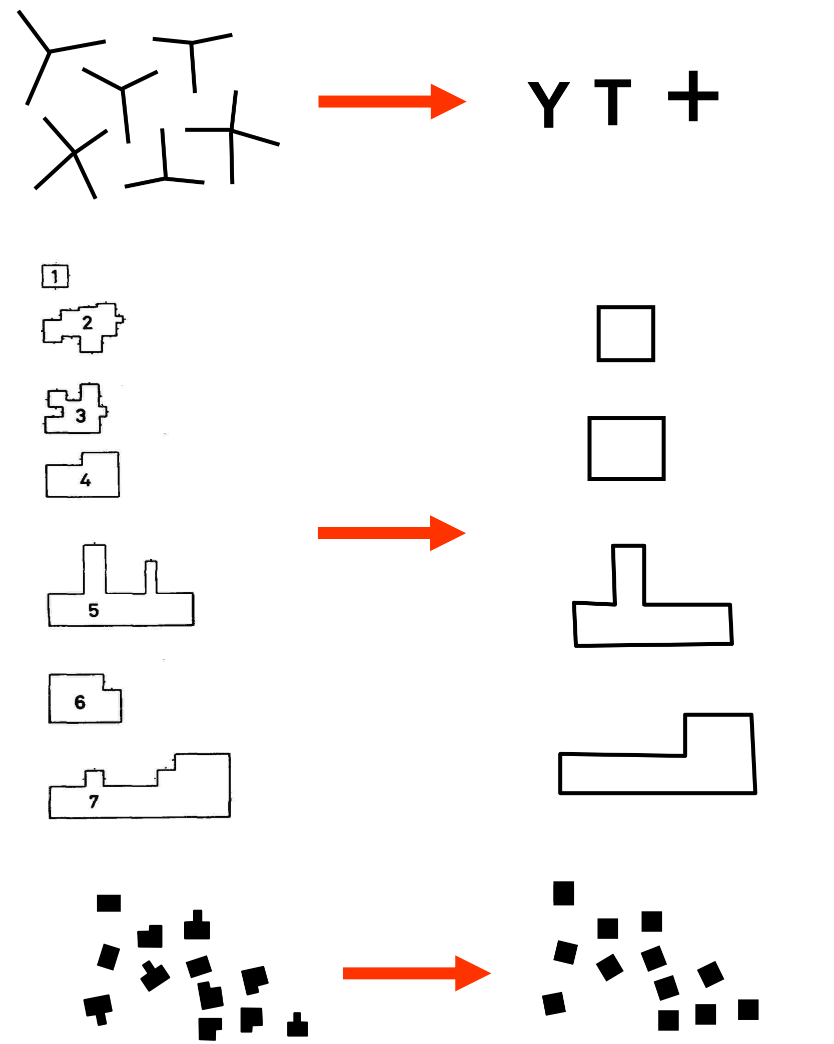
Classification

- It works on discrete objects.
- Common or dominant characteristics of the objects are preserved. Subtle distinctions are ignored.
- It results in reduced semantic variations and a higher level of semantic abstraction.



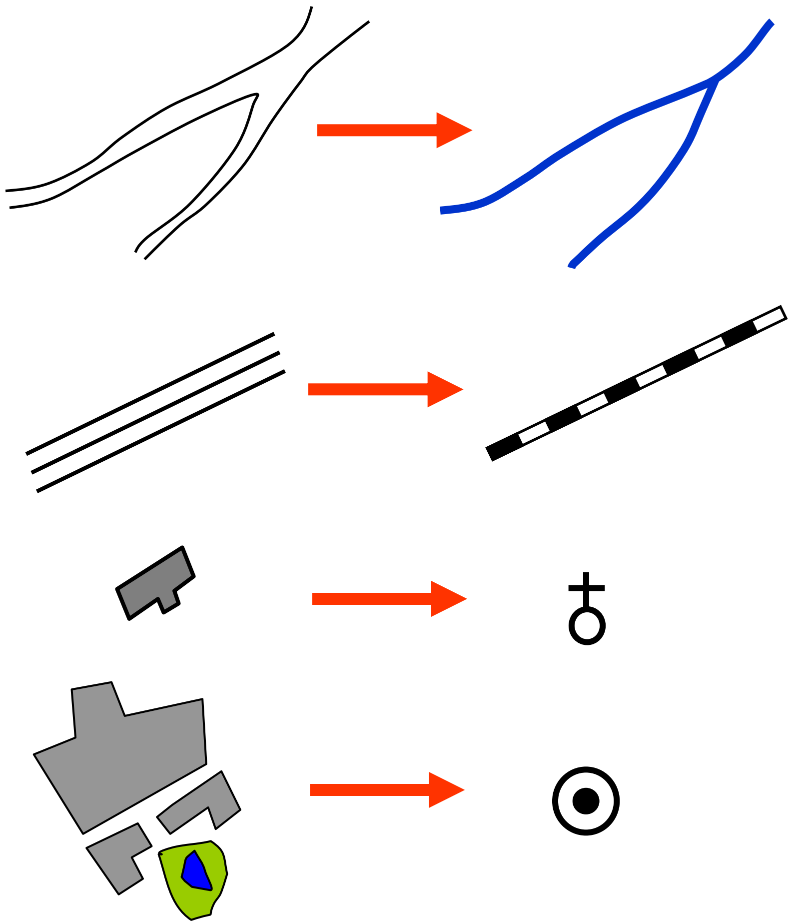
Graphic typification

- It works on discrete objects.
- Various forms are replaced by a few prototypical forms.
- It results in reduced form variations and an improved visual perception of spatial distributions.



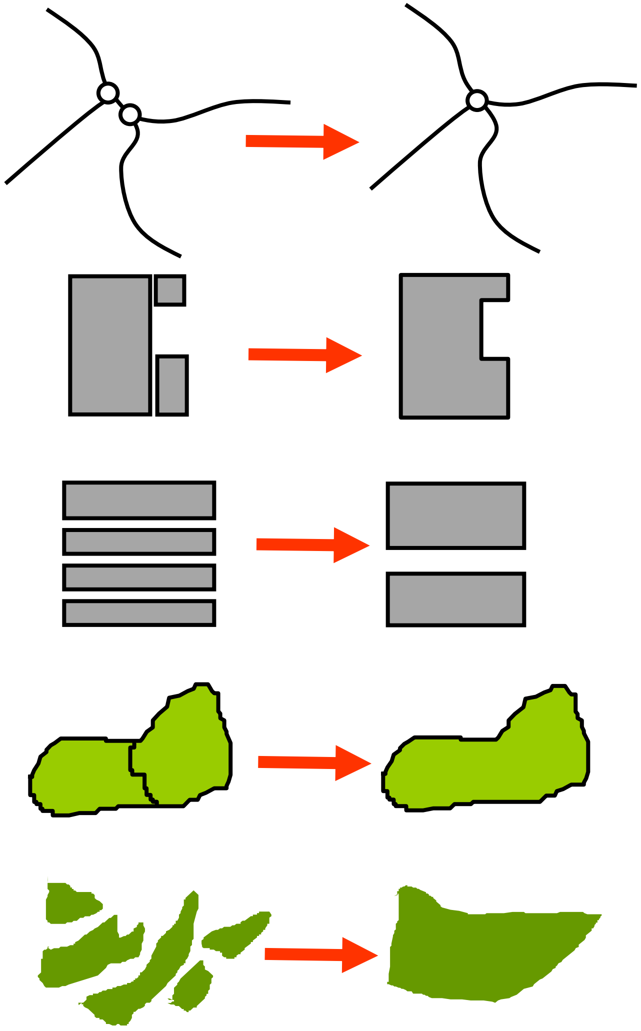
Dimensional collapse

- It works on discrete objects.
- Primary characteristics of the objects are preserved. Secondary characteristics are ignored.
- It results in an improved clarity and a higher level of semantic abstraction.



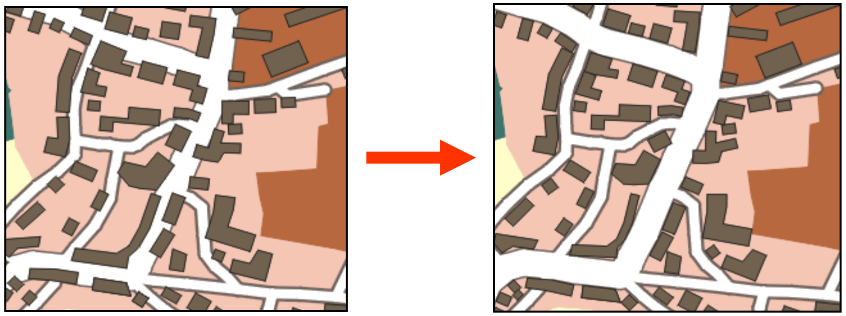
Aggregation / Amalgamation

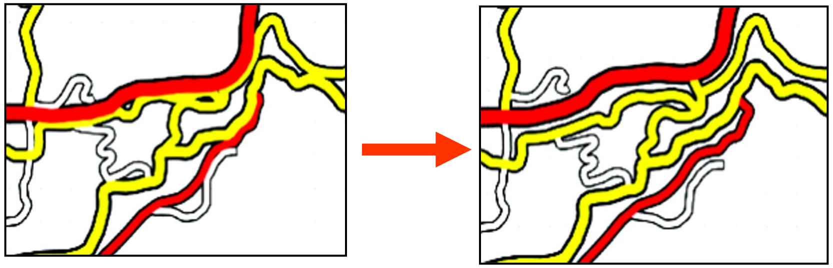
- It works on proximity objects.
- Common boundaries, small gaps or bridges between proximity objects are eliminated. Universal forms are preserved.
- It results in a reduced number of objects and an improved clarity.

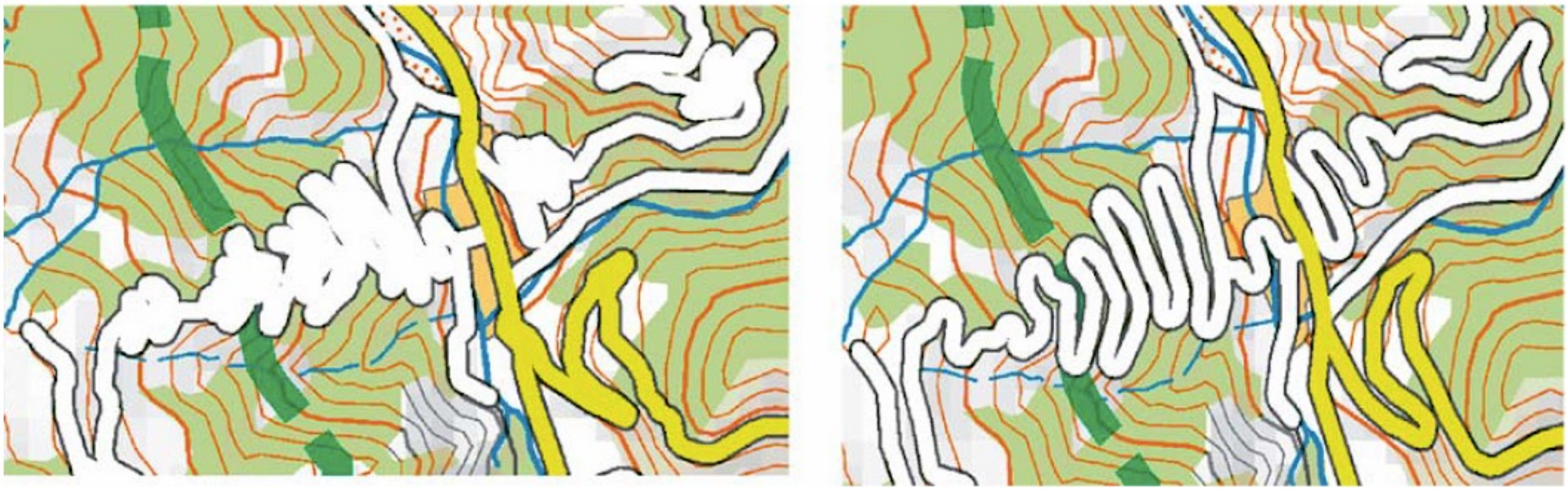


Displacement

- It works on spatially congested objects.
- Important objects remain on their locations. Unimportant objects are moved away.
- It results in an improved clarity and correct map graphics.







© IGN

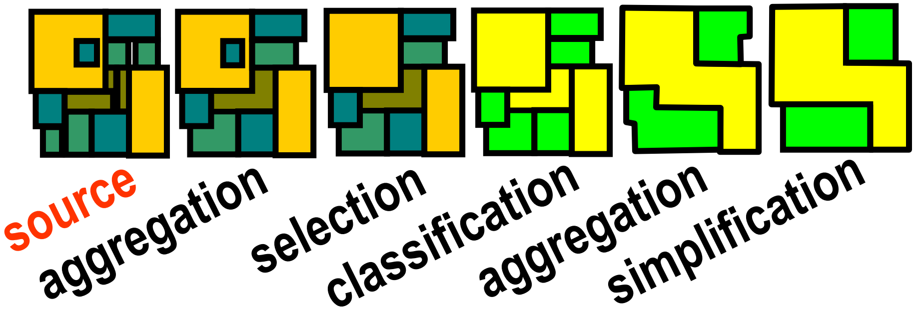
Often many generalization operations have to work iteratively to solve the conflicts

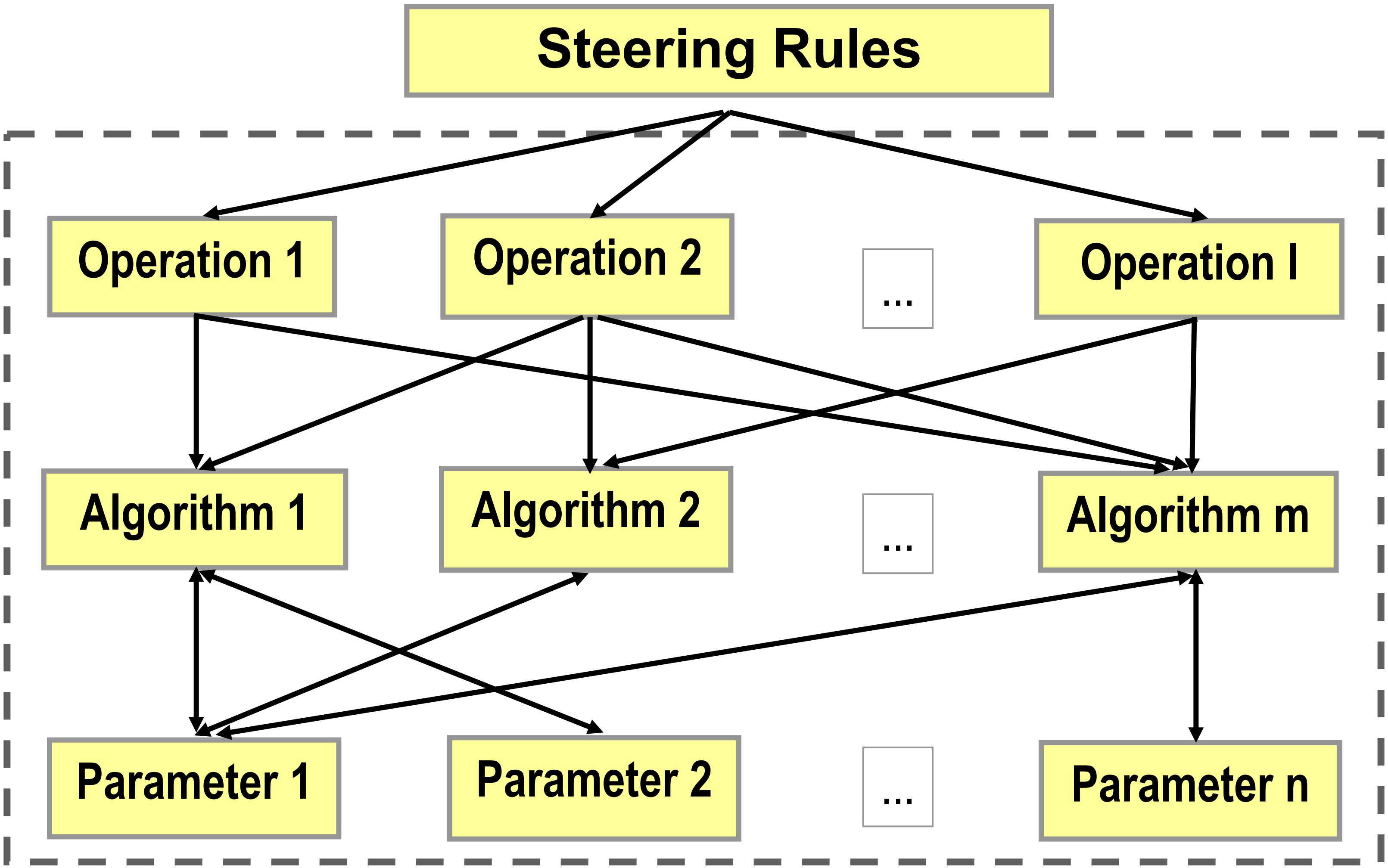
Example of a generalisation workflow

Dividing the generalization space into small chunks and working on chunks

Beginning with the most important feature types and using their generalization results as anchor points for other feature types

Beginning with operations that result in the reduced number of objects but no graphic conflicts

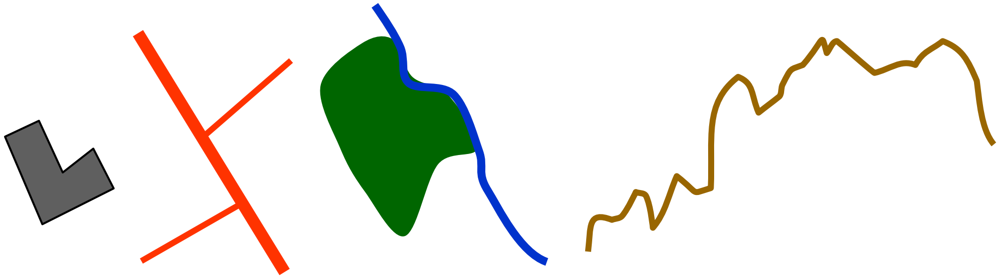



Steering Rules
Operation 1
Operation 2
...
Operation l
Algorithm 1
Algorithm 2
...
Algorithm m
Parameter 1
Parameter 2
...
Parameter n

Algorithms – *Töpfer*'s radical law

It decides how many objects (object parts) should be included in the derived map or database according to:

$$n_t = n_s k_i \sqrt{\frac{s_t}{s_s}}$$

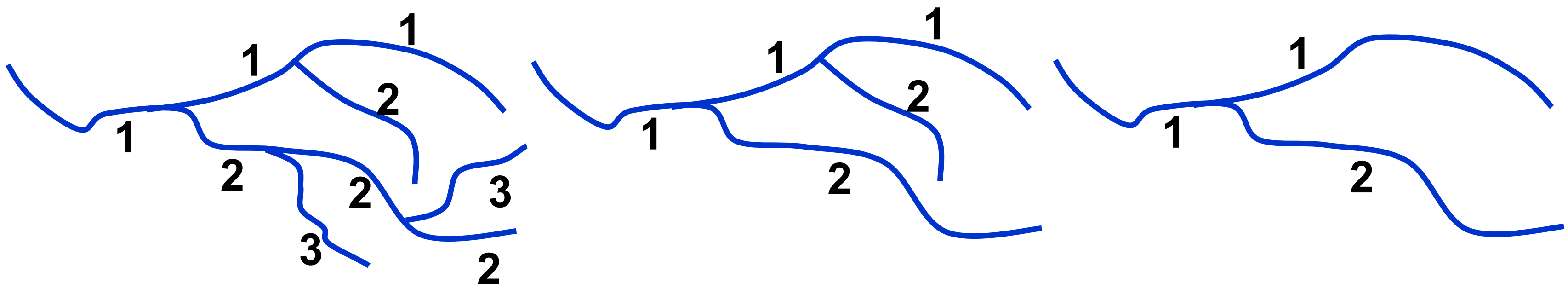


n_t the number of objects on the target map
n_s the number of objects on the source map
k_i weight for the relative importance of object type i ($i = 1, 2, 3, \ldots$)
s_t scale of the target map
s_s scale of the source map

Algorithms – *Horton*'s orders

It decides the relative importance of individual streams.

A downstream that intersects an upstream at the smallest angle will be identified as the continuation of the upstream and is assigned the same order class. The smaller the order number, the more important is the stream segment.



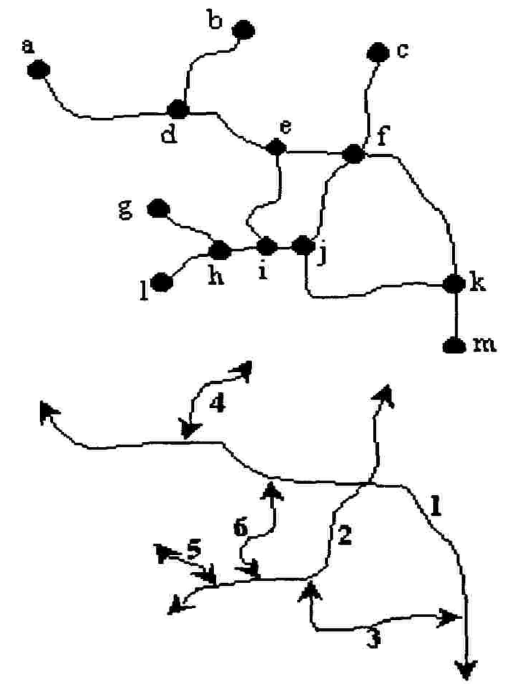
Successive elimination

Algorithms – perception-based orders

It decides the relative importance of line objects.

Line objects are partitioned into strokes according to the "good-continuity" principle. The strokes are then ranked based on criteria such as relative length, the number of line segments, or the number of intersections.

stroke	end nodes		number of line segments
1	a	m	5
2	c	l	5
3	j	k	1
4	b	d	1
5	g	h	1
6	e	i	1

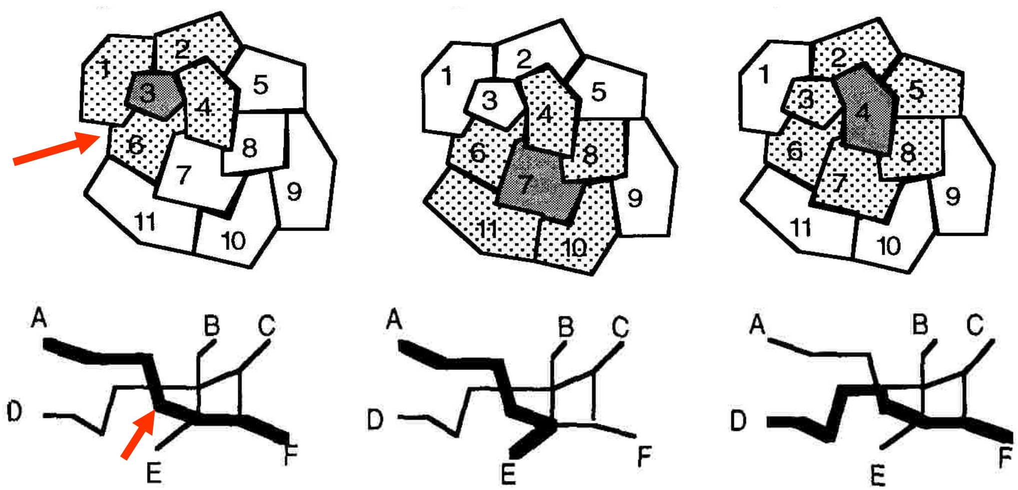


Algorithms - set-relationship orders

The number of associations an object belongs to reflects its relative importance.

“neighborhood” association

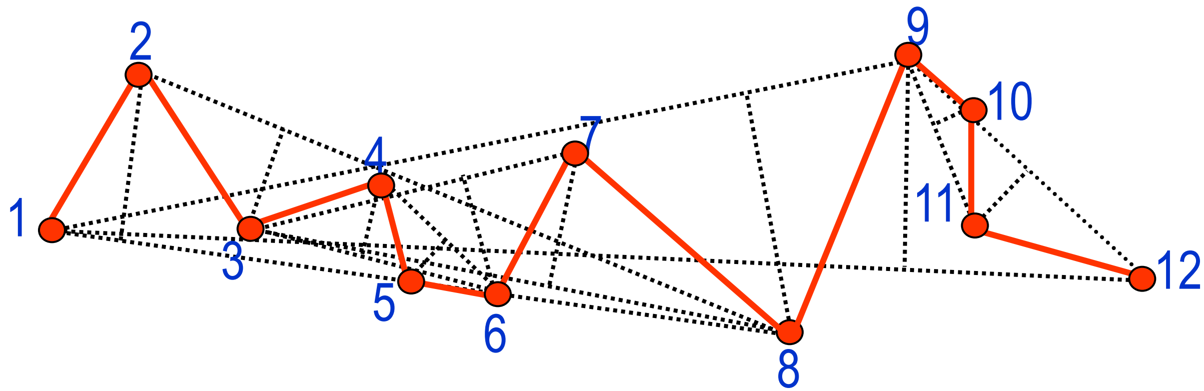
“route” association



Algorithms - *Douglas-Peucker*

It selects characteristic points on a line, *P* = tolerance length

1. define a straight line segment between the terminating points
2. measure the perpendiculars between the line segment and the intermediate points
3. split the line into two parts at the furthest point beyond *P*
4. repeat 1-3 on the two resulting parts until all perpendiculars fall within the tolerance



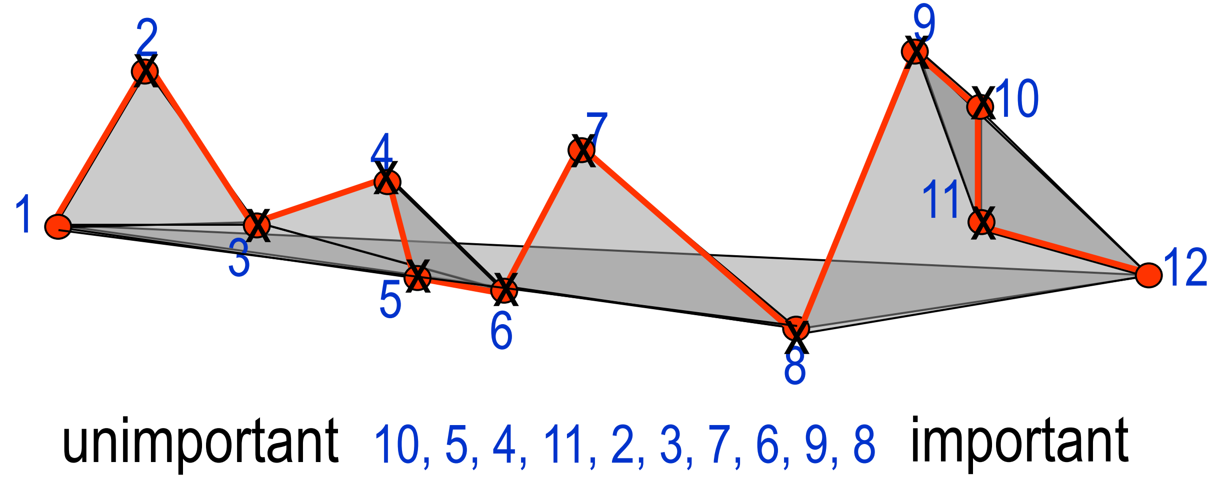
important 9, 8, 2, 3, 7, 6, 11, 4, 5, 10 unimportant

http://129.187.45.33/CartographyPlayground/playgrounds/douglas-peucker-algorithm/

Algorithms - *Visvalingham-Williamson*

It eliminates unimportant points on a line, *P* = tolerance area

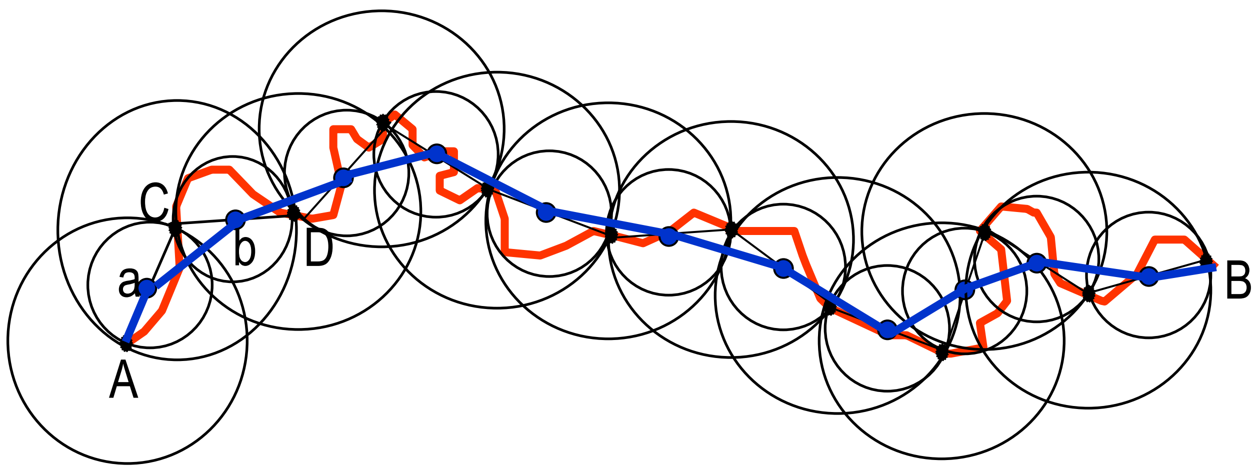
1. measure the relative importance of the intermediate points
2. eliminates the least important point, e.g. the point with the smallest area of the triangular feature (<*P*) formed by connecting the point with its two neighbors
3. repeat 1-2 until all remaining points are beyond the importance threshold



Algorithms - *Li-Openshaw*

It smoothens a line, *P* = tolerance length

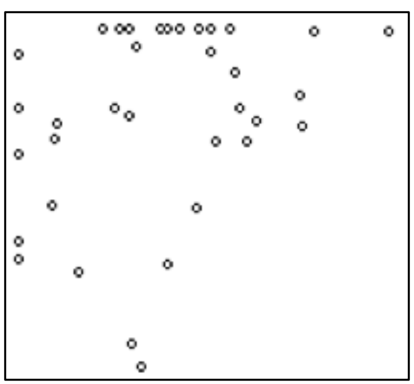
1. draw a circle at starting point A with diameter double so much as *P*, which intersects the point C on the line
2. draw a circle with AC as diameter. The center 'a' along the straight line section AC is selected as a point along the generalized line
3. set C as the new starting point and repeat 1-2

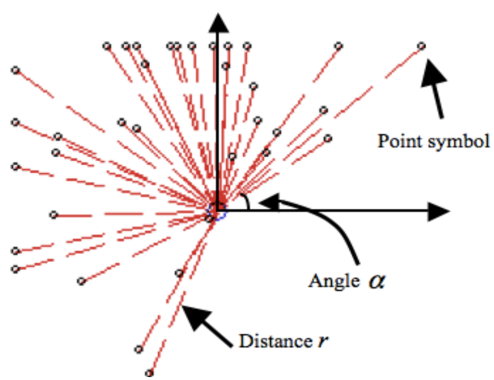


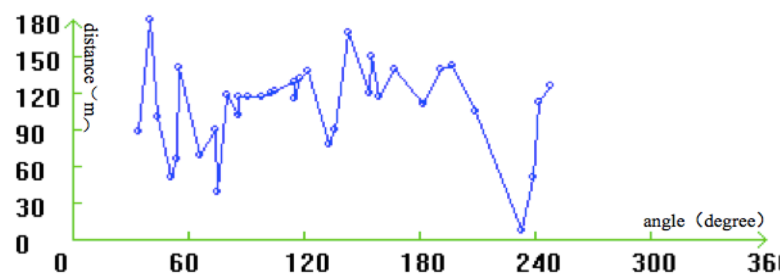
Algorithms – *Qian-Meng*

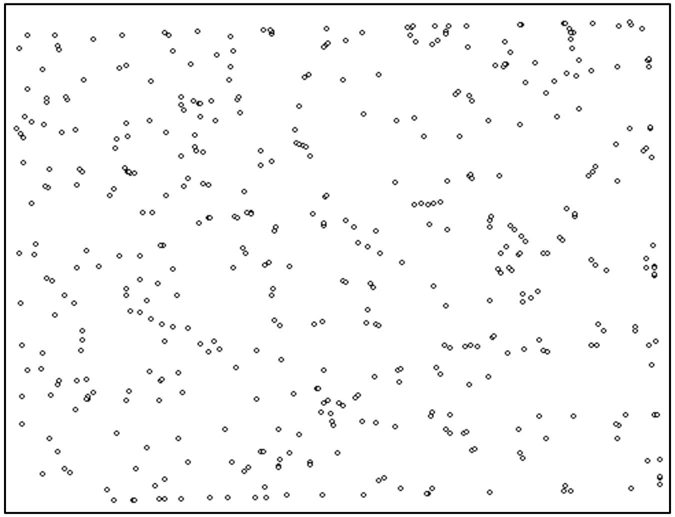
It simplifies a point cluster with a distance tolerance Δr and an angle tolerance $\Delta\alpha$

1. determine the polar center p_c of a cluster, which has the minimum average distance to all other points
2. determine the polar radius r_i and the polar angle α_i of each point $p_{i,}$ $i = 1, 2, \ldots n-1$ in the cluster to p_c
3. convert the cluster into a polar line ranging from $1°$ to $360°$,
4. delete the points on the polar line that fall below Δr , $\Delta\alpha$ and convert it back to a simplified cluster

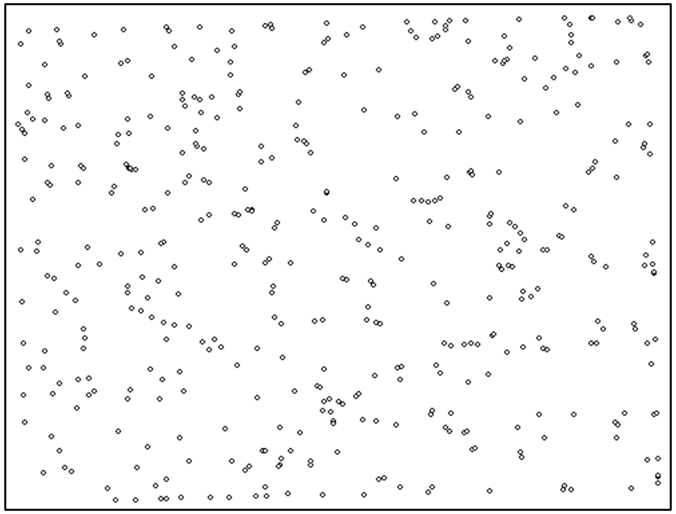




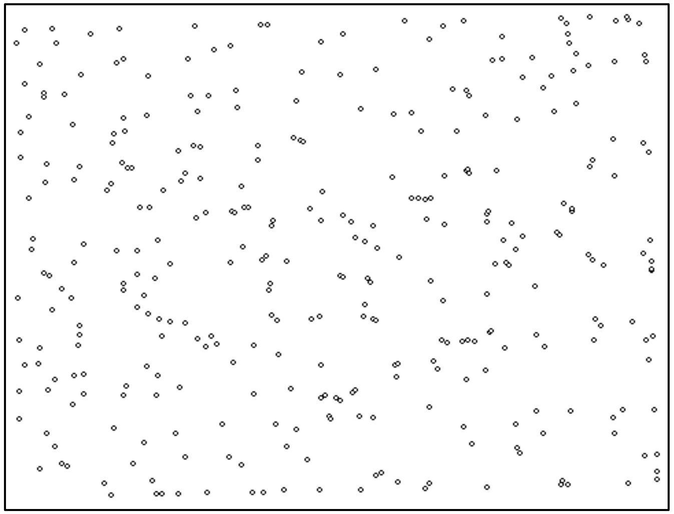




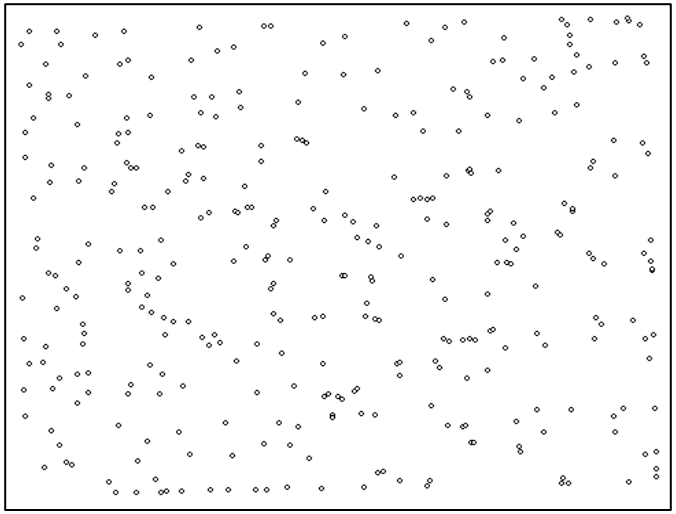
n = 461



$\Delta\alpha, \Delta r = (10,10)$ 437



$\Delta\alpha, \Delta r = (10,200)$ 332



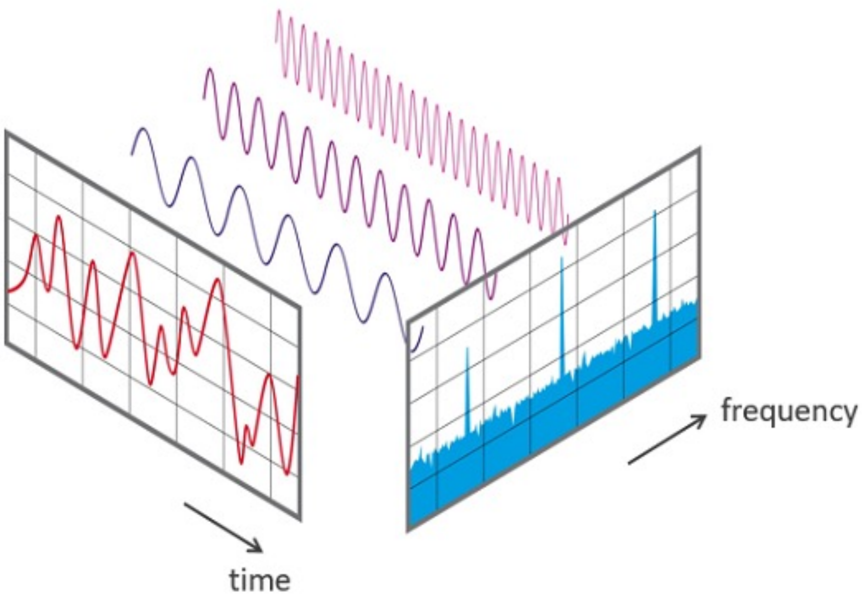
$\Delta\alpha, \Delta r = (20,100)$ 337

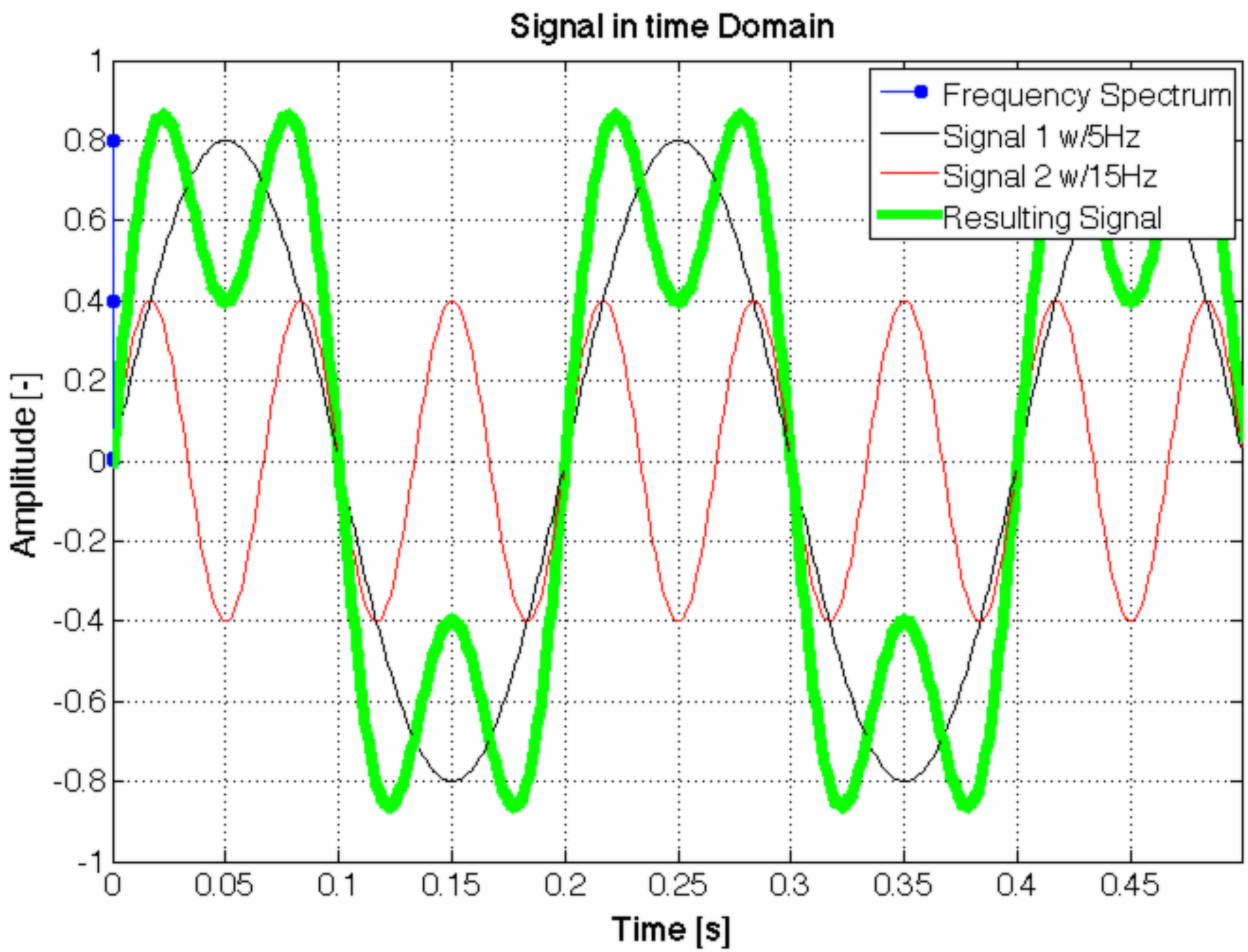
Algorithms – Fourier Transform

It smoothens a spectrum line globally, *P* = the number of Fourier coefficient pairs

Any spectrum can be approximated by the summation of a series of sine and cosine waves. The determination of sine and cosine components of a given spectrum is known as FT.

FT converts the time domain to frequency domain. The useful information is found at lower frequencies and the higher frequencies can be ignored.




Signal in time Domain
Frequency Spectrum
Signal 1 w/5Hz
Signal 2 w/15Hz
Resulting Signal
Amplitude [-]
Time [s]
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45

If a spectrum is recorded at *n* equally spaced wavelength intervals, $\gamma(1)$, $\gamma(2)$, $\gamma(3)$,…, $\gamma(n)$, then it can be approximated by:

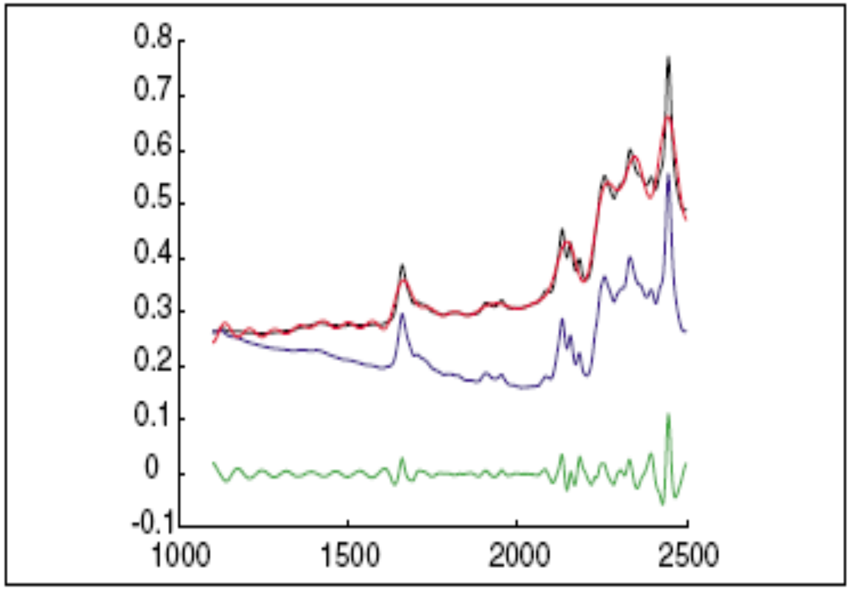
$$\hat{\gamma}(h) = a_0 + \sum_{j=1}^{k} a_j \cos(\frac{2\pi jh)}{n}) + \sum_{j=1}^{k-1} b_j \sin(\frac{2\pi jh)}{n})$$

$$a_0 = \frac{1}{n}\sum_{h=1}^{n}\gamma(h)$$

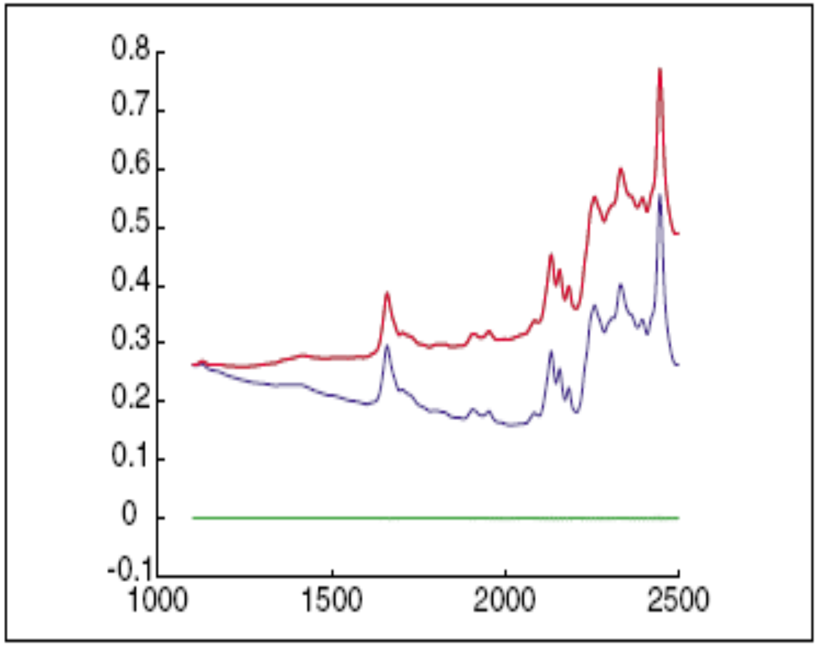
$$a_j = \frac{2}{n}\sum_{h=1}^{n}\gamma(h)\cos(\frac{2\pi jh}{n})$$ for j = 1,2,3,…, k-1

$$a_k = \frac{1}{n}\sum_{h=1}^{n}\gamma(h)\cos\pi h$$

$$b_j = \frac{2}{n}\sum_{h=1}^{n}\gamma(h)\sin(\frac{2\pi jh)}{n})$$ for j = 1,2,3,…, k-1



The original curve (black), reconstructed (red) using 20 pairs of Fourier coefficients, tilted (blue), difference (green)



The reconstructed (red) using 100 pairs of Fourier coefficients, tilted (blue), difference (green)

Algorithms - morphological transformations

Dilation $A \oplus B = \bigcup_{b \in B} A_b$

Erosion $A \Theta B = \bigcap_{b \in B} A_b$

A

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	0	0	0	0	1	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	1	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0
0	0	1	0	0	0	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0
0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$$B = \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

dilation of *A* by *B*

0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	1	1	0	1	1	1	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0

Blue "1" pixels form the expanded area

erosion of *A* by *B*

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Blue "0" pixels form the eroded area

A

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	0	0	0	0	1	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	1	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0
0	0	1	0	0	0	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0
0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

C = dilation of *A* by *B*

0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	1	1	0	1	1	1	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0

D= erosion of *C* by *B*

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	0	0	0	1	0	0
0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0
0	0	1	0	0	0	1	1	1	1	1	1	1	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0
0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$$B = \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

First dilation, then erosion results in amalgamation, boundary simplification and elimination of small details.

A

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	0	0	0	0	1	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	1	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0
0	0	1	0	0	0	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0
0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

C = erosion of *A* by *B*

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

D= dilation of *C* by *B*

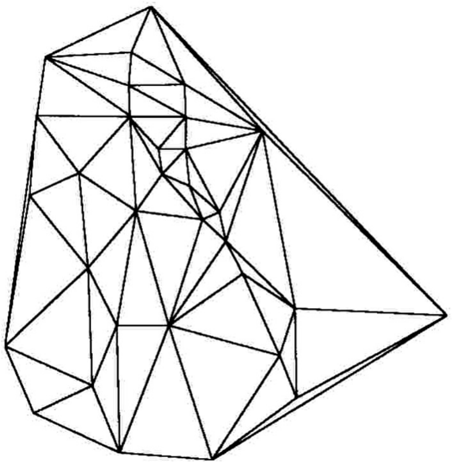
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$$B = \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

First erosion, then dilation results in boundary simplification, elimination of small details and bridges.

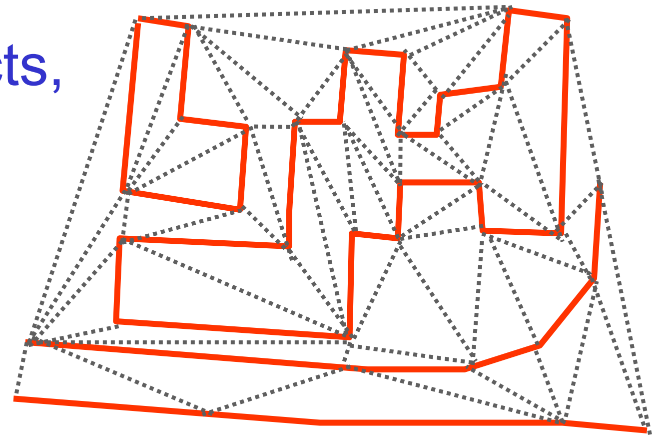
Algorithms - Graph-based data structures

Delaunay Triangulation (DT)



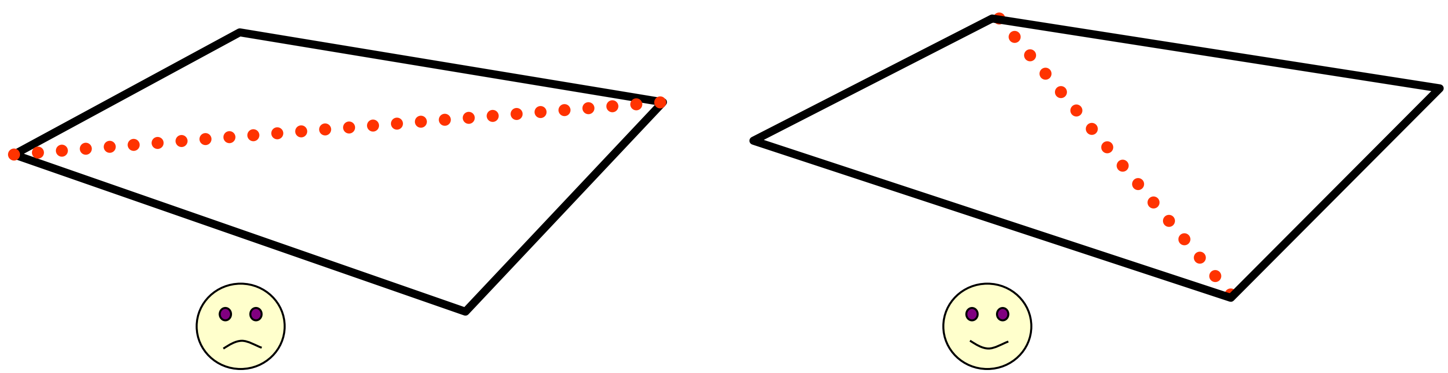
Constraint Delaunay Triangulation (CDT)

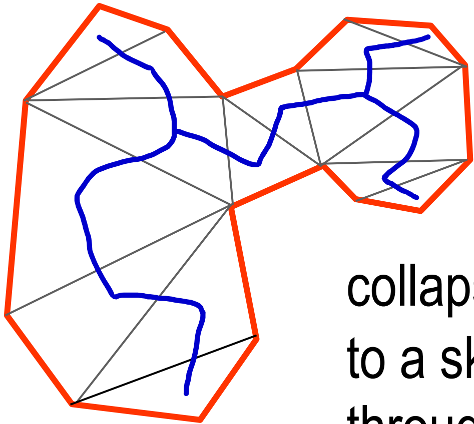
It is a modification of a Delaunay triangulation, in which edges, belonging to polygon and line objects, are enforced as edges within the triangulation.



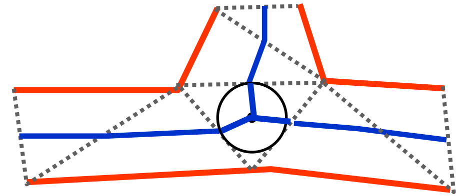
Quality criteria for Delaunay triangulation

- The edges of triangles should have similar length.
- The angle difference of each triangle should be kept as small as possible.
- The area difference between the neighboring triangles should be kept as small as possible.

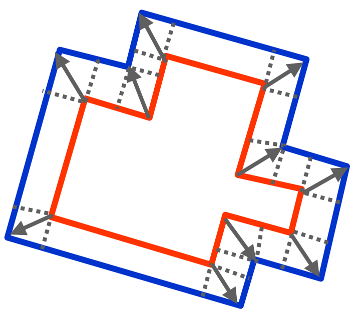




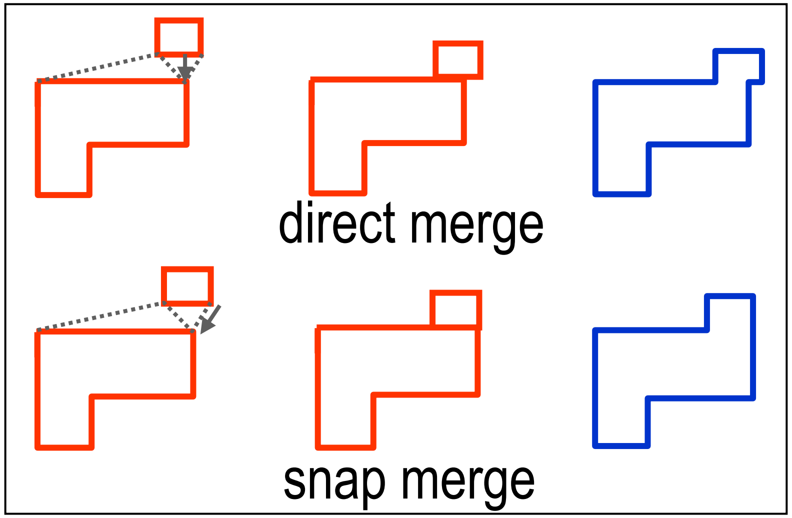
collapse of a polygon to a skeleton, e.g. through centers of triangle circumcircles

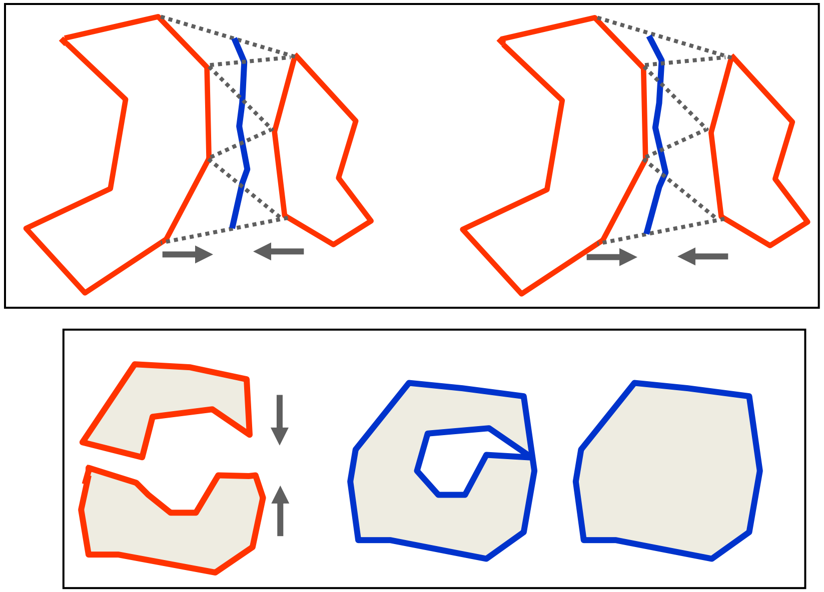


collapse of a ribbon-shaped object to a skeleton, e.g. through the centers of triangle edges connecting opposite sides

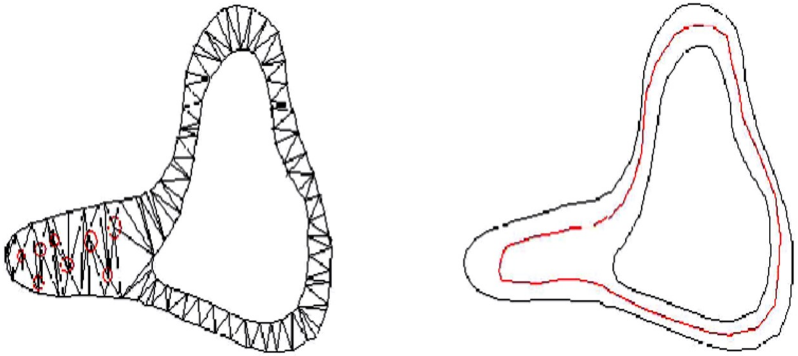


graphic exaggeration of a concave polygon based on displacement vector of triangle nodes

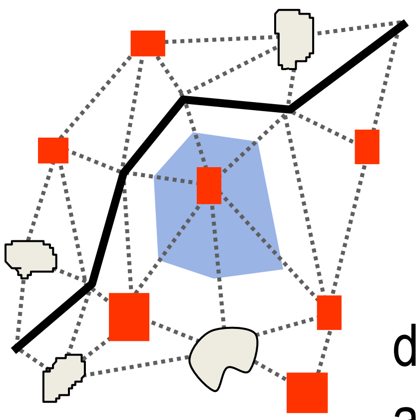




middle axis and weighted middle axis as merging boundary



Derivation of a new contour line based on CDT of existing contour lines and terrain points



displacement buffer around an arbitrary object

Questions

1. Which differences can two data sources of the same region reveal?
2. Why do we need to integrate different data sources?
3. Describe the general workflow of geodata integration.
4. How can you define the similarity between objects to be matched?
5. How can you determine the tolerance threshold for similar objects?
6. Give some examples for the applications of data integration.
7. Why do we have to generalize maps?
8. Which automatic generalization operations do you know and how do they work?
9. What is the Töpfer's radical law?
10. Which methods of ranking the relative importance of individual objects do you know and how do they work?
11. Which automatic algorithms of line simplification do you know and how do they work?
12. Explain the working principle of Fourier Transform for line generalization.
13. How can the morphological transformations be used as generalization means?
14. How can the graph-based data structures be used to generalize maps?