19th International Conference on
Computing in Civil & Building Engineering
CAPE TOWN 26 - 28 OCTOBER 2022

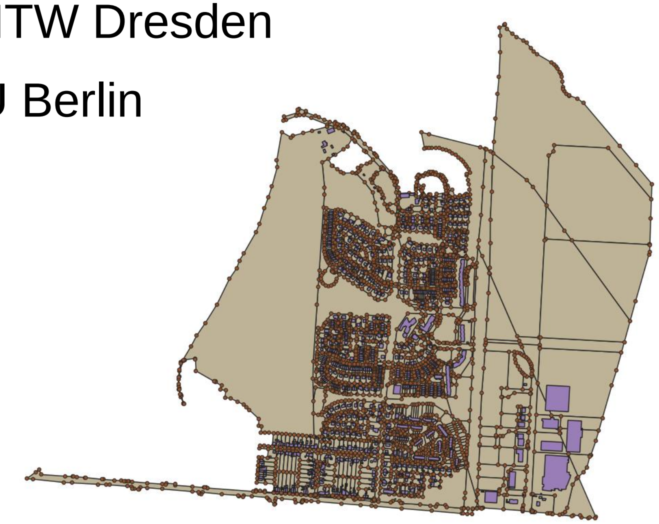
HOCHSCHULE FÜR TECHNIK UND WIRTSCHAFT DRESDEN
UNIVERSITY OF APPLIED SCIENCES

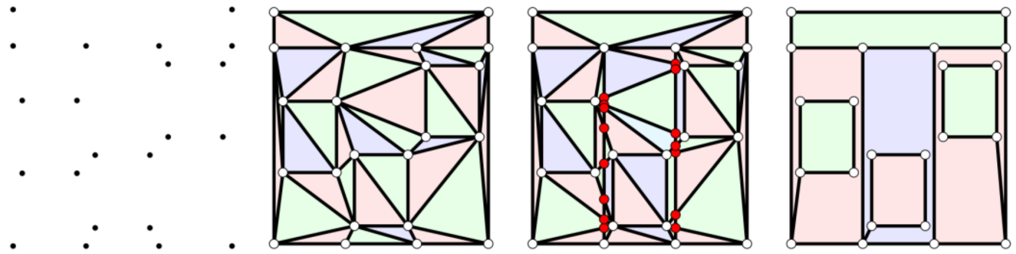
A computational robust method for spatial decomposition - Test case with cadastral data

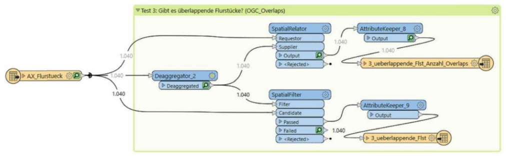
Enrico Romanschek, HTW Dresden

Christian Clemen, HTW Dresden

Wolfgang Huhnt, TU Berlin

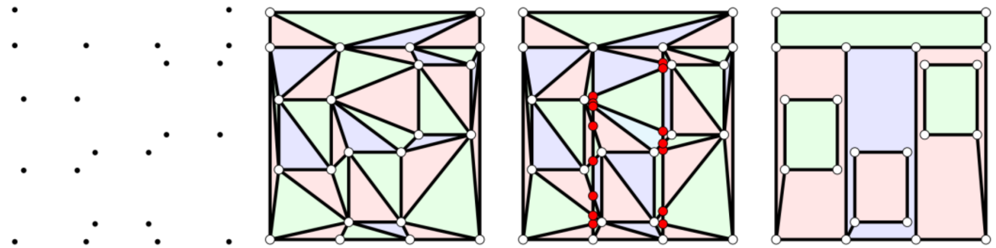


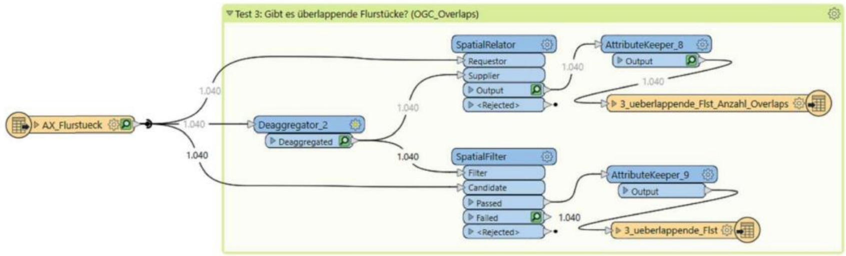




Agenda

- 1. Standard (GIS) vs. space decomposition.
- 2. Short explanation how our algorithm works.
- 3. Test cases with results and cross-check.
- 4. Conclusion and Outlook.





Comparison

Standard

- Simple features (B-Rep) possibly with attached semantic data.

Our Algorithm

Comparison

Standard

- Simple features (B-Rep) possibly with attached semantic data.
- Simple features without topology.
- Only features modeled.
- Coordinates as floating-point numbers.
- Intersections as computed coordinates with possible loss of precision.

Our Algorithm

Comparison

Standard

- Simple features (B-Rep) possibly with attached semantic data.

- Simple features without topology.
- Only features modeled.
- Coordinates as floating-point numbers.
- Intersections as computed coordinates with possible loss of precision.

Our Algorithm

- Complete, gapless and overlap-free two-dimensional space decomposition.

Comparison

Standard

- Simple features (B-Rep) possibly with attached semantic data.
- Simple features without topology.
- Only features modeled.
- Coordinates as floating-point numbers.
- Intersections as computed coordinates with possible loss of precision.

Our Algorithm

- Complete, gapless and overlap-free two-dimensional space decomposition.
- A data structure that maps the topology.
- Modeling even empty spaces as objects.
- Exclusive use of integer coordinates.
- Intersections as positions on edges and not by coordinates.
- A special algorithm for reconstructing.

Comparison

Standard

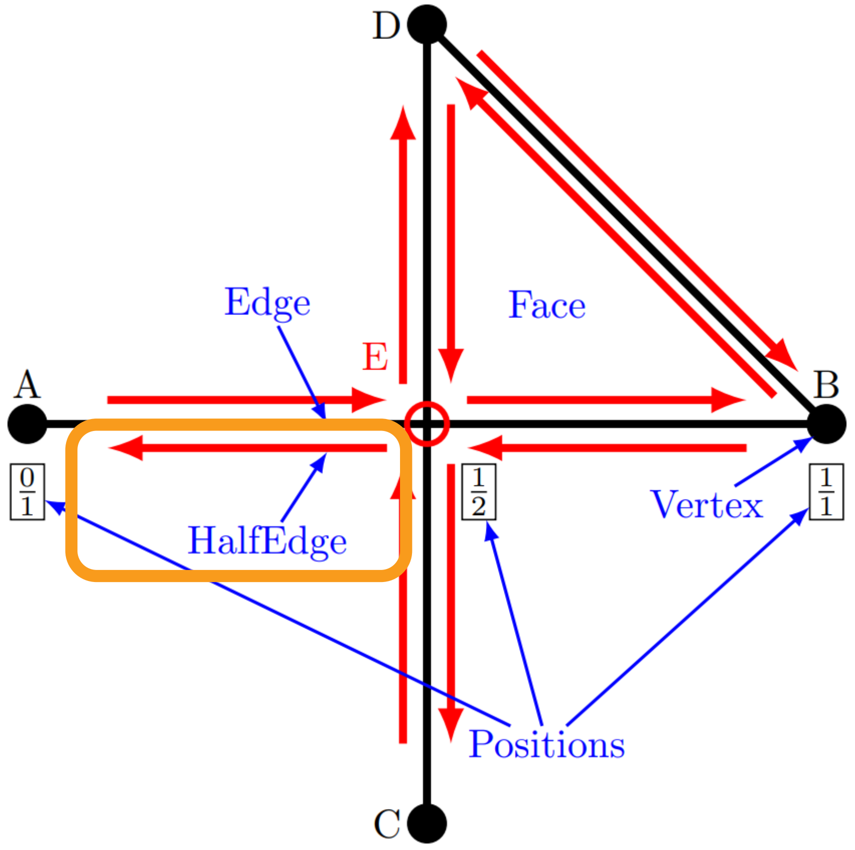
- Simple features (B-Rep) possibly with attached semantic data.
- Simple features without topology.
- Only features modeled.
- Coordinates as floating-point numbers.
- Intersections as computed coordinates with possible loss of precision.
- Spatial relations with error-prone numerical calculations.

Our Algorithm

- Complete, gapless and overlap-free two-dimensional space decomposition.
- A data structure that maps the topology.
- Modeling even empty spaces as objects.
- Exclusive use of integer coordinates.
- Intersections as positions on edges and not by coordinates.
- A special algorithm for reconstructing.
- Spatial relations with DE-9IM matrices as set operations.

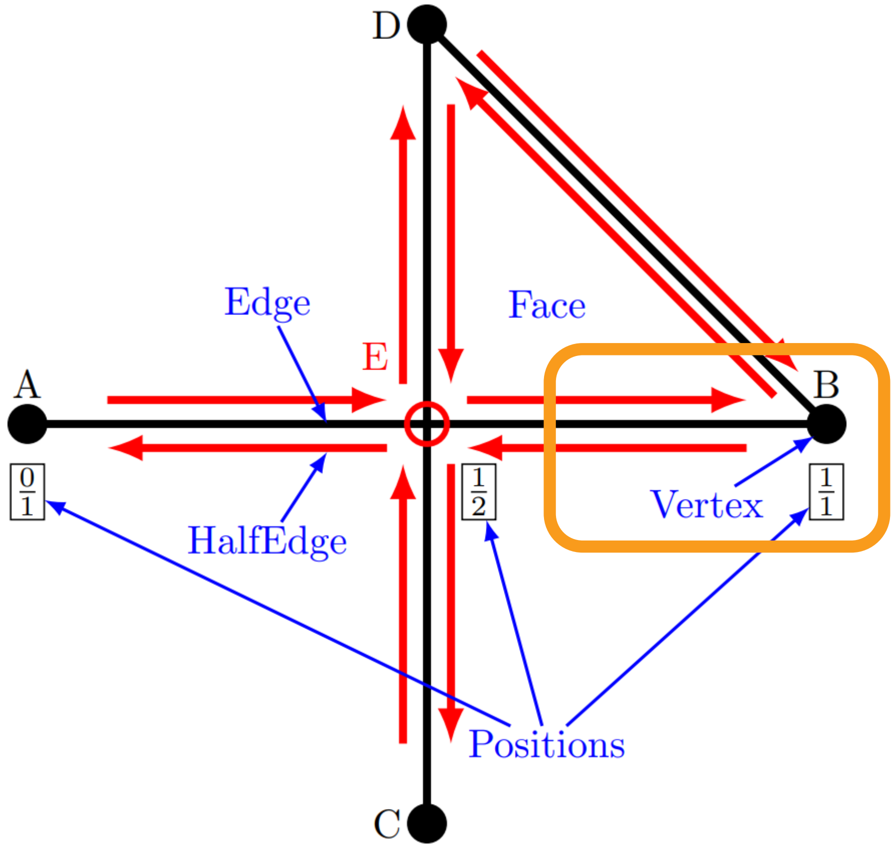
Procedure – data structure

Half-edges



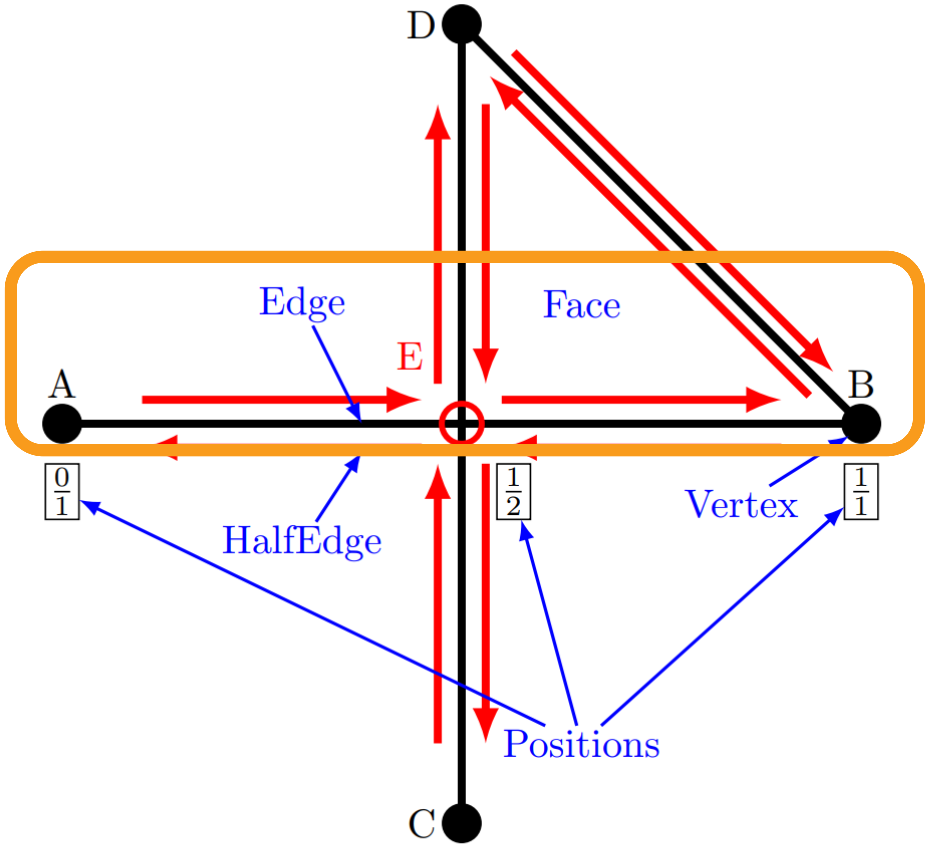
Procedure – data structure

- Half-edges
- Vertex



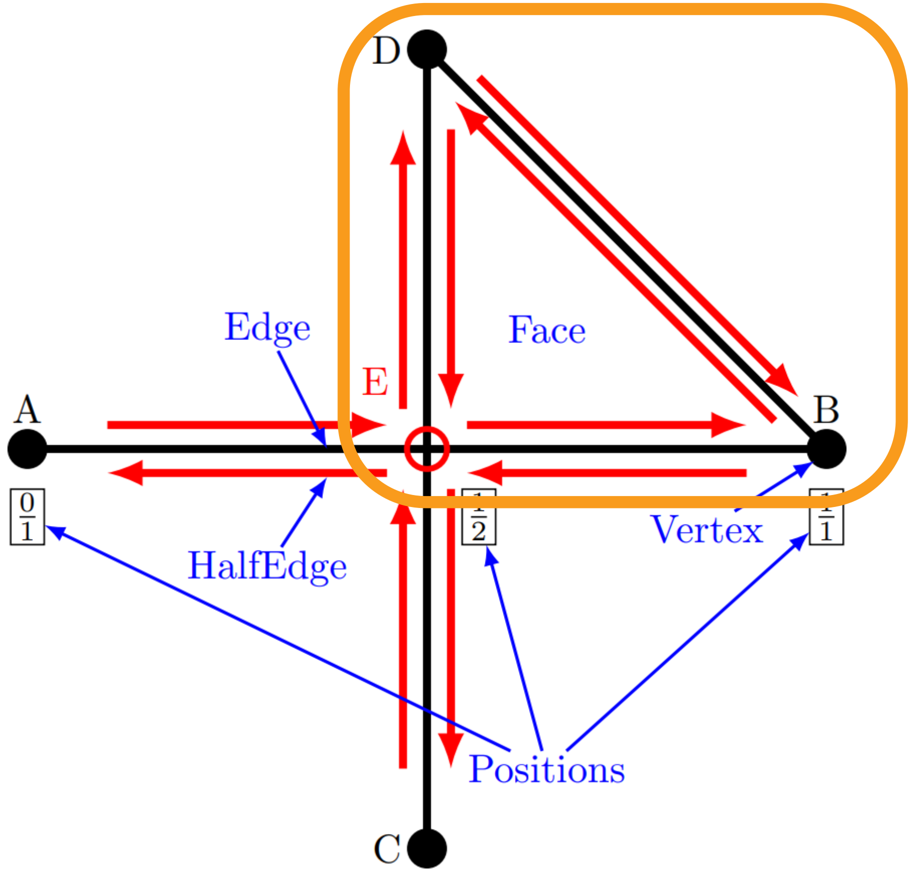
Procedure – data structure

- Half-edges
- Vertex, Edge



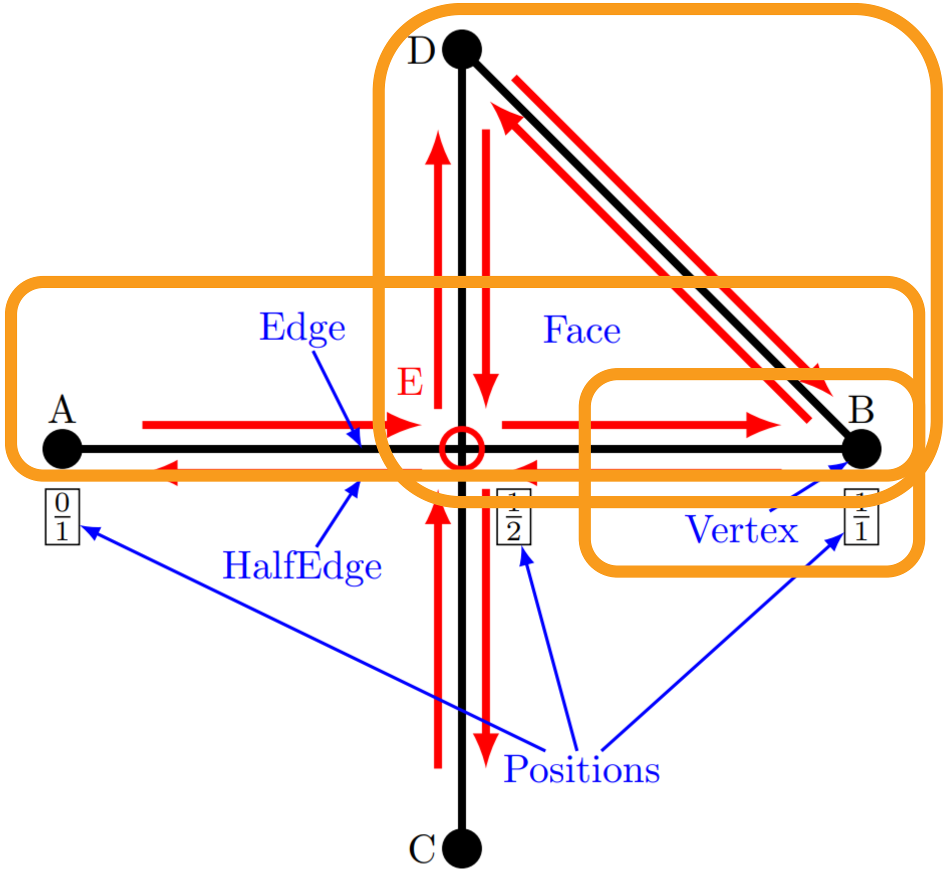
Procedure – data structure

- Half-edges
- Vertex, Edge, Face



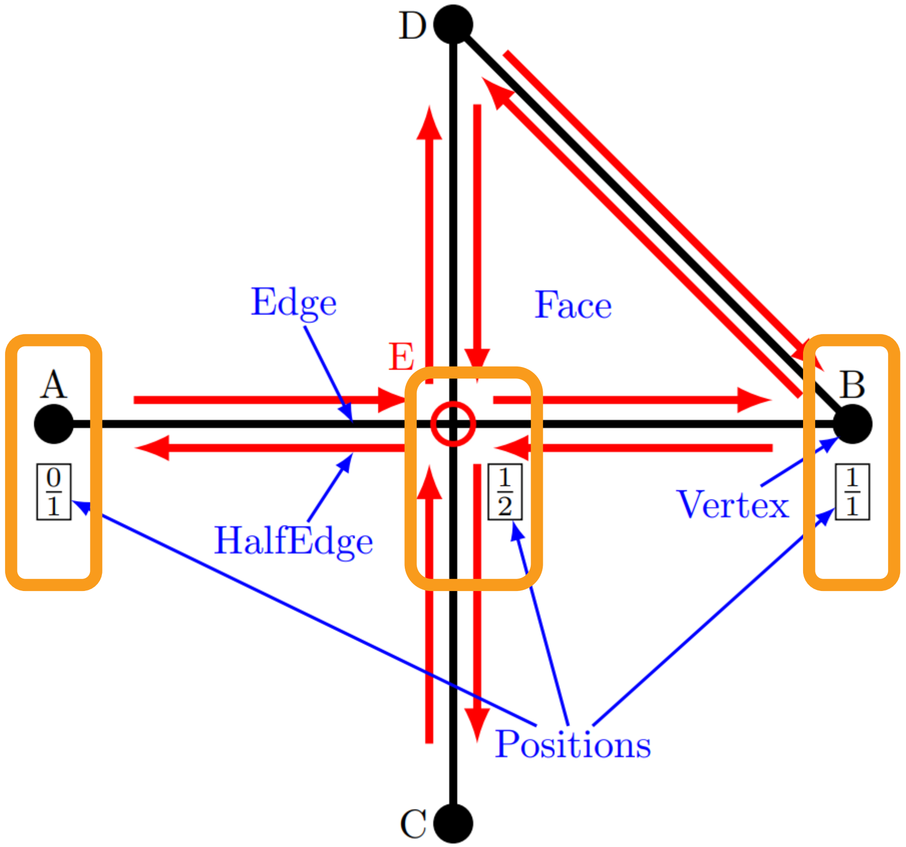
Procedure – data structure

- Half-edges
- Vertex, Edge, Face
- with Ids of the origin data



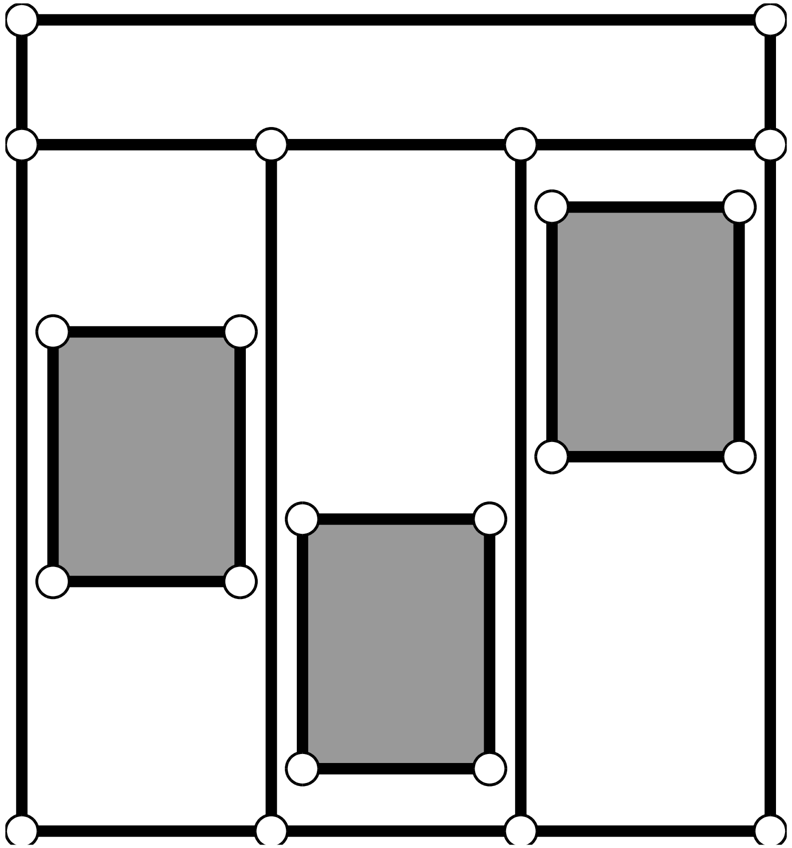
Procedure – data structure

- Half-edges
- Vertex, Edge, Face
- with Ids of the origin data
- Vertices as positions on edges



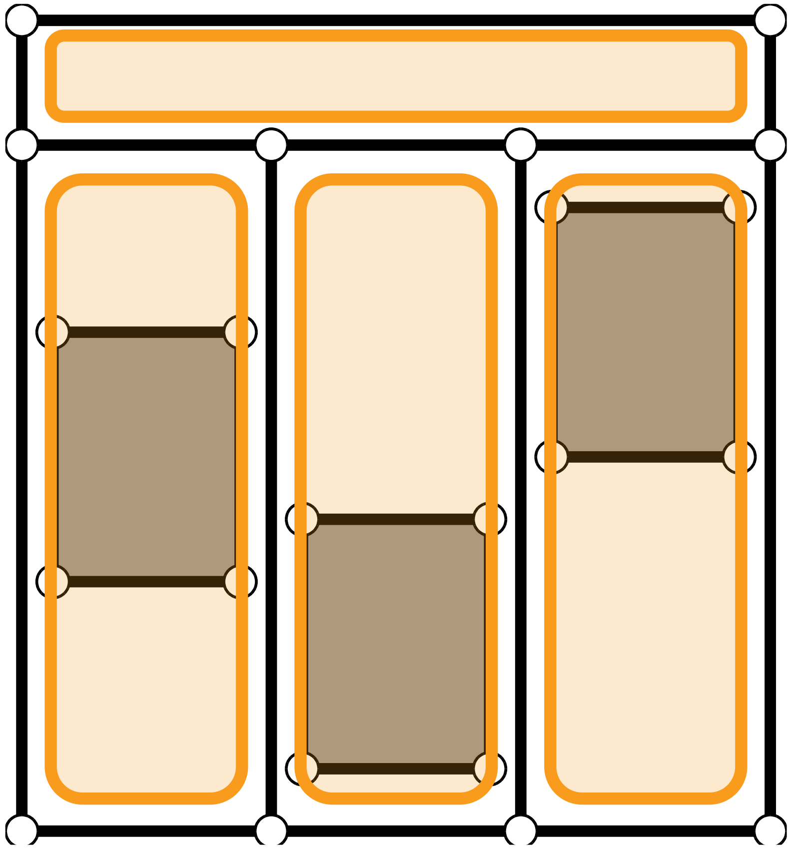
Procedure – initial data

- 7 Polygons



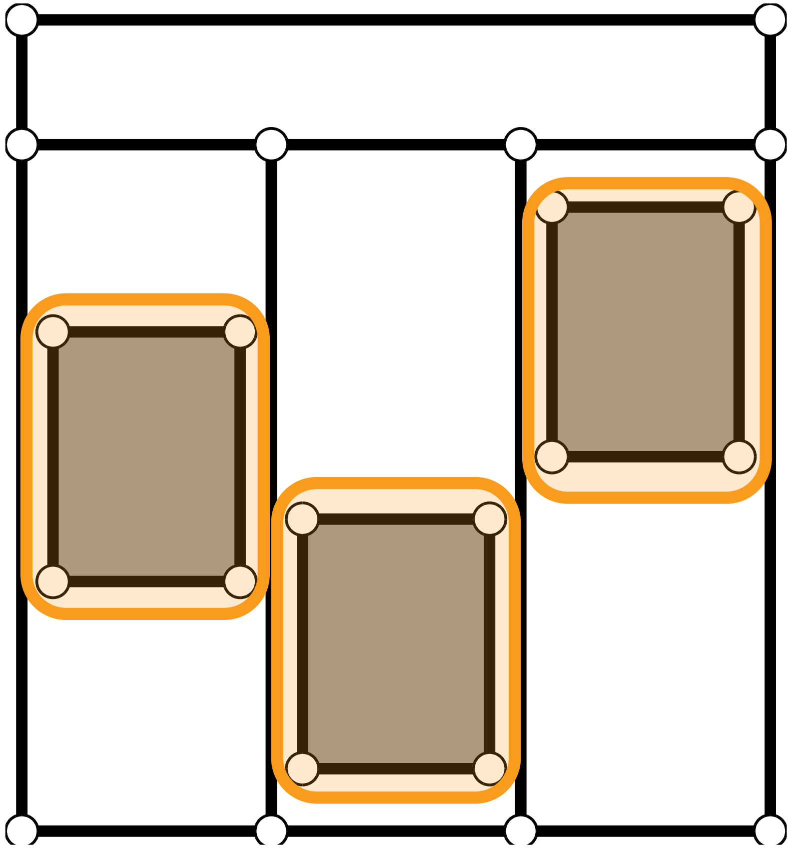
Procedure – initial data

- 7 Polygons
  - 4 Parcels

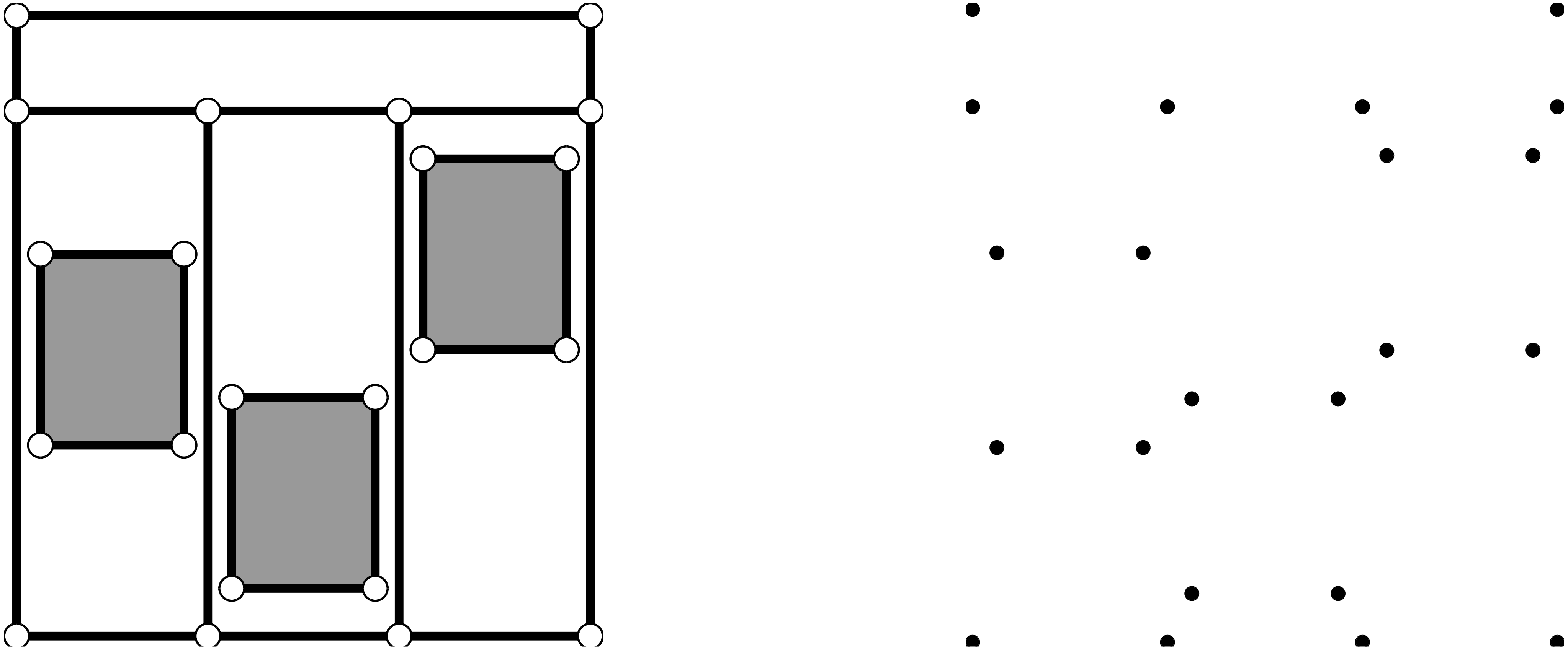


Procedure – initial data

- 7 Polygons
  - 4 Parcels
  - 3 Buildings



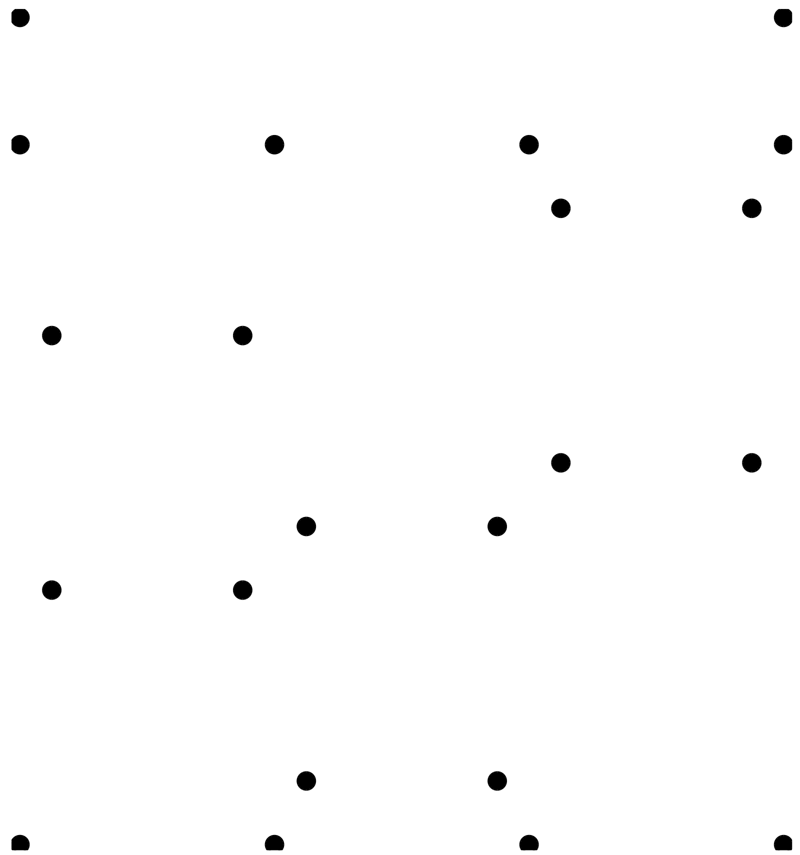
Procedure – step 1 – integer coordinates



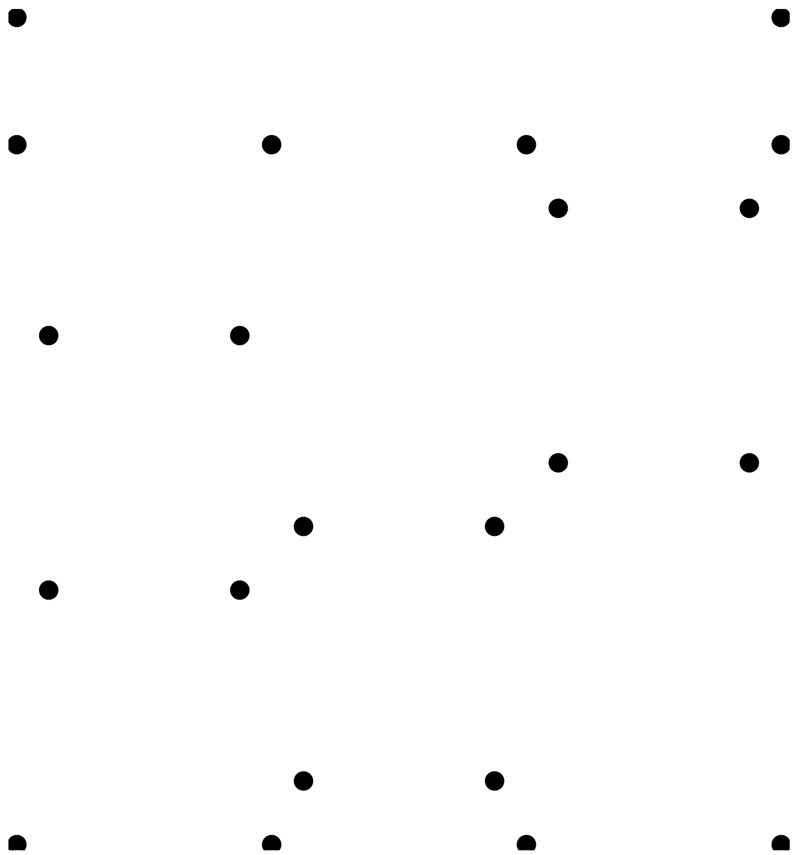
Procedure – step 1 – integer coordinates

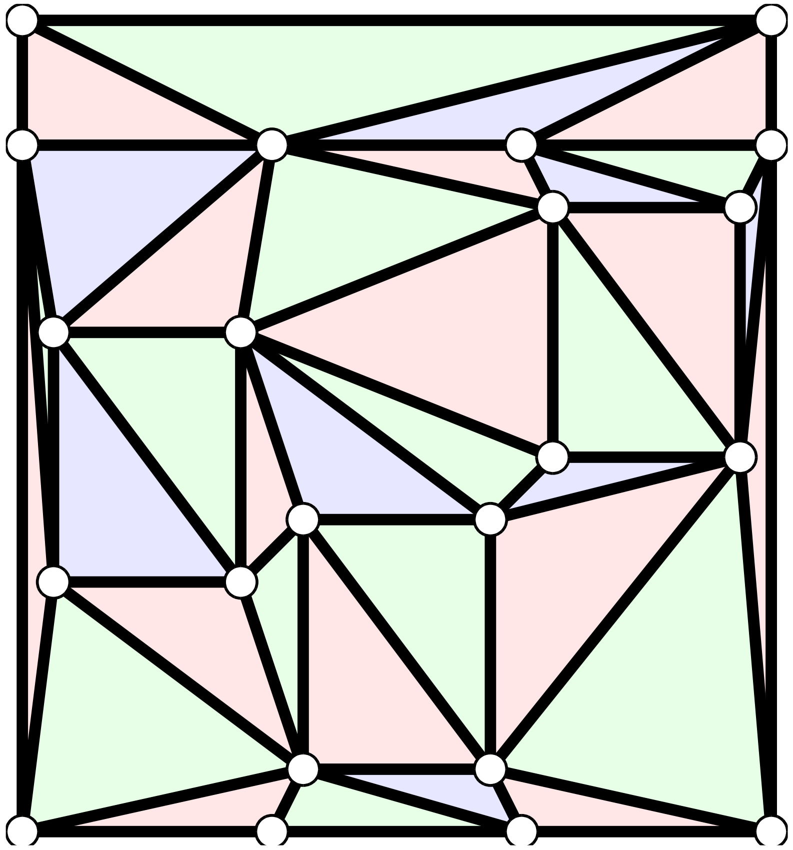
- 22 points
  - Conversion from floating point to integer

$$\begin{pmatrix} x_{int} \\ y_{int} \end{pmatrix} = scale \left[\begin{pmatrix} x_{float} \\ y_{float} \end{pmatrix} - \begin{pmatrix} x_{min} \\ y_{min} \end{pmatrix} \right]$$



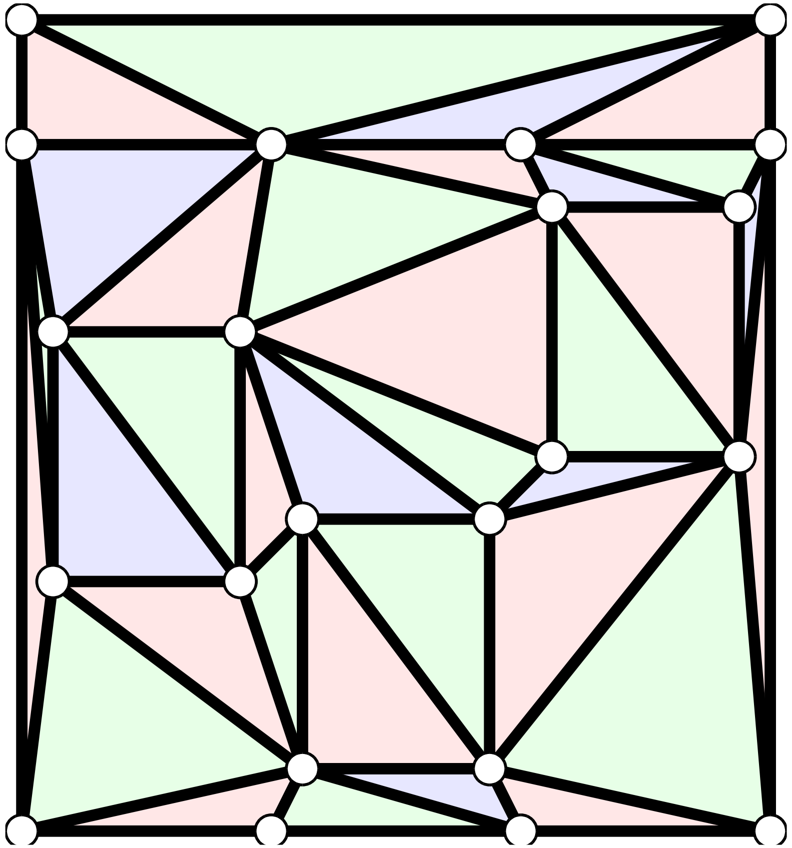
Procedure – step 2 – triangulation





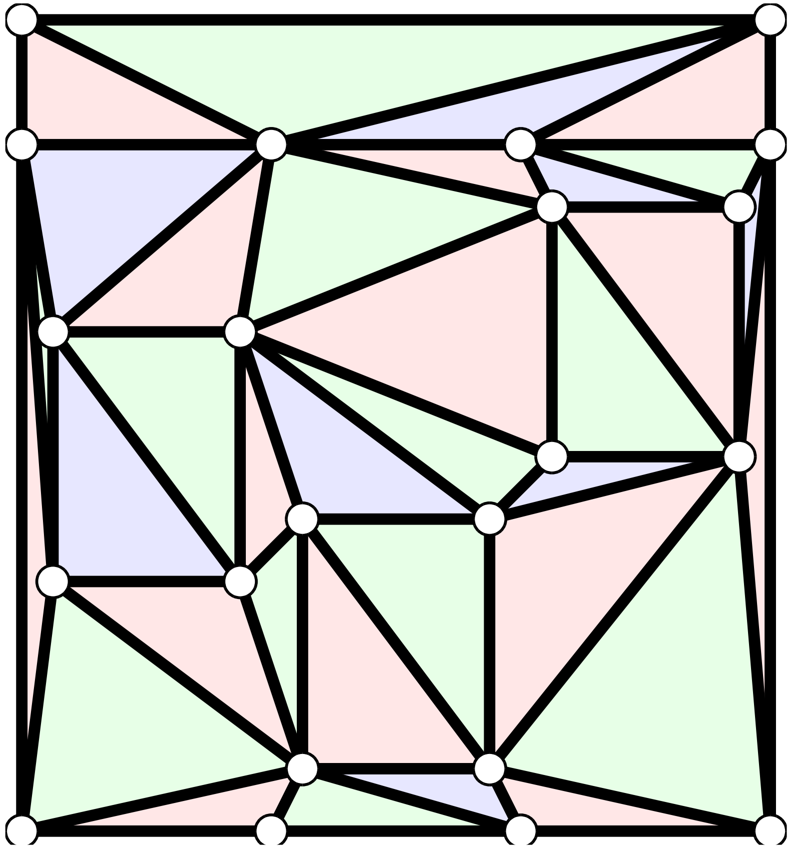
Procedure – step 2 – triangulation

- Delaunay triangulation

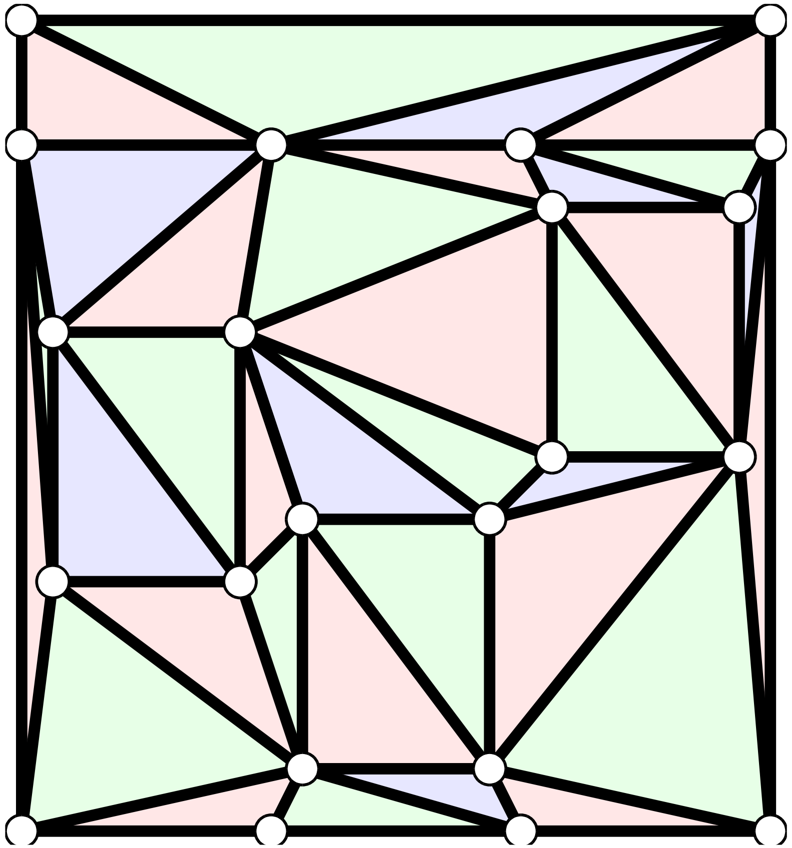


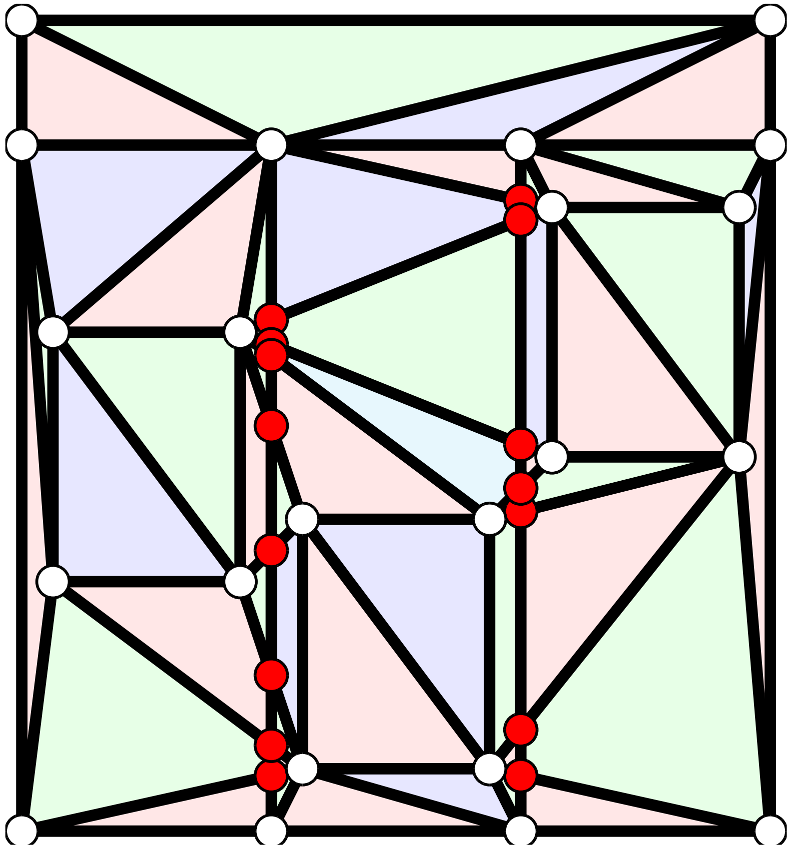
Procedure – step 2 – triangulation

- Delaunay triangulation
  - 34 triangles



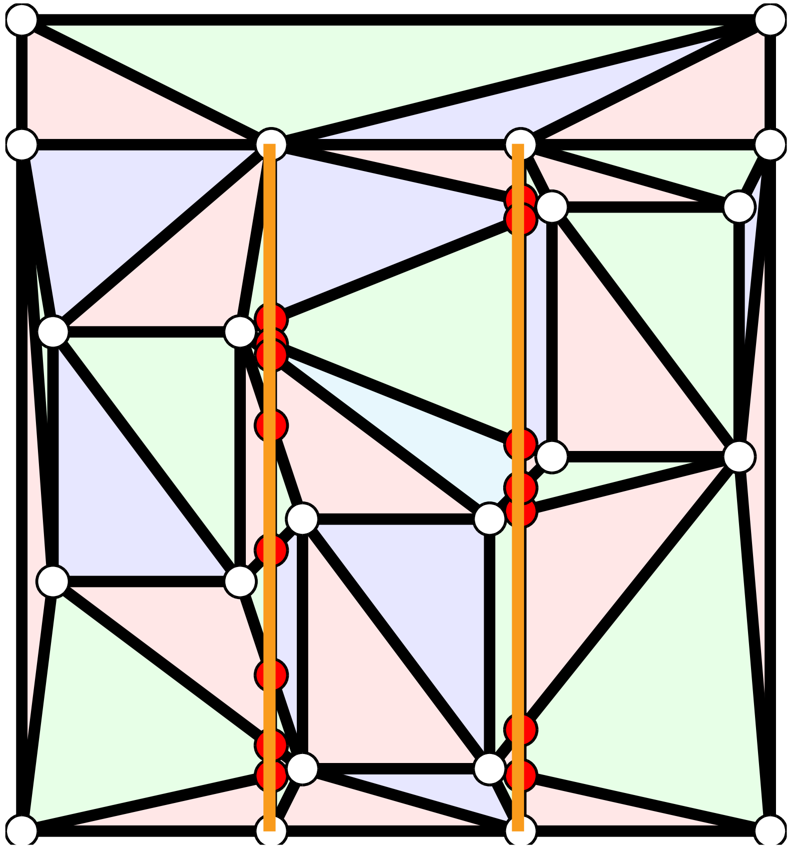
Procedure – step 3 – reconstruction





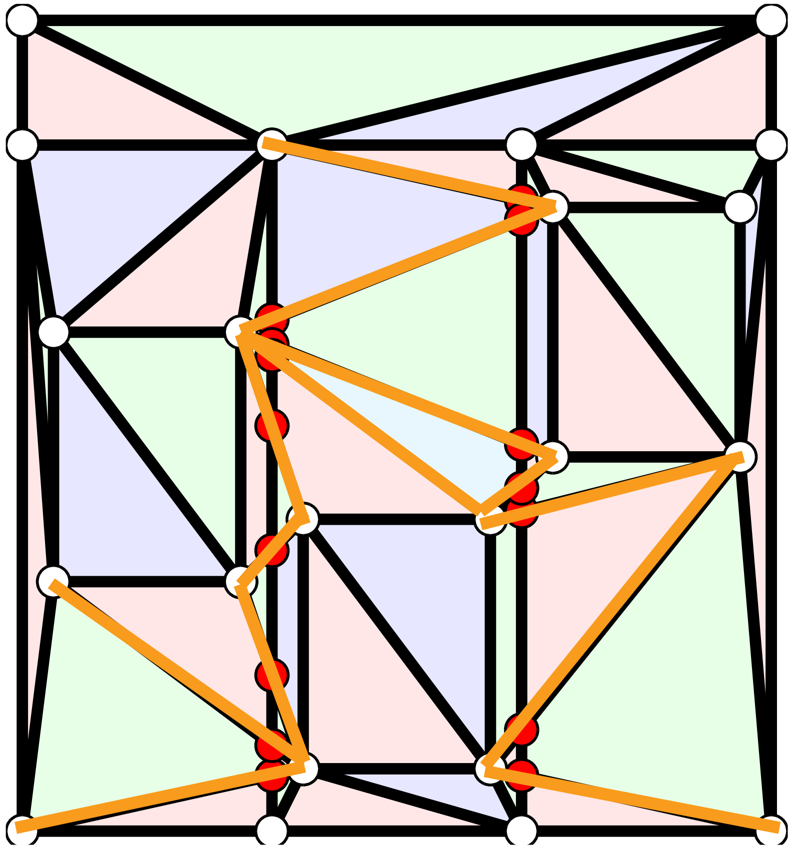
Procedure – step 3 – reconstruction

- Reconstruction of the remaining edges



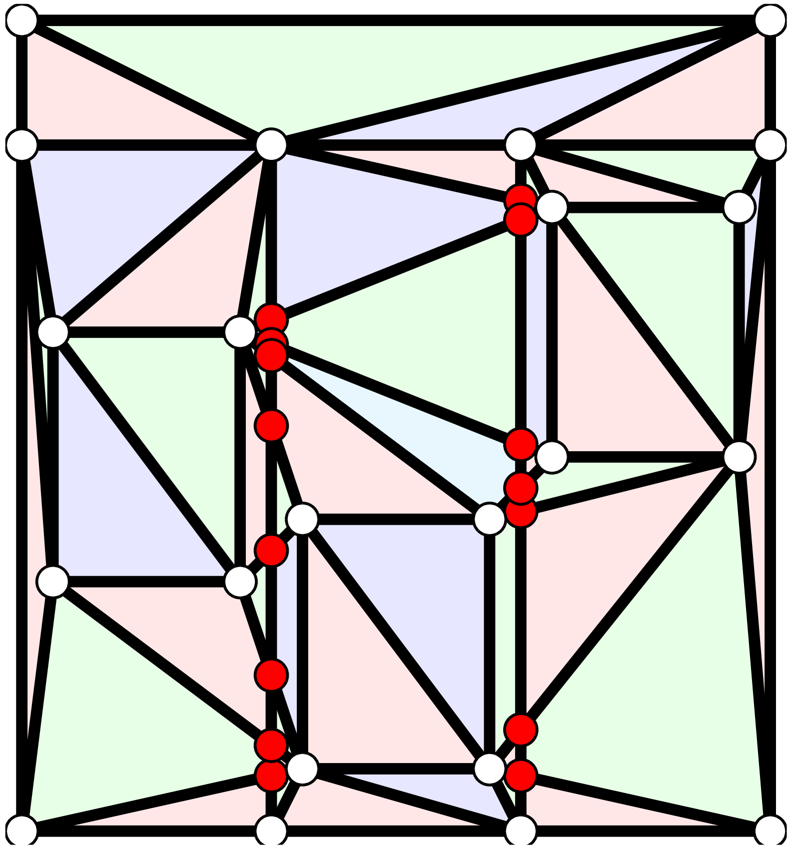
Procedure – step 3 – reconstruction

- Reconstruction of the remaining edges
  - Search for edges to be intersected

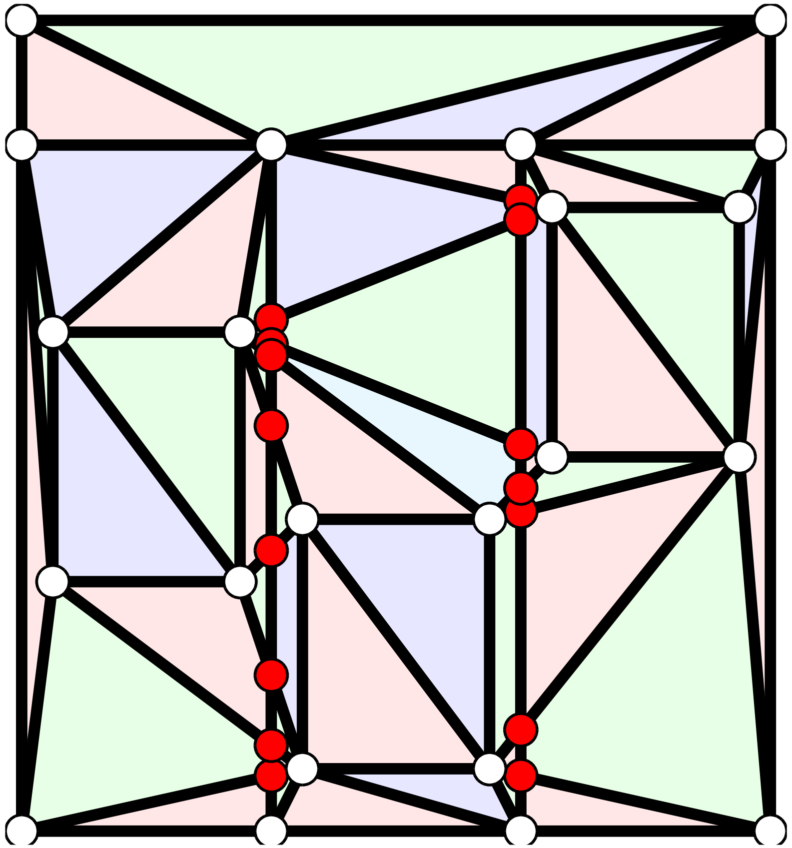


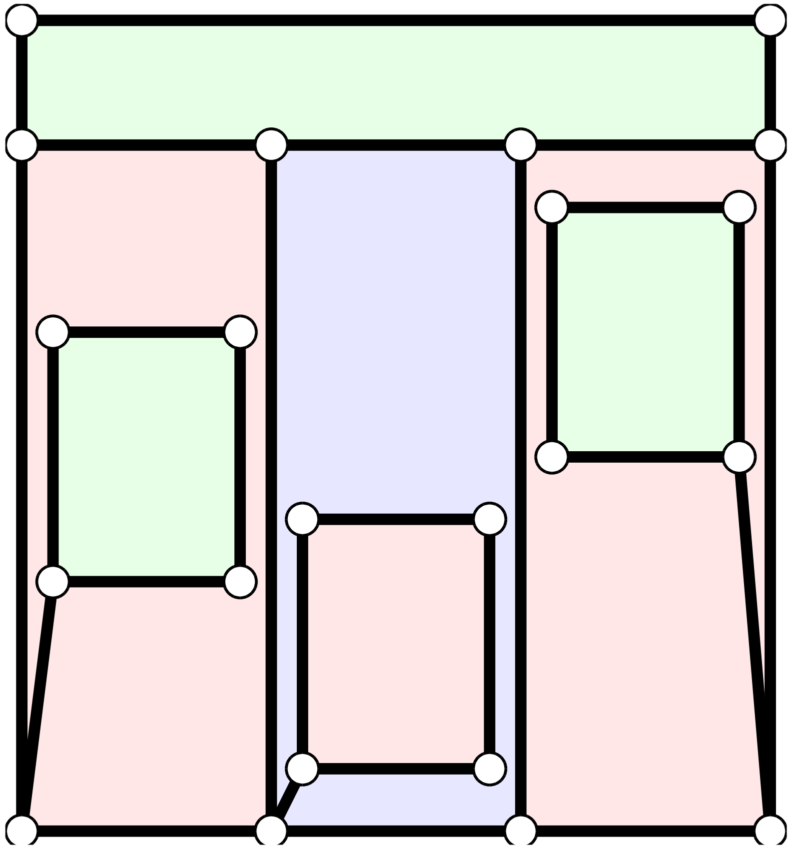
Procedure – step 3 – reconstruction

- Reconstruction of the remaining edges
  - Search for edges to be intersected
  - Calculation of intersection points with the mesh from triangulation



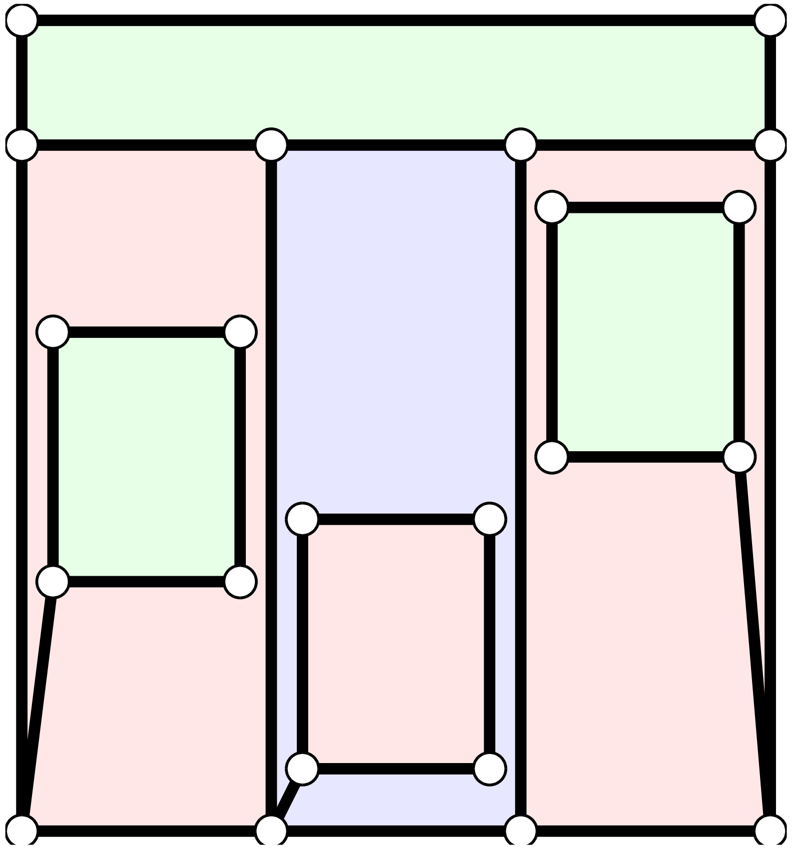
Procedure – step 4 –assignment and clean-up





Procedure – step 4 –assignment and clean-up

- Assignment of feature-ids to polygons, edges and vertices
- Removal of unnecessary edges, faces and vertices

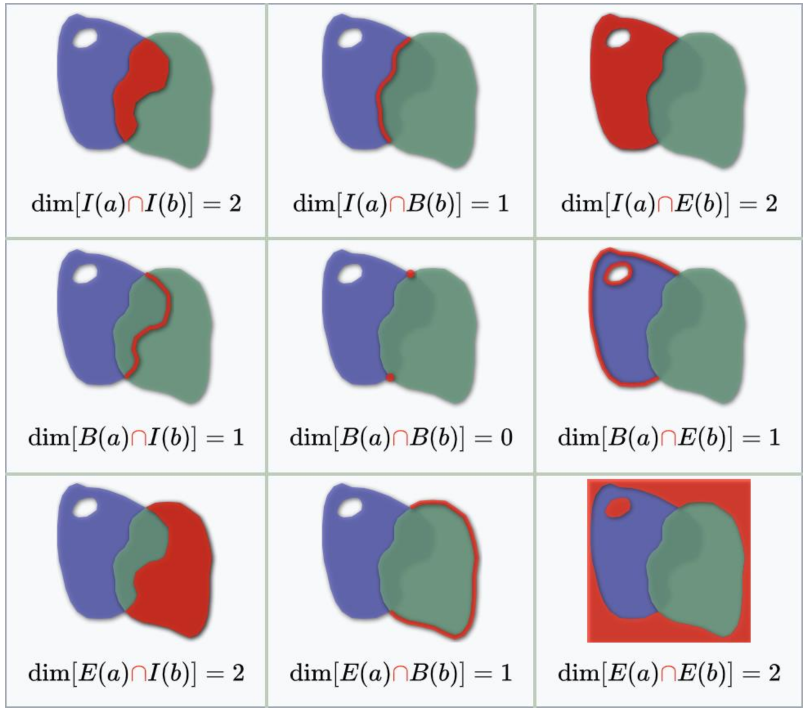


DE-9IM matrices

$$DE9IM(a,b) = \begin{bmatrix} dim(I(a) \cup I(b)) & dim(I(a) \cup B(b)) & dim(I(a) \cup E(b)) \\ dim(B(a) \cup I(b)) & dim(B(a) \cup B(b)) & dim(B(a) \cup E(b)) \\ dim(E(a) \cup I(b)) & dim(E(a) \cup B(b)) & dim(E(a) \cup E(b)) \end{bmatrix}$$

I… Interoir
B… Boundary
E… Exterior

- Simple and unambiguous way to represent spatial relationships



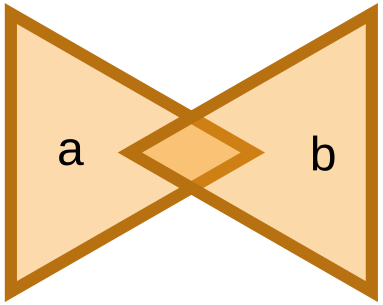
Source: DE-9IM-logoSmall.png (976×862) (wikimedia.org)

DE-9IM matrices

$$DE9IM(a,b) = \begin{bmatrix} dim(I(a) \cup I(b)) & dim(I(a) \cup B(b)) & dim(I(a) \cup E(b)) \\ dim(B(a) \cup I(b)) & dim(B(a) \cup B(b)) & dim(B(a) \cup E(b)) \\ dim(E(a) \cup I(b)) & dim(E(a) \cup B(b)) & dim(E(a) \cup E(b)) \end{bmatrix}$$

I… Interior
B… Boundary
E… Exterior

- Simple and unambiguous way to represent spatial relationships



$$\rightarrow \begin{bmatrix} 2 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 2 \end{bmatrix}$$

Source: DE-9IM-logoSmall.png (976×862) (wikimedia.org)

Dimension-based feature sets

	0-Dimension Vertex	1-Dimension HalfEdge	2-Dimension Face
Interior	Dim0Set Dim1Set Dim2Set	Dim1Set Dim2Set	Dim2Set
Boundary	Dim1Set[a] Dim2Set	Dim2Set	

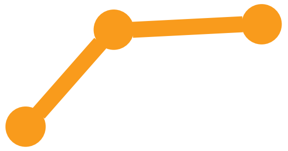
Dimension-based feature sets

- **Point** features correspond to Dim0Set objects

	0-Dimension Vertex	1-Dimension HalfEdge	2-Dimension Face
Interior	Dim0Set Dim1Set Dim2Set	Dim1Set Dim2Set	Dim2Set
Boundary	Dim1Set[a] Dim2Set	Dim2Set	

Dimension-based feature sets

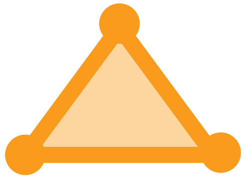
- **Point** features correspond to Dim0Set objects
- **LineStrings** correspond to Dim1Set objects



	0-Dimension Vertex	1-Dimension HalfEdge	2-Dimension Face
Interior	Dim0Set **Dim1Set** Dim2Set	**Dim1Set** Dim2Set	Dim2Set
Boundary	**Dim1Set**[a] Dim2Set	Dim2Set	

Dimension-based feature sets

- **Point** features correspond to Dim0Set objects
- **LineStrings** correspond to Dim1Set objects
- **Polygons** correspond to Dim2Set objects



	0-Dimension Vertex	1-Dimension HalfEdge	2-Dimension Face
Interior	Dim0Set Dim1Set Dim2Set	Dim1Set Dim2Set	Dim2Set
Boundary	Dim1Set[a] Dim2Set	Dim2Set	

Test design

1. Do the features meet the requirements for OGC simple features?

Test design

1. Do the features meet the requirements for OGC simple features?
2. Are there overlapping parcels?

Test design

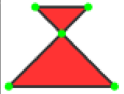
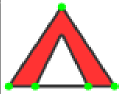
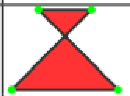
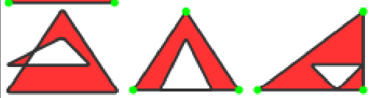
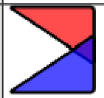
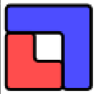
1. Do the features meet the requirements for OGC simple features?
2. Are there overlapping parcels?
3. Are there gaps between parcels?

Test design

1. Do the features meet the requirements for OGC simple features?
2. Are there overlapping parcels?
3. Are there gaps between parcels?
4. On which parcels is a building located?

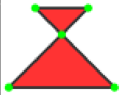
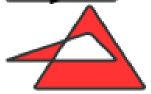
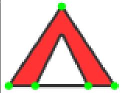
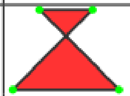
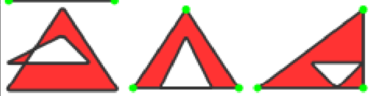
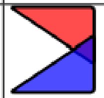
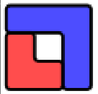
Test 1: Requirements for OGC Simple Features

- Checking for functionality with simple test files.
- Validations always performed at import.

#	Defect	Processing step	WKT or figure
1	consecutive identical points		POLYGON ((0 0,2 0,2 0,...))
2	too few points		POLYGON ((0 0))
3	not closed		POLYGON ((0 0,2 0,1 1,0 1))
4	point used twice	feature / import	
5	overlapping edges		POLYGON ((0 0,0 1,0 0))
6	empty polygon		POLYGON ()
7	wrong orientation		
8	inner polygon is outside		
9	more than one vertex connected to the inner polygon		
10	self intersection	decomposition / generation	
11,12,13	inner face shares outer edge		
14	Test 2 (overlap)	decomposition / query	
15	Test 3 (gap)		

Test 1: Requirements for OGC Simple Features

- Checking for functionality with simple test files.
- Validations always performed at import.
- All defects detected,
  - with our algorithm
  - with FME.

#	Defect	Processing step	WKT or figure
1	consecutive identical points	feature / import	POLYGON ((0 0,2 0,2 0,...))
2	too few points		POLYGON ((0 0))
3	not closed		POLYGON ((0 0,2 0,1 1,0 1))
4	point used twice		
5	overlapping edges		POLYGON ((0 0,0 1,0 0))
6	empty polygon		POLYGON ()
7	wrong orientation		
8	inner polygon is outside		
9	more than one vertex connected to the inner polygon		
10	self intersection	decomposition / generation	
11,12,13	inner face shares outer edge		
14	Test 2 (overlap)	decomposition / query	
15	Test 3 (gap)		

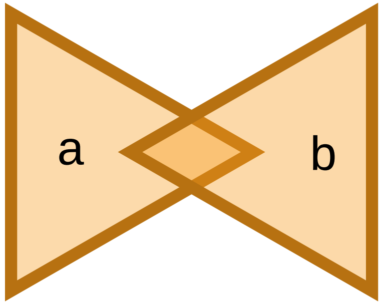
Test 2: Are there overlapping parcels?

- 1040 parcels

Test 2: Are there overlapping parcels?

- 1040 parcels
- Search with simple set operations.

$$relate(a,b) = \begin{pmatrix} 2 & * & * \\ * & * & * \\ * & * & * \end{pmatrix}$$

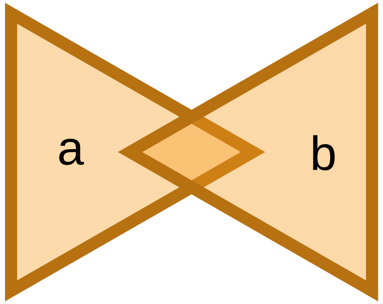


Test 2: Are there overlapping parcels?

- 1040 parcels
- Search with simple set operations.

- No overlapping parcels found,
  - with our algorithm
  - with FME.

$$relate(a,b) = \begin{pmatrix} 2 & * & * \\ * & * & * \\ * & * & * \end{pmatrix}$$

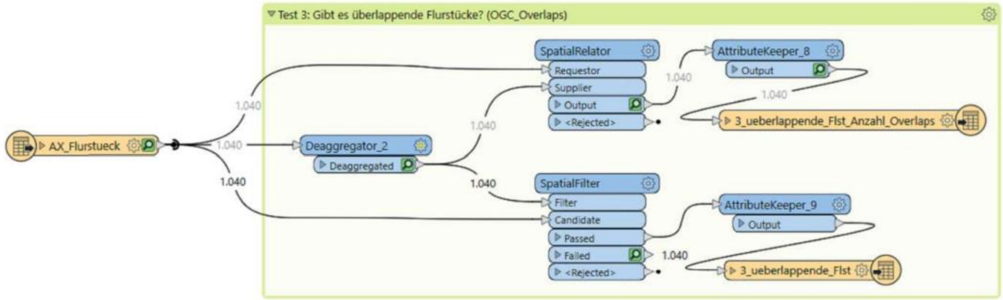


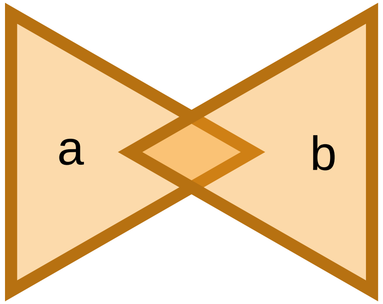
Test 2: Are there overlapping parcels?

- 1040 parcels
- Search with simple set operations.

- No overlapping parcels found,
  - with our algorithm
  - with FME.

$$relate(a,b) = \begin{pmatrix} 2 & * & * \\ * & * & * \\ * & * & * \end{pmatrix}$$





Test 3: Are there gaps between parcels?

- 1040 parcels

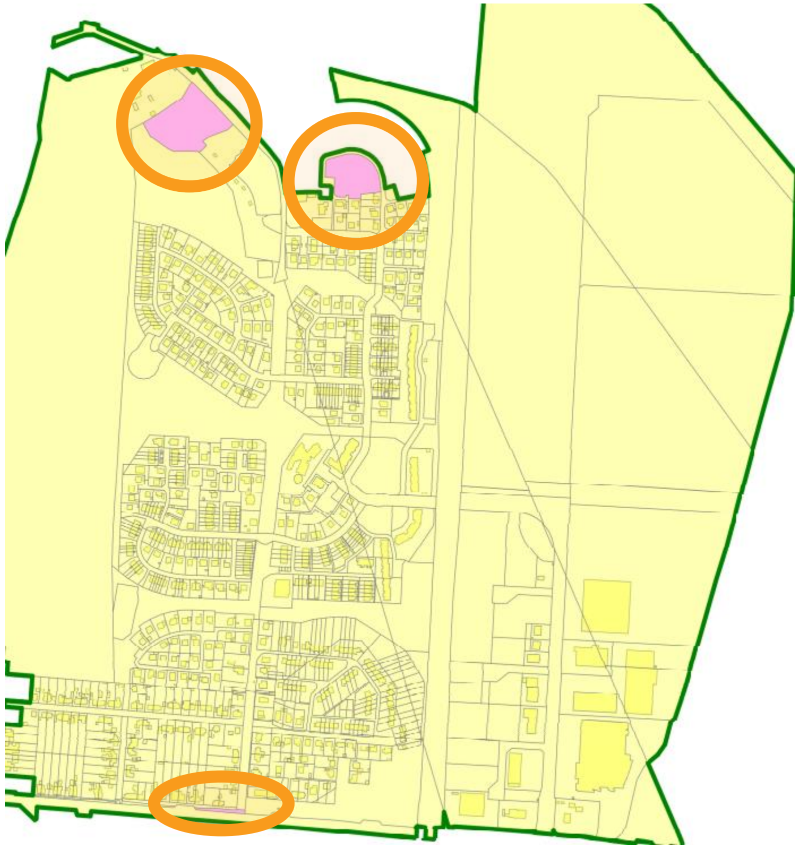
Test 3: Are there gaps between parcels?

- 1040 parcels
- Simple search for faces without id.

Test 3: Are there gaps between parcels?

- 1040 parcels
- Simple search for faces without id.

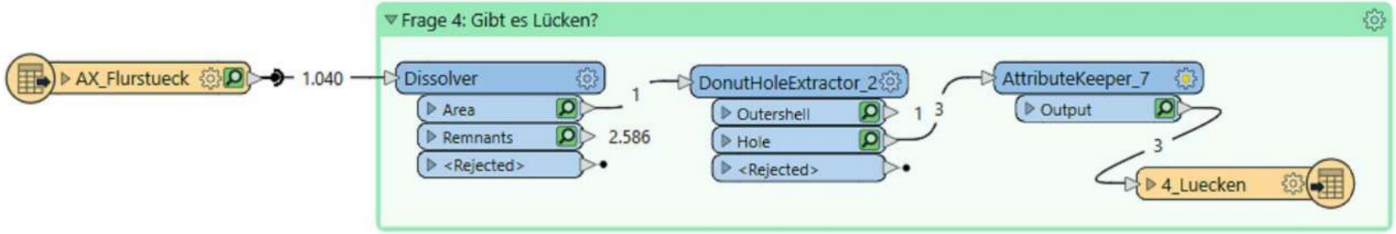
- 3 regions found,
- with our algorithm
- with FME.

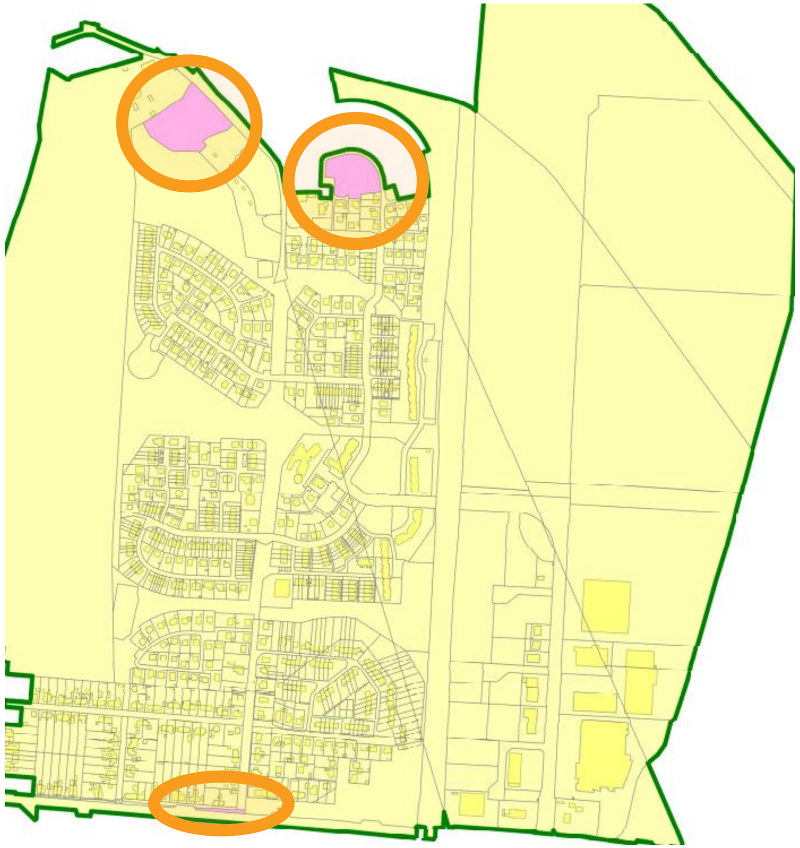


Test 3: Are there gaps between parcels?

- 1040 parcels
- Simple search for faces without id.

- 3 regions found,
- with our algorithm
- with FME.





Test 4: On which parcels is a building located?

- 1040 parcels, 1068 Buildings

Test 4: On which parcels is a building located?

- 1040 parcels, 1068 Buildings
- Search with simple set operations.
  - Buildings on parcels

$$relate(a,b) = \begin{pmatrix} 2 & * & * \\ * & * & * \\ * & * & * \end{pmatrix}$$

Test 4: On which parcels is a building located?

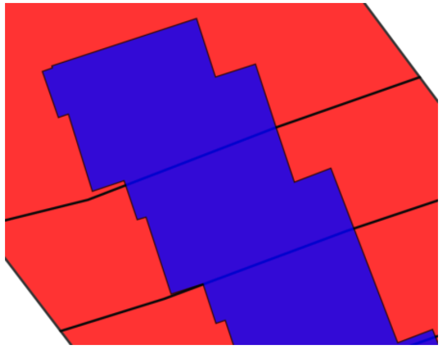
- 1040 parcels, 1068 Buildings
- Search with simple set operations.
  - Buildings on parcels
  - Buildings on boundaries

$$relate(a,b) = \begin{pmatrix} 2 & * & * \\ * & * & * \\ * & * & * \end{pmatrix}$$

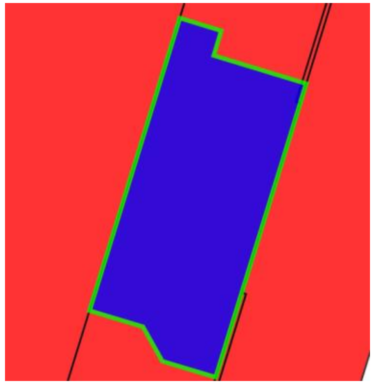
$$relate(a,b) = \begin{pmatrix} * & * & * \\ * & 1 & * \\ * & * & * \end{pmatrix}$$

Test 4: On which parcels is a building located?

- 1040 parcels, 1068 Buildings
- Search with simple set operations.
  - Buildings on parcels
  - Buildings on boundaries



- Same results
  - with our algorithm
  - and with FME.



Conclusion and Outlook

- Functionality of our approach is proven.
- Due to set operations always unambiguous results and easily scalable.

Conclusion and Outlook

- Functionality of our approach is proven.
- Due to set operations always unambiguous results and easily scalable.

- Can also be used in other areas such as TLS or clash detection.
- Use in the three-dimensional field is under development.

Conclusion and Outlook

- Functionality of our approach is proven.
- Due to set operations always unambiguous results and easily scalable.

- Can also be used in other areas such as TLS or clash detection.
- Use in the three-dimensional field is under development.

Kraft B (2016) Ein verfahren der raumzerlegung als grundlage zur prüfung von geometrie und topologie digitaler bauwerksmodelle. DOI 10.14279/depositonce-5117

Huhnt W (2018) Reconstruction of edges in digital building models. Advanced Engineering Informatics 38:474–487, DOI https://doi.org/10.1016/j.aei.2018.08.004

Vetter J (2019) Eine untersuchung zum aufwand bei der berechnung einer raumzerlegung im 2d aus einer gegebenen menge an polygonen. Master's thesis, Technische Universität, Berlin

Vetter J, Huhnt W (2021) Accuracy aspects when transforming a boundary representation of solids into a tetrahedral space partition. In: Proceedings of the EG-ICE 2021 Workshop on Intelligent Computing in Engineering, Universitätsverlag TU Berlin, Germany, pp 320–329

Romanschek E, Clemen C, Huhnt W (2020) From terrestrial laser scans to a surface model of a building: Proof of concept in 2d. In: Ungureanu L, Hartmann T (eds) EG-ICE 2020 Proceedings, Universitätsverlag TU Berlin, pp 432–442

Romanschek E, Clemen C, Huhnt W (2021) A novel robust approach for computing de-9im matrices based on space partition and integer coordinates. ISPRS International Journal of Geo-Information 10(11), DOI 10.3390/ijgi10110715