Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Chair for
INFORMATICS I
Efficient Algorithms and
Knowledge-Based Systems

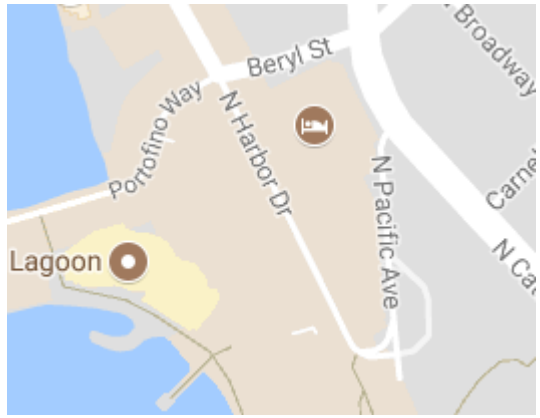
Institute for Informatics

Continuously Generalizing Buildings to Built-up Areas by Aggregating and Growing

Dongliang Peng[1], Guillaume Touya[2]

[1]Chair of Computer Science I, University of Würzburg, Germany
[2]COGIT, IGN, France

zoom out



(Google Maps)

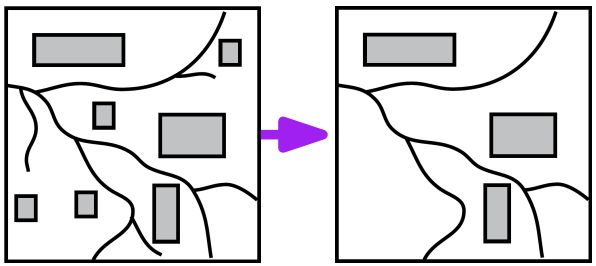
Map Generalization. . .

. . . is about deriving a smaller-scale map from exsiting map.

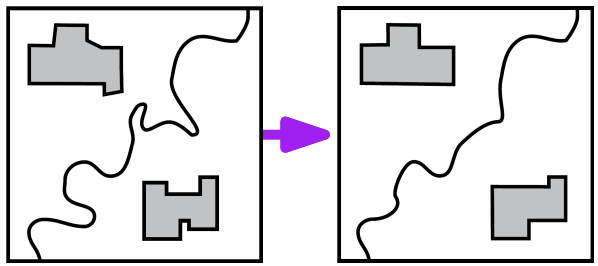
Map Generalization. . .

. . . is about deriving a smaller-scale map from exsiting map.

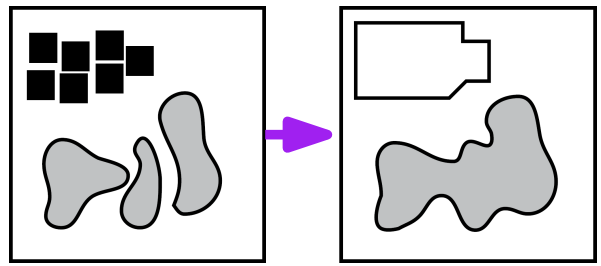
Typical generalization operators (ESRI 1996):



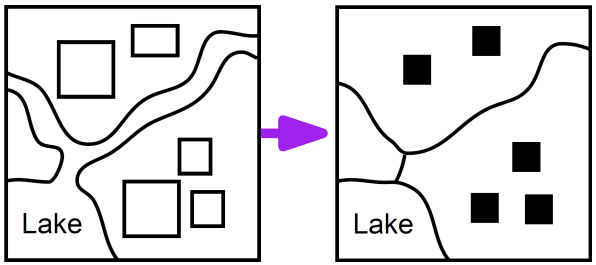
Elimination



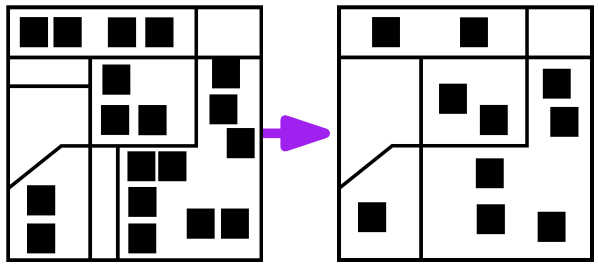
Simplification



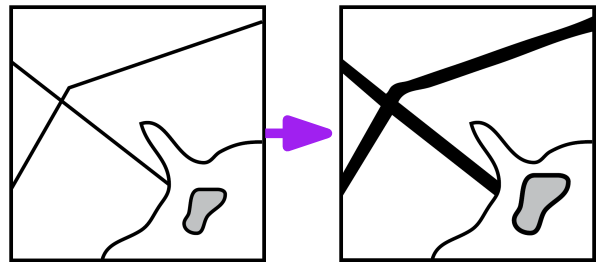
Aggregation



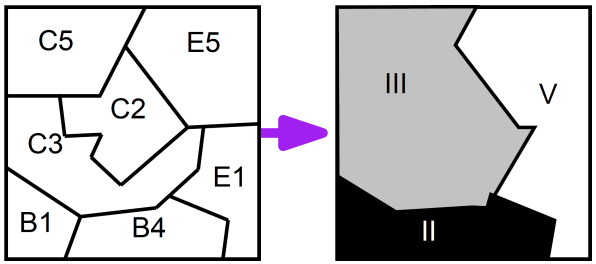
Collapse



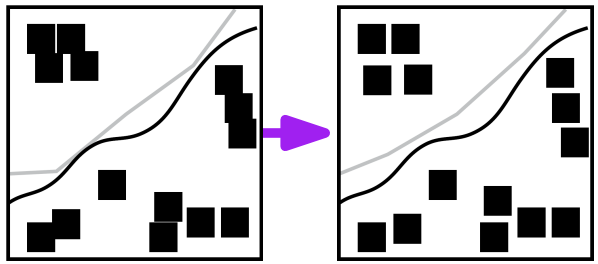
Typification



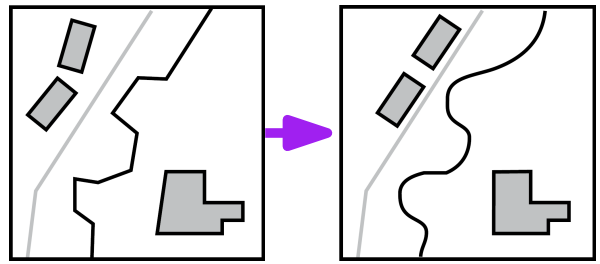
Exaggeration



Classifi. and symboli.



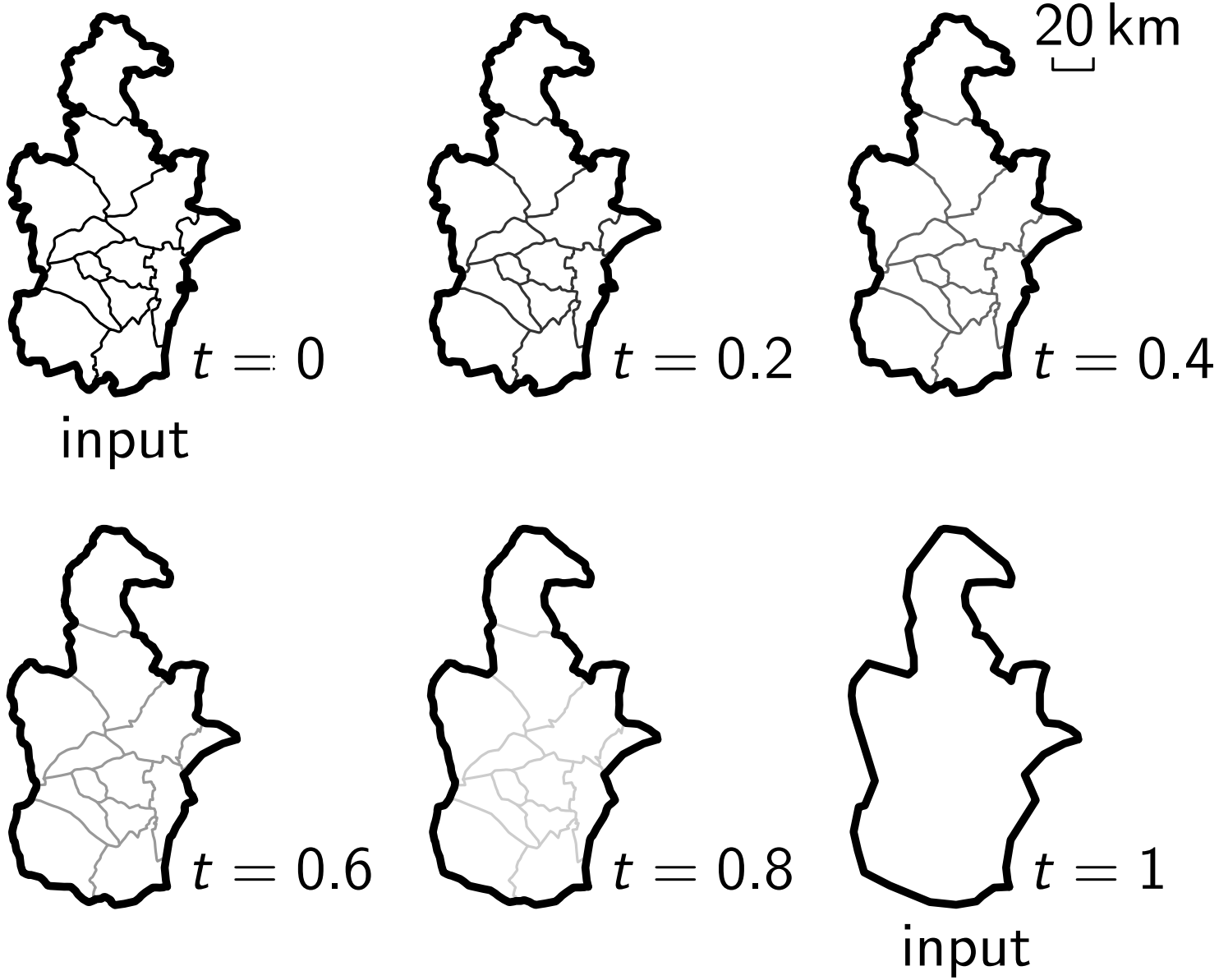
Displacement



Refinement

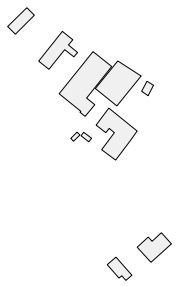
Continuous Map Generalization...

. . . is to derive a series of maps with smooth changes.



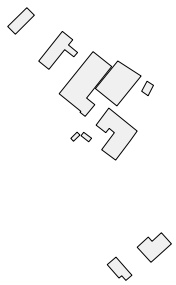
Research Problem

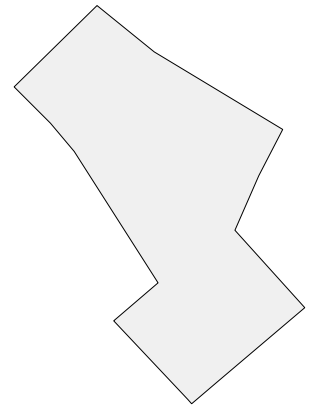
input



Research Problem

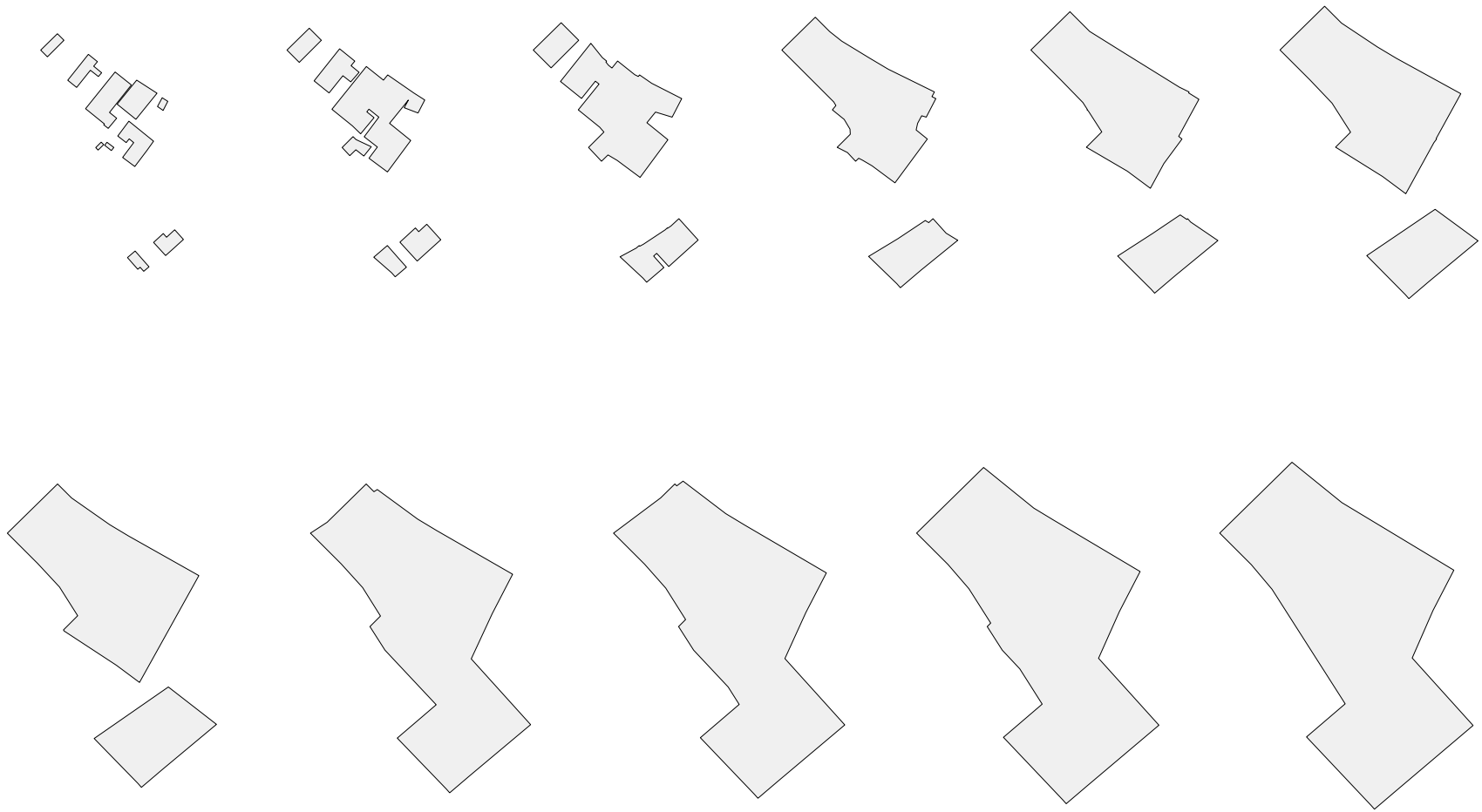
input





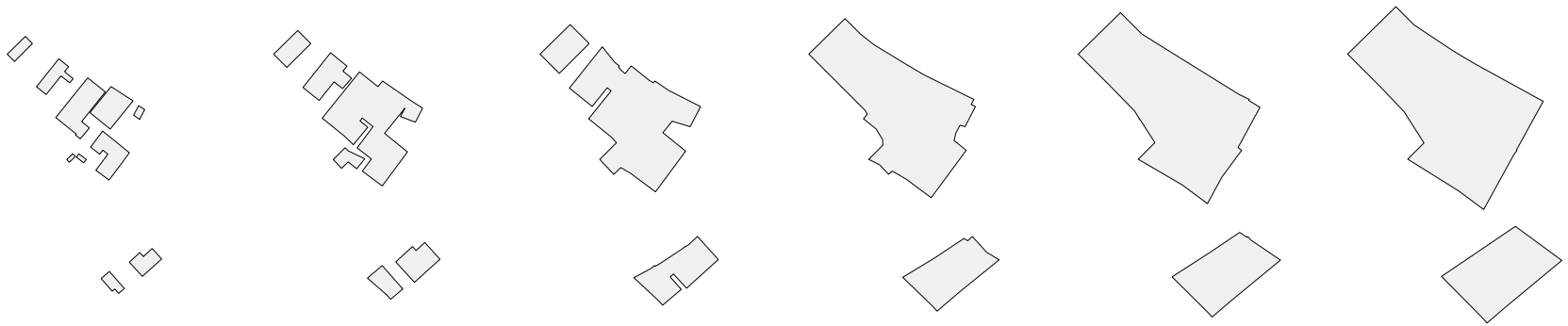
Research Problem

input

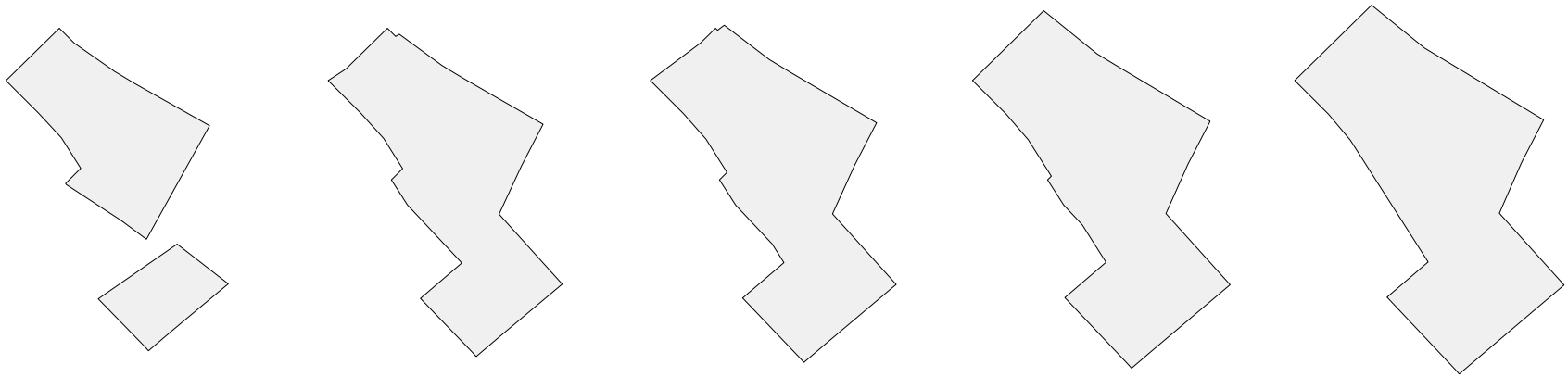


Research Problem

input

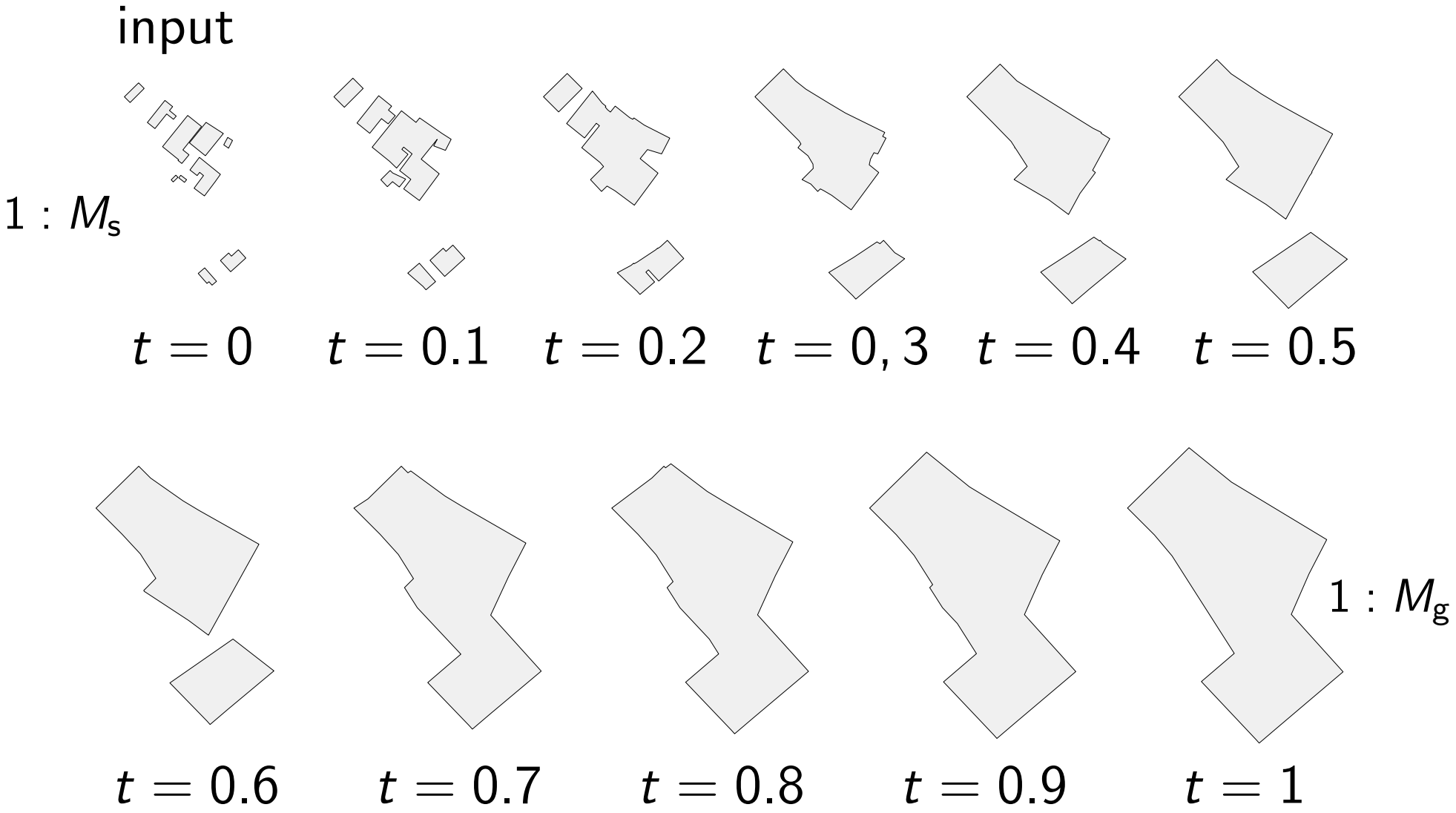


$t = 0$ $t = 0.1$ $t = 0.2$ $t = 0,3$ $t = 0.4$ $t = 0.5$



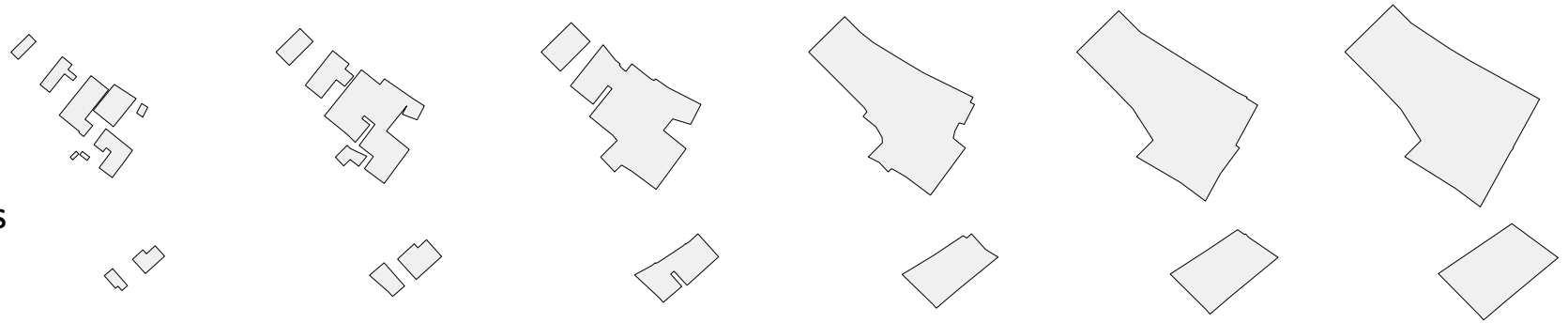
$t = 0.6$ $t = 0.7$ $t = 0.8$ $t = 0.9$ $t = 1$

Research Problem



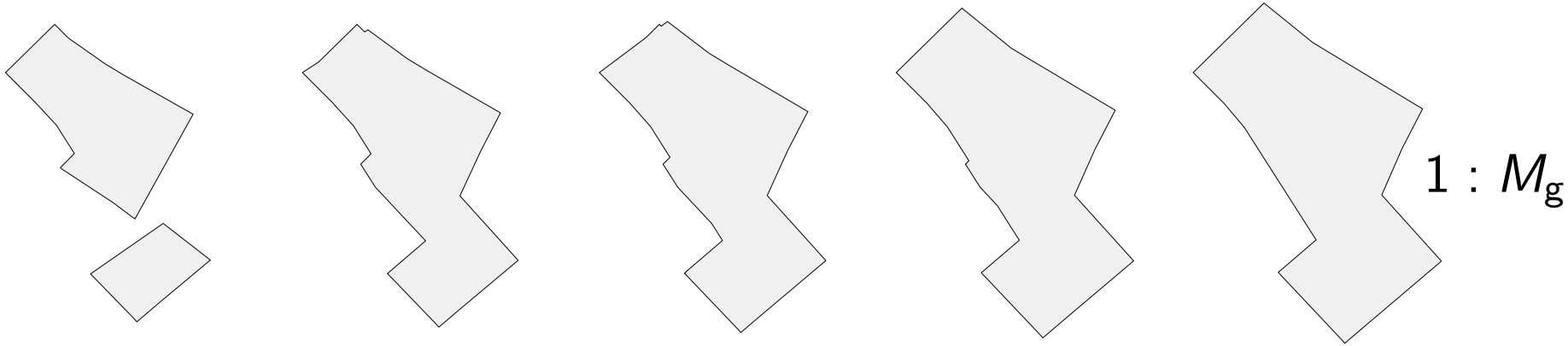
Research Problem

input



$1 : M_s$

$t = 0$ $t = 0.1$ $t = 0.2$ $t = 0,3$ $t = 0.4$ $t = 0.5$



$1 : M_g$

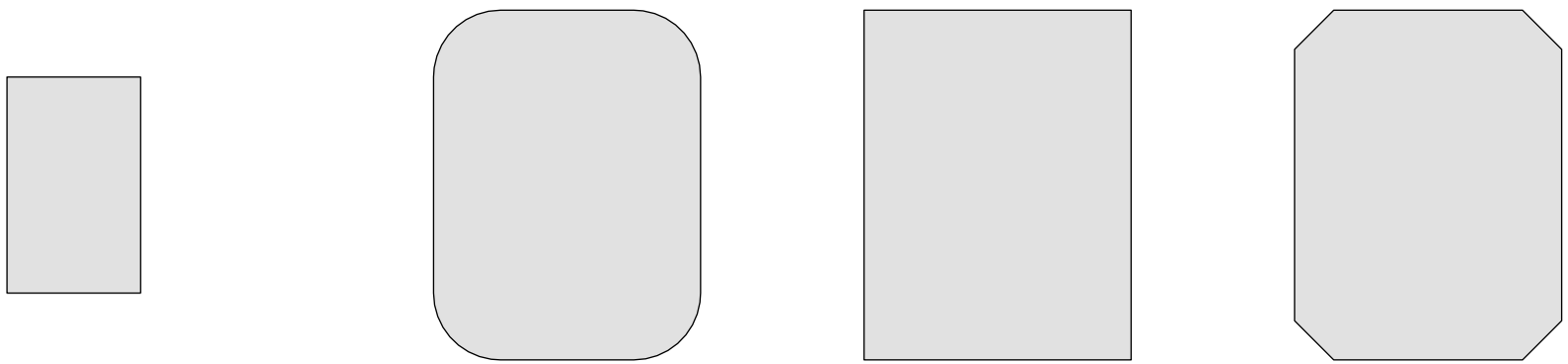
$t = 0.6$ $t = 0.7$ $t = 0.8$ $t = 0.9$ $t = 1$

$$M_t = M_s + t \cdot (M_g - M_s)$$

Outline

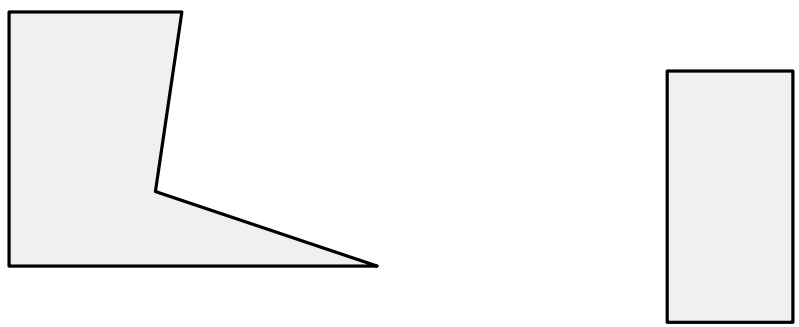
- Introduction
- Methodology
- Case Study
- Concluding Remarks

Three Types of Buffering

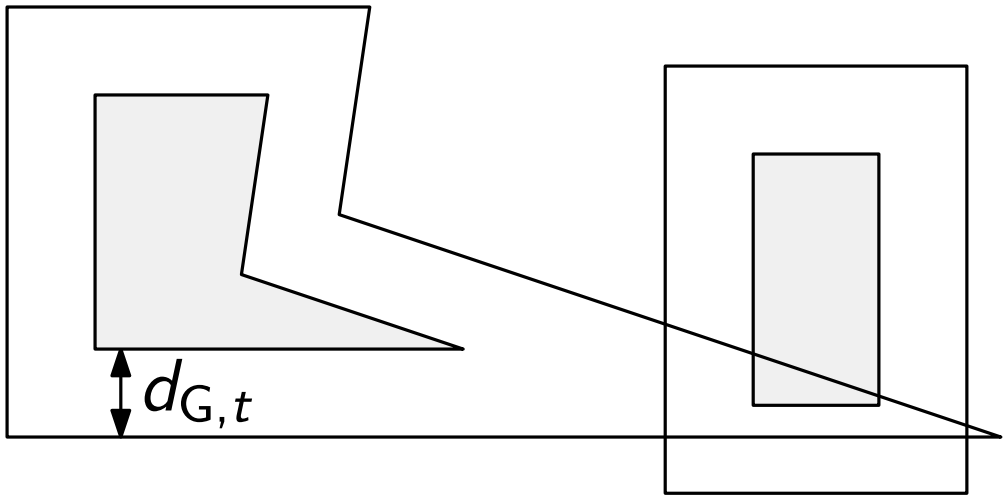


rectangle round miter square

Growing Buildings by Buffering

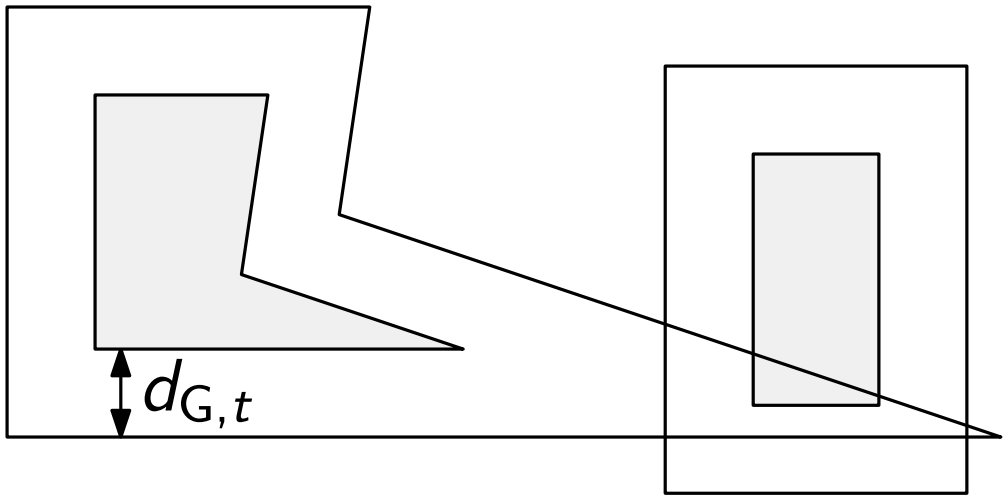


Growing Buildings by Buffering



buffering using miter joins to keep right angles

Growing Buildings by Buffering

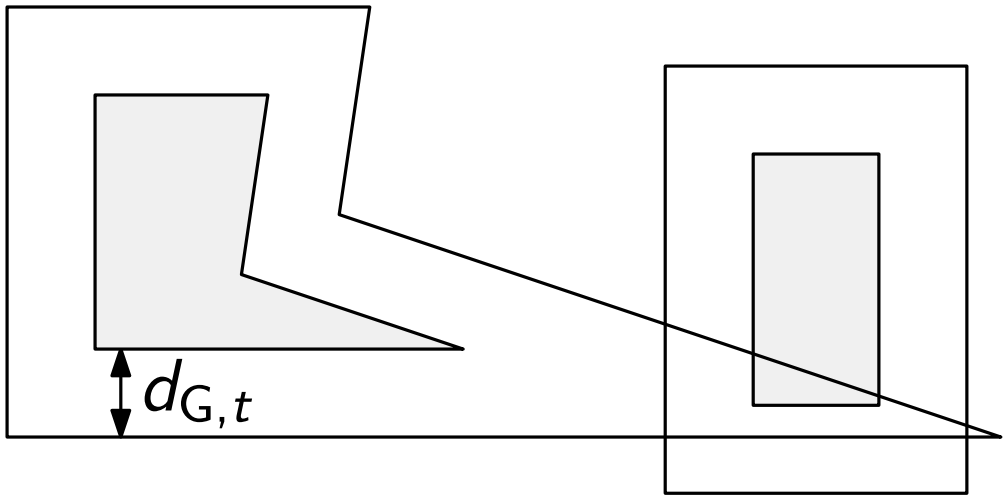


$d_{G,t} = t \cdot d_G$

d_G: input

buffering using miter joins to keep right angles

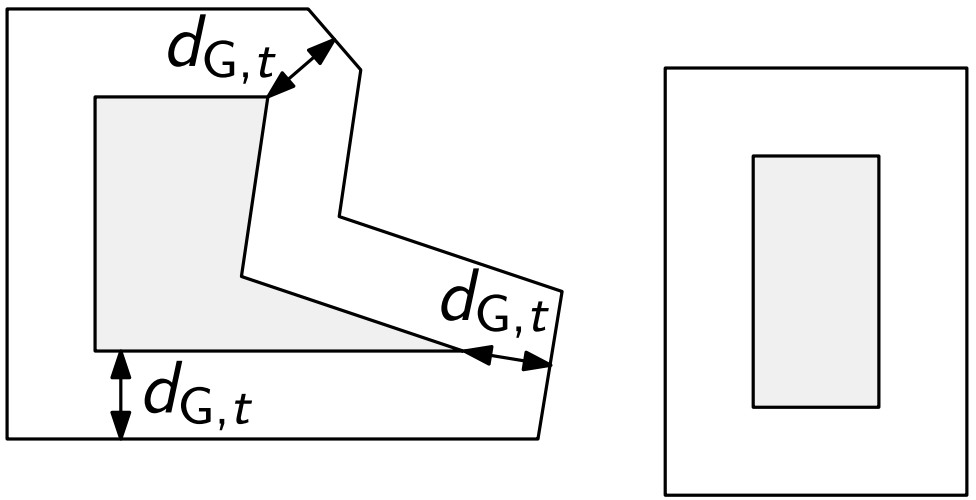
Growing Buildings by Buffering



$d_{G,t} = t \cdot d_G$

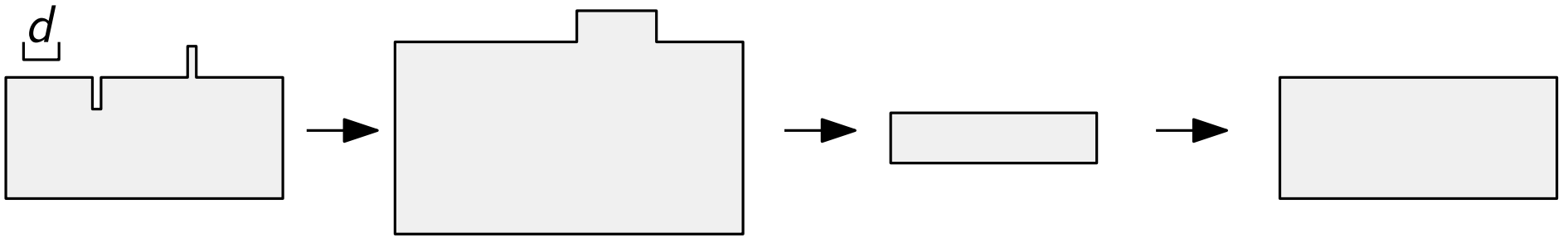
d_G: input

buffering using miter joins to keep right angles

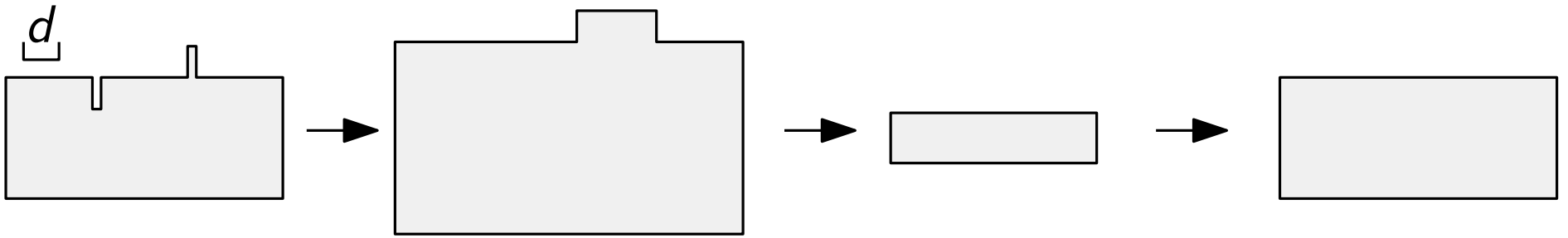


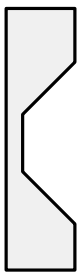
squaring if spikes are too long:
distance larger than $\alpha d_{G,t}$, where we set $\alpha = 1.5$

Simpifying Based on Dilation and Erosion

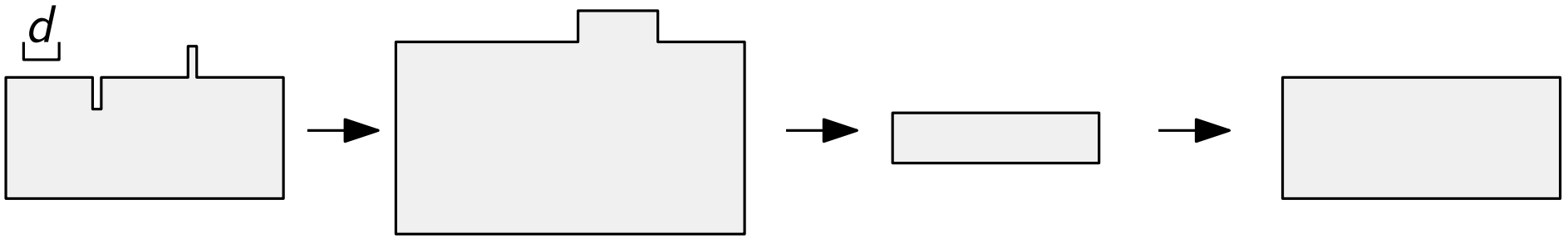


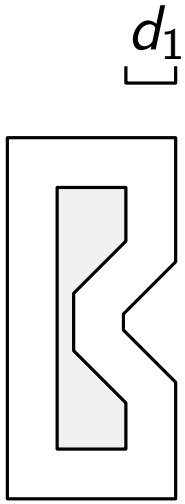
Simpifying Based on Dilation and Erosion



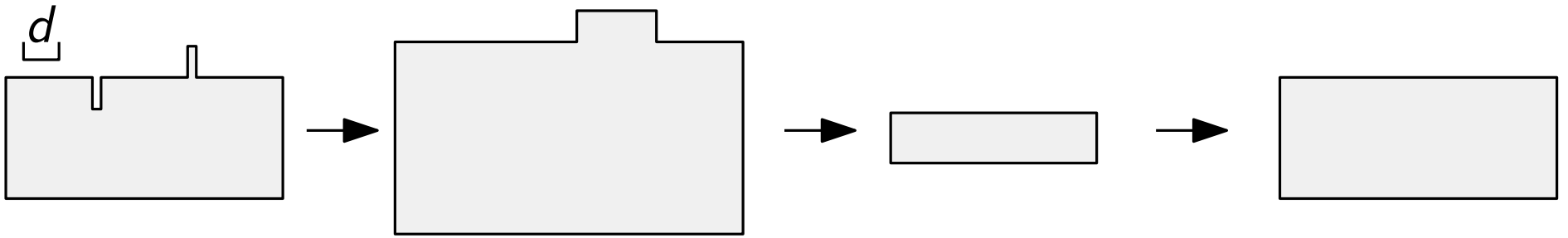


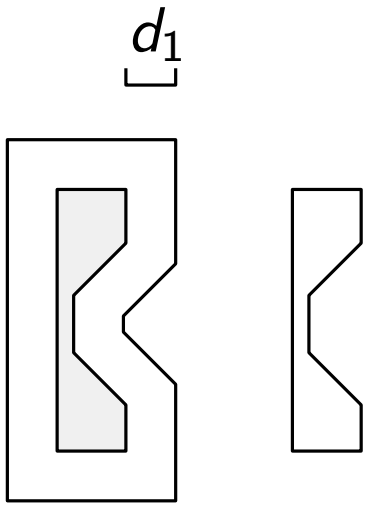
Simpifying Based on Dilation and Erosion



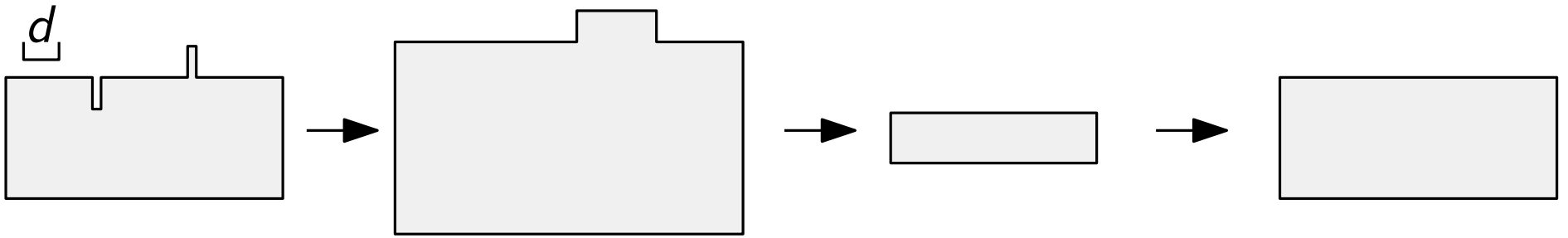


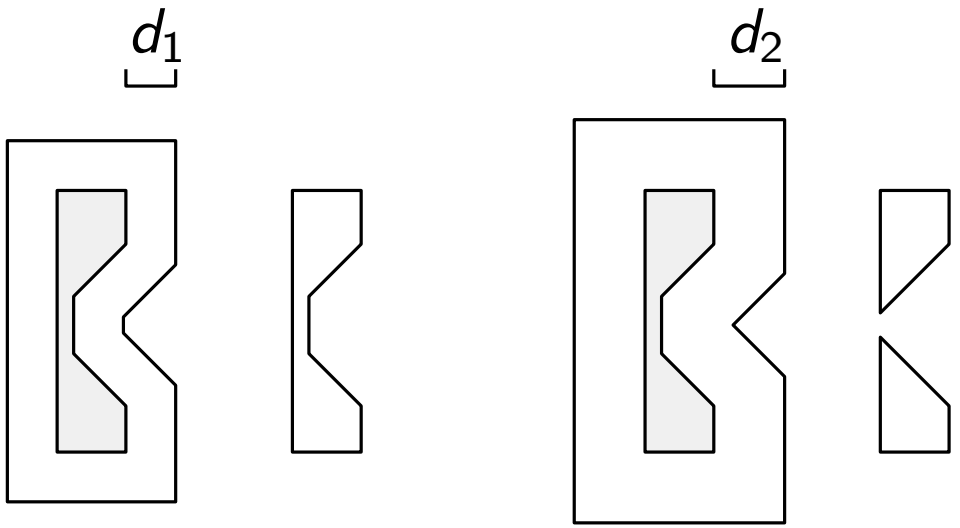
Simpifying Based on Dilation and Erosion



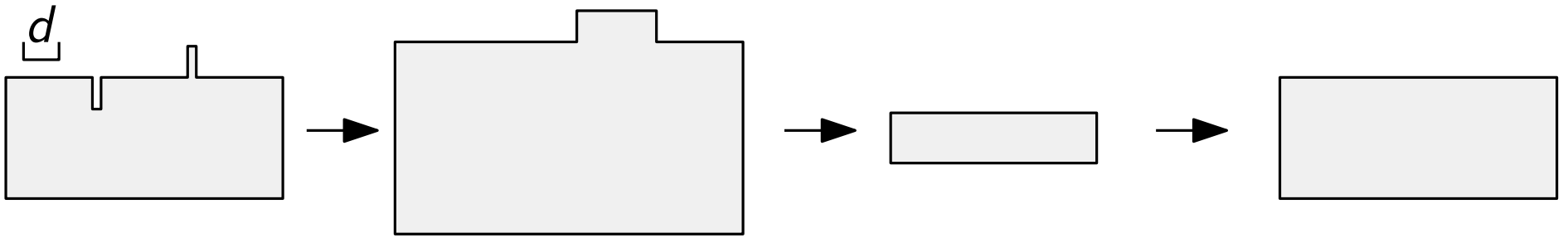


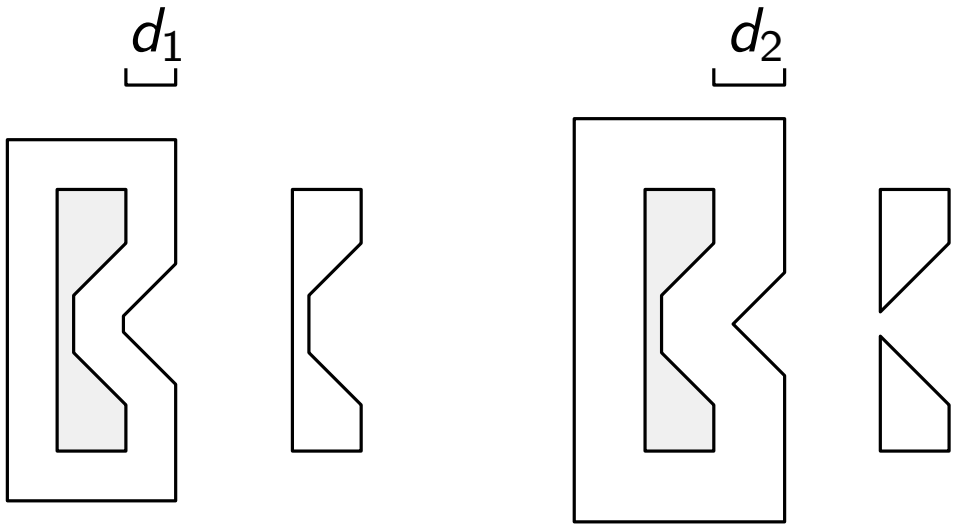
Simpifying Based on Dilation and Erosion





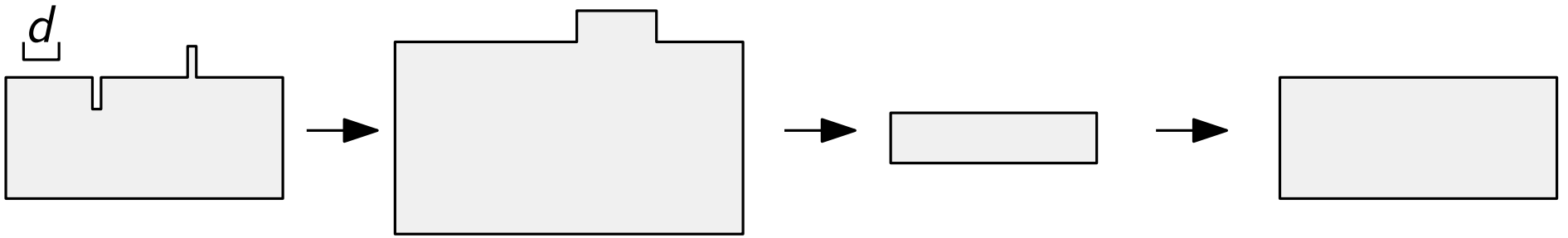
Simpifying Based on Dilation and Erosion

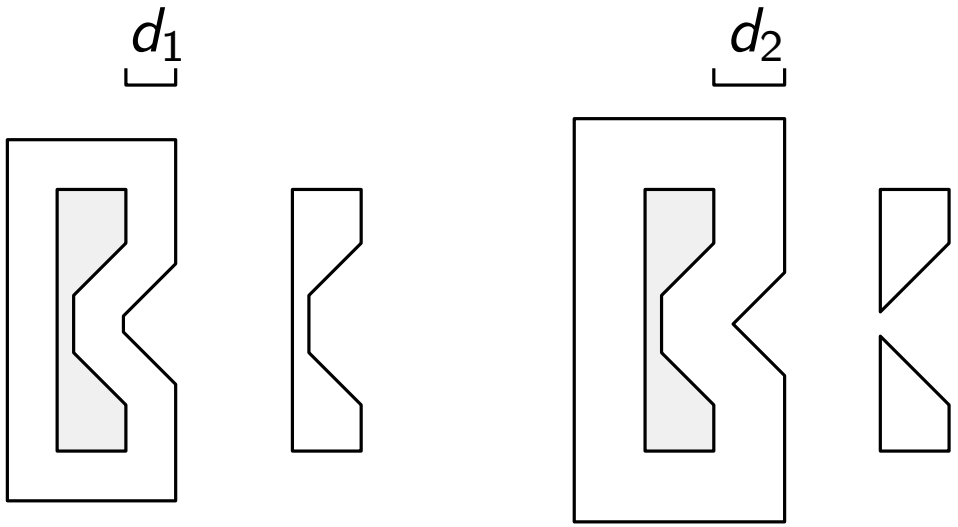




$$d_{\mathrm{E},t} = t \cdot \tfrac{\ell}{2} M_{\mathrm{g}}$$
$$\ell = 0.3\,\mathrm{mm}$$

Simpifying Based on Dilation and Erosion





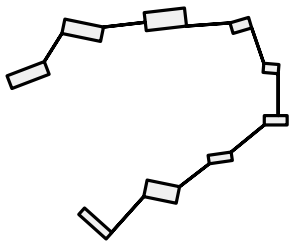
$$d_{\mathrm{E},t} = t \cdot \tfrac{\ell}{2} M_{\mathrm{g}}$$

$$\ell = 0.3\,\mathrm{mm}$$

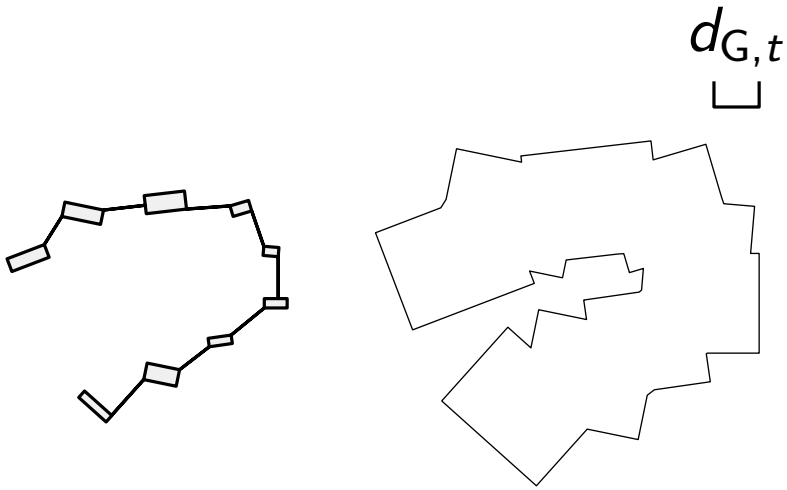
avoid breaking:

$$d_{\mathrm{D},t} = \tfrac{d_{\mathrm{G},t} - d_{\mathrm{E},t}}{\alpha - 1}$$

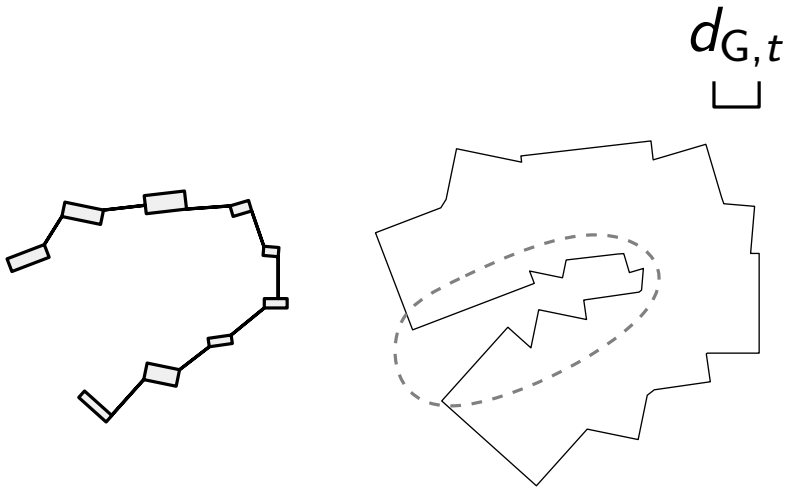
A “bay” Removed by Dilation



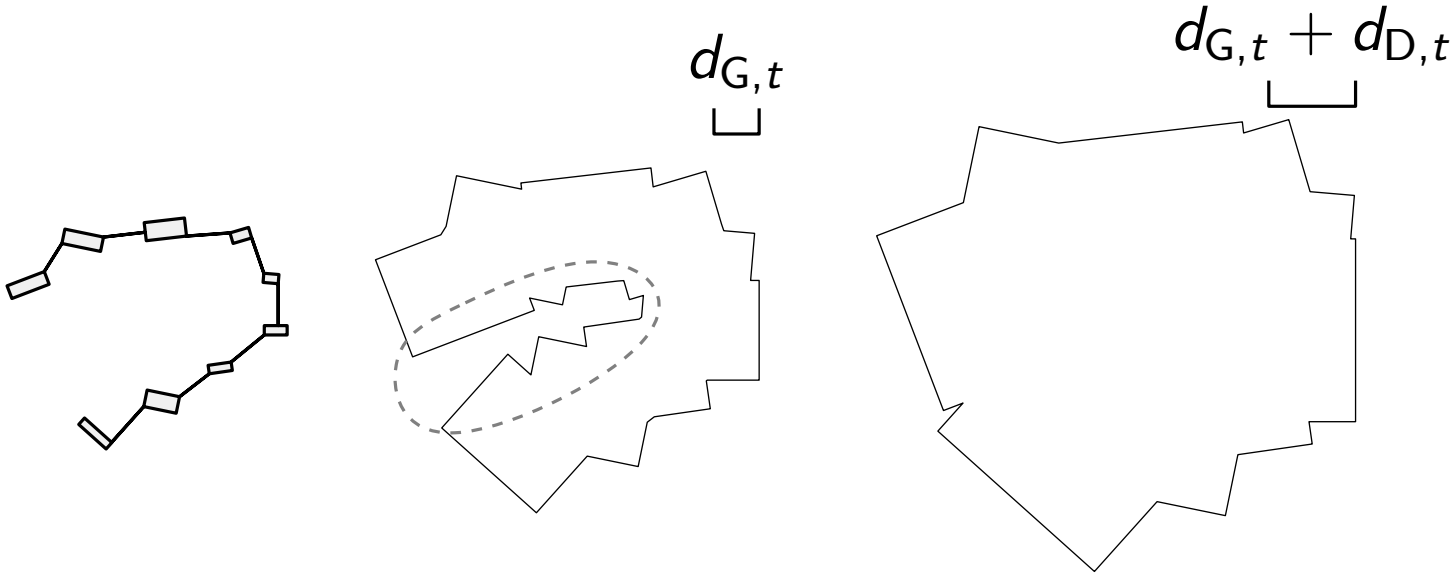
A “bay” Removed by Dilation



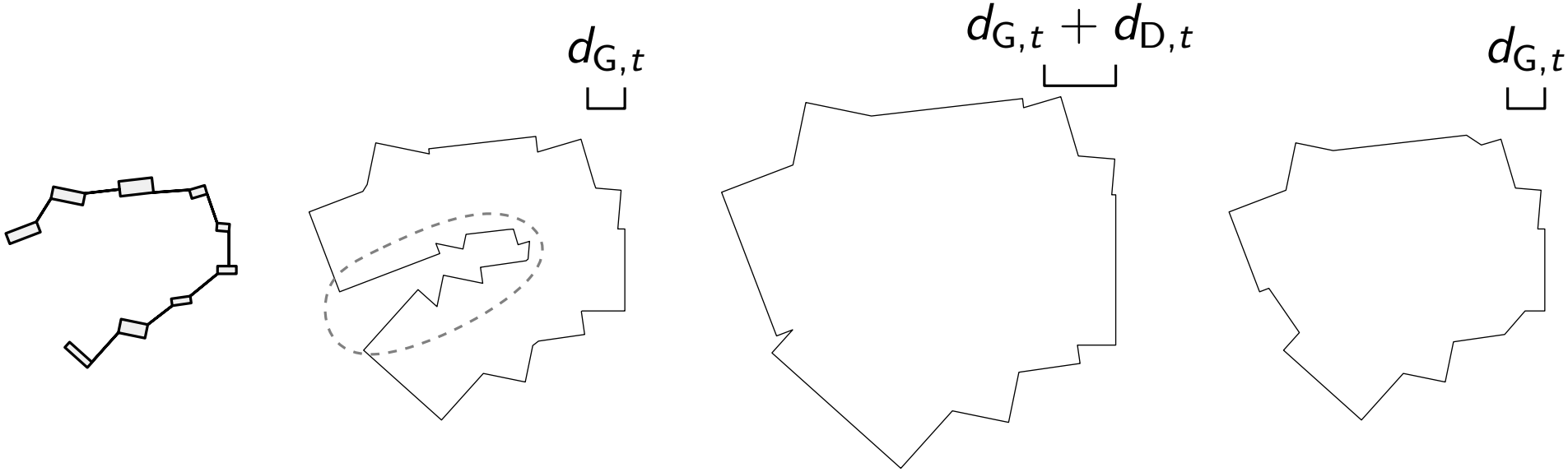
A “bay” Removed by Dilation



A “bay” Removed by Dilation

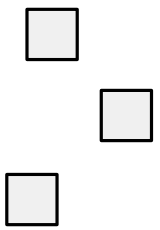


A “bay” Removed by Dilation



Aggregating Buildings by Adding Bridges

- Bridges and buildings constitute
  a minimum spanning tree (MST)

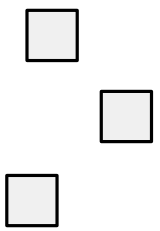


original buildings

$t = 0$

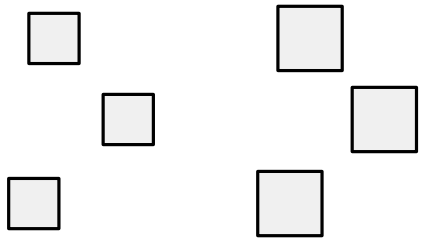
Aggregating Buildings by Adding Bridges

- Bridges and buildings constitute
  a minimum spanning tree (MST)



original buildings

$t = 0$

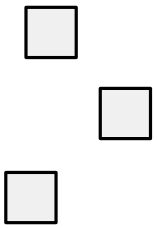


bridge grow

$t = 0.4$

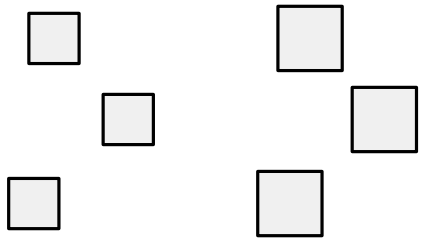
Aggregating Buildings by Adding Bridges

- Bridges and buildings constitute
  a minimum spanning tree (MST)



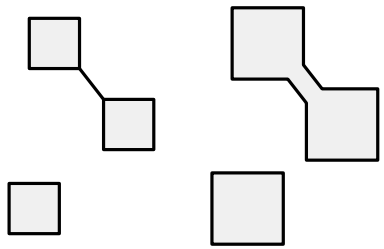
original buildings

$t = 0$



bridge grow

$t = 0.4$

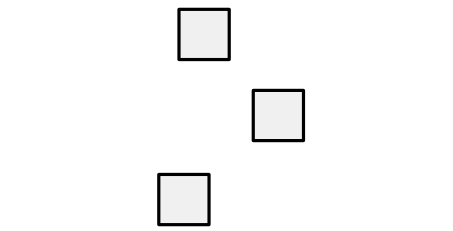


bridge grow

$t = 0.6$

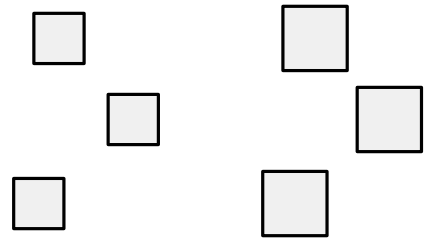
Aggregating Buildings by Adding Bridges

- Bridges and buildings constitute
  a minimum spanning tree (MST)



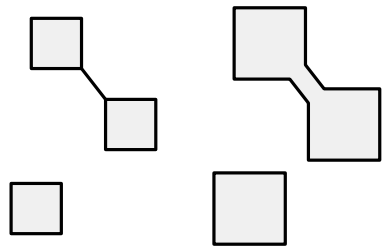
original buildings

$t = 0$



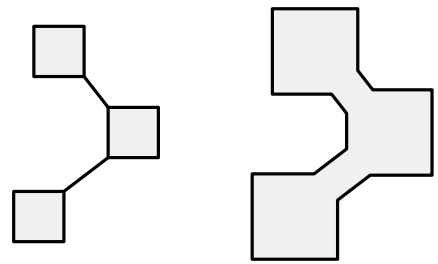
bridge grow

$t = 0.4$



bridge grow

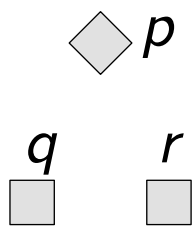
$t = 0.6$



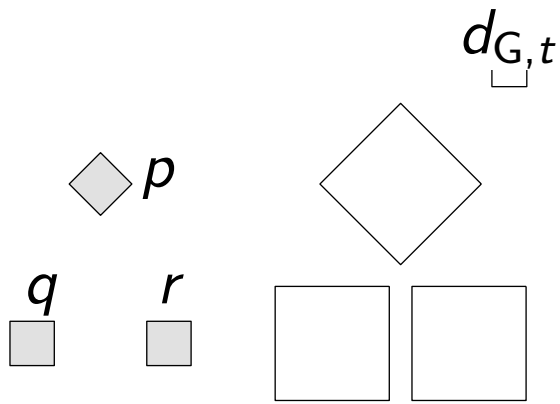
bridge grow

$t = 1$

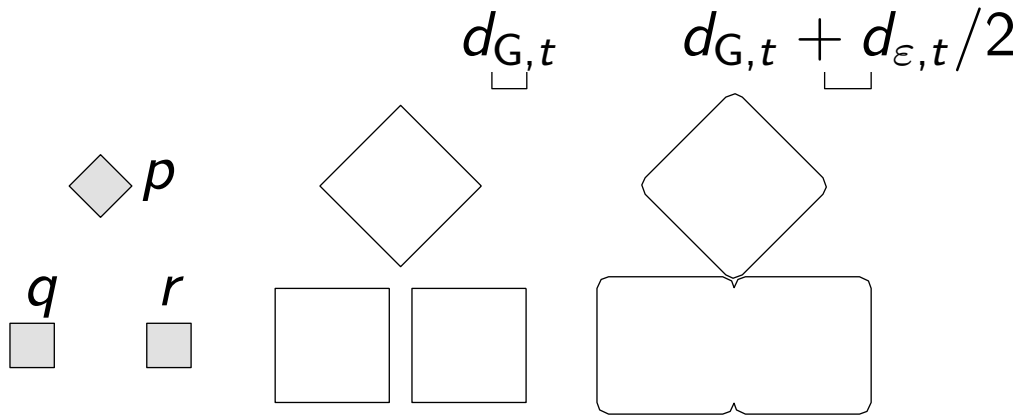
Iteratively Aggregating



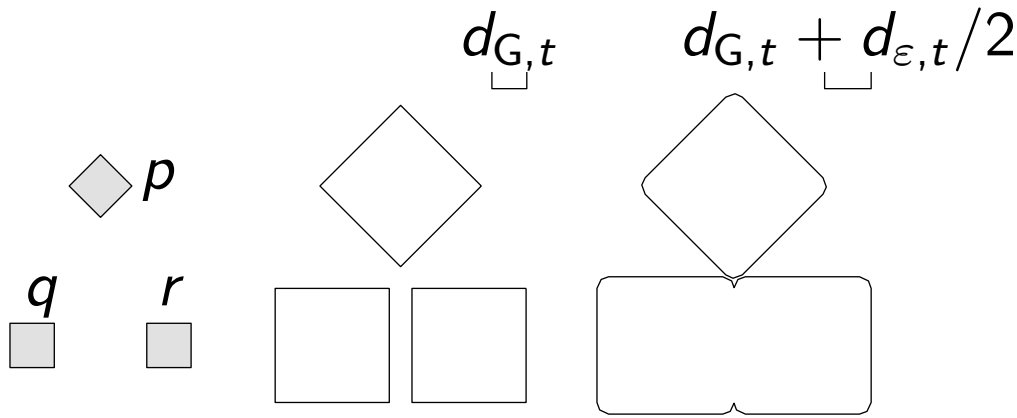
Iteratively Aggregating



Iteratively Aggregating

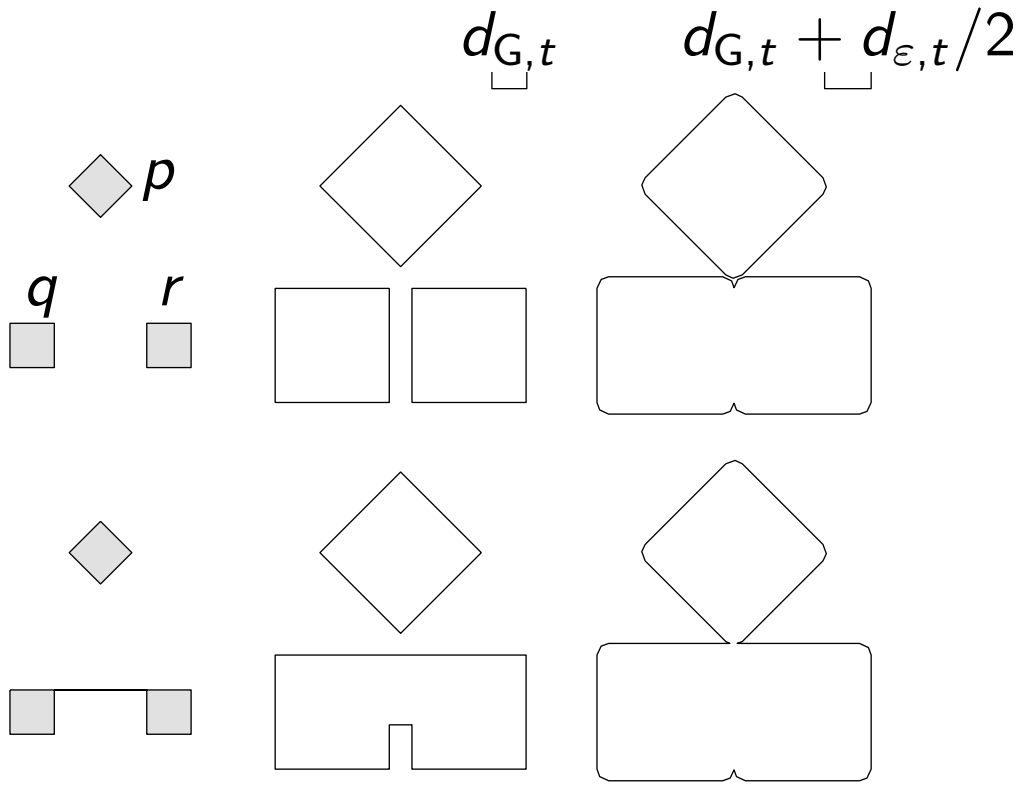


Iteratively Aggregating



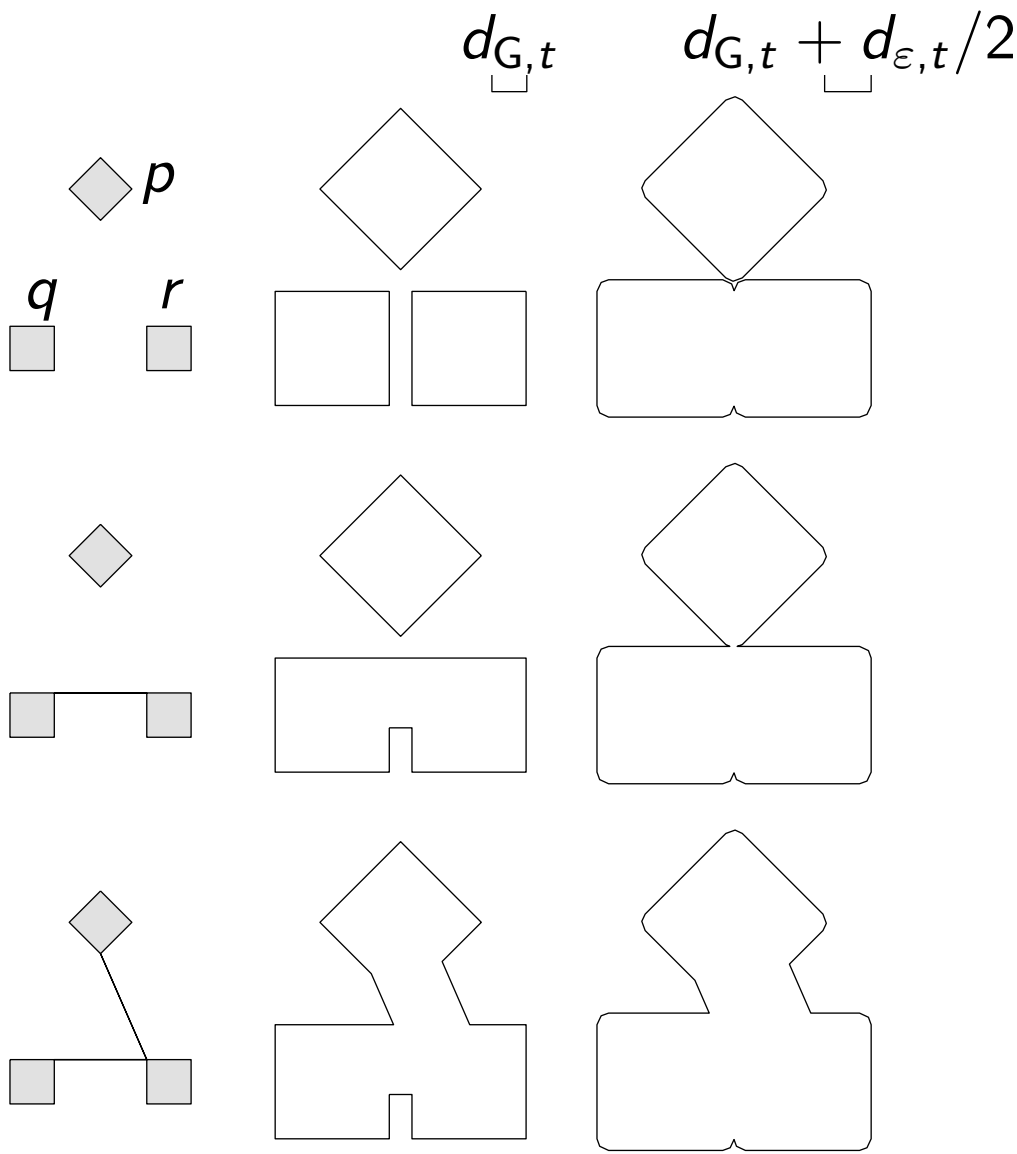
two buildings are too close:
distance $< d_{\varepsilon,t}$

Iteratively Aggregating



two buildings are too close:
distance $< d_{\varepsilon,t}$

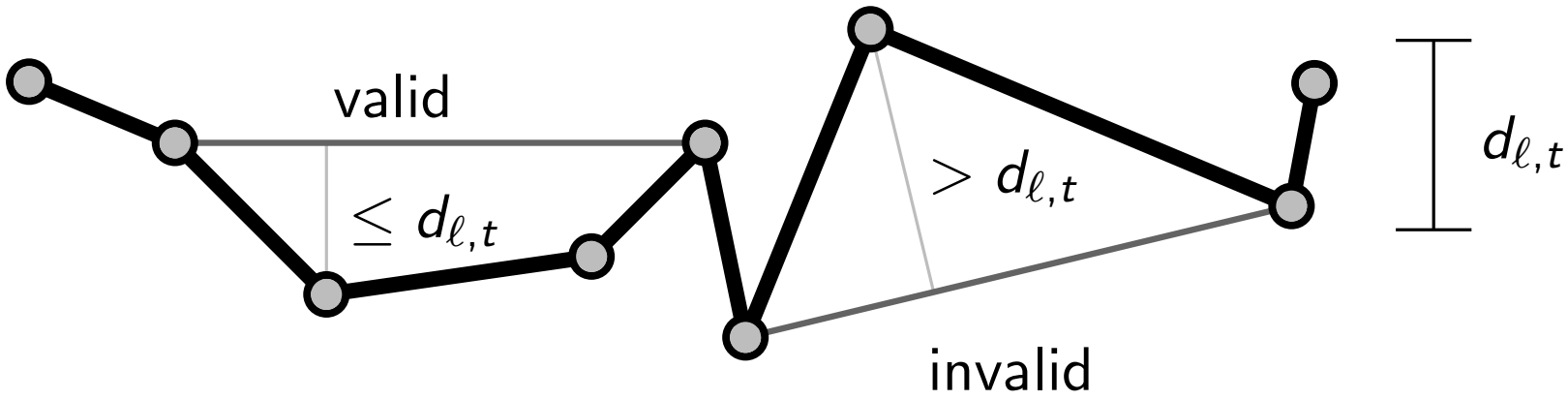
Iteratively Aggregating



two buildings are too close:
distance $< d_{\varepsilon,t}$

Simplifying Using Imai–Iri Algorithm

- Finding all valid shortcuts

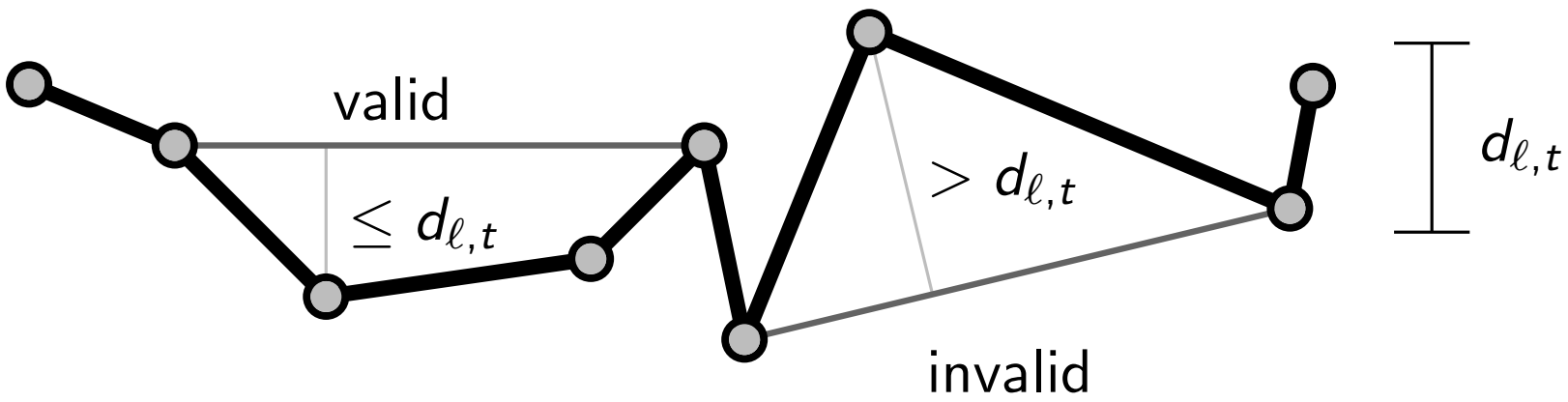


$d_{\ell,t} = \ell \cdot M_t$

$\ell = 0.3\,\mathrm{mm}$

Simplifying Using Imai–Iri Algorithm

- Finding all valid shortcuts
- Finding a sequence of valid shortcus with the least number using breadth-first search

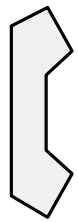


$d_{\ell,t} = \ell \cdot M_t$

$\ell = 0.3\,\mathrm{mm}$

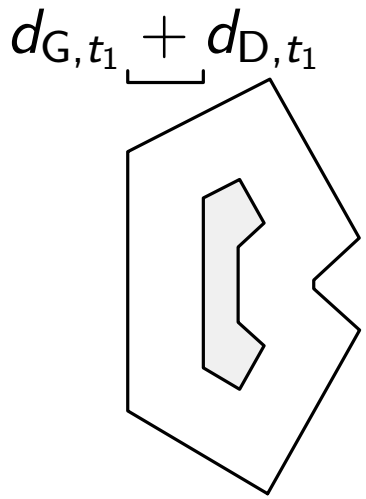
Producing Intermediate-scale maps

A building may shrink because of erosion



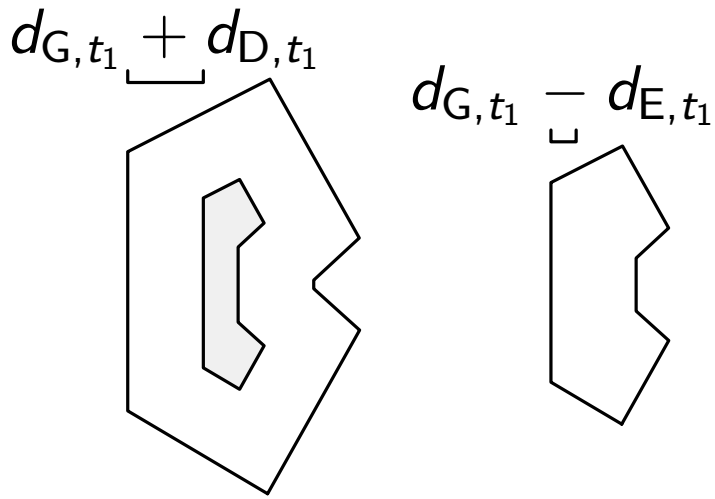
Producing Intermediate-scale maps

A building may shrink because of erosion



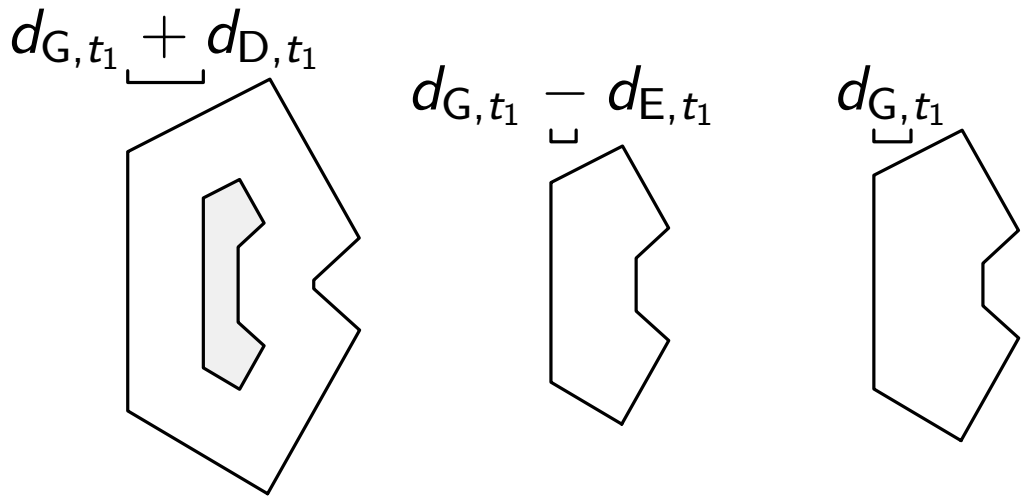
Producing Intermediate-scale maps

A building may shrink because of erosion



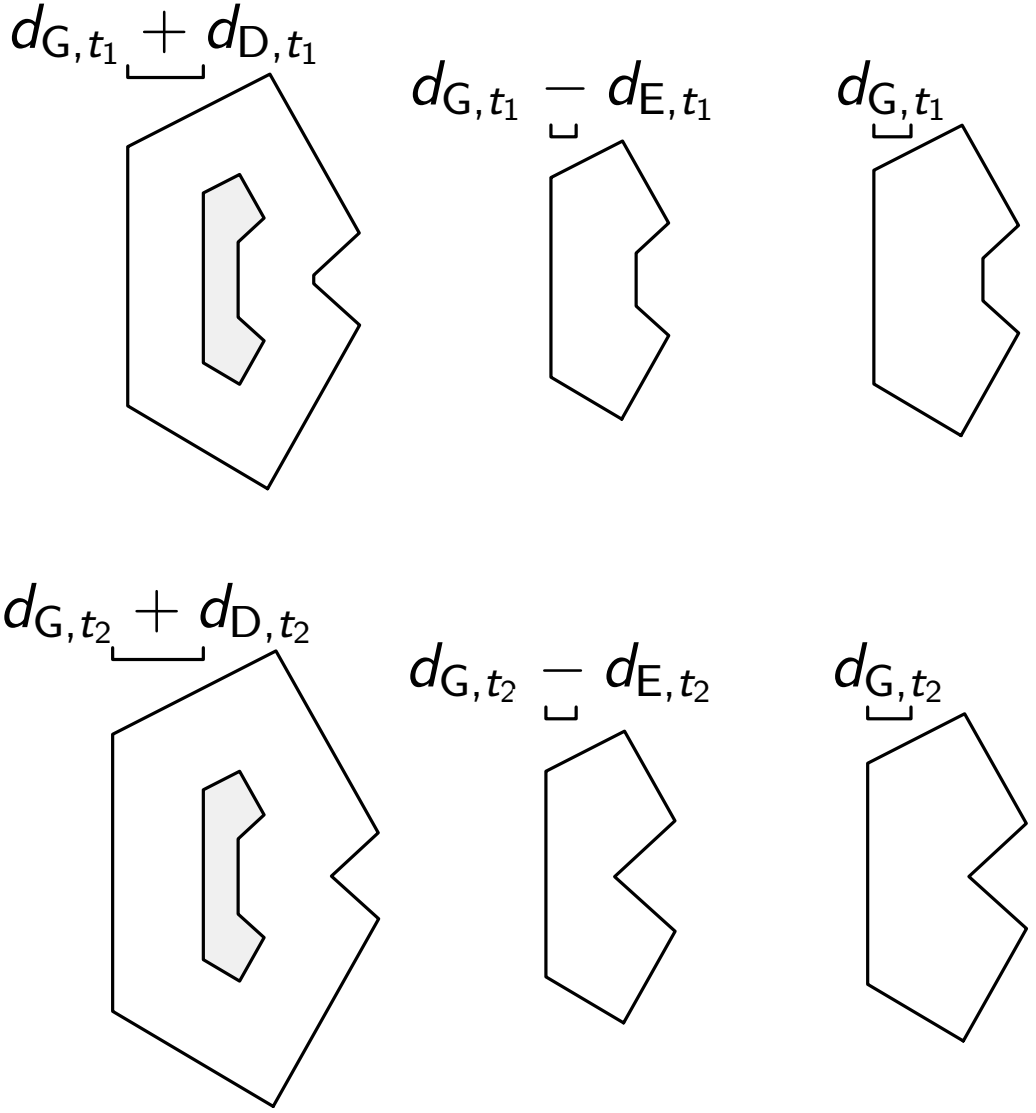
Producing Intermediate-scale maps

A building may shrink because of erosion



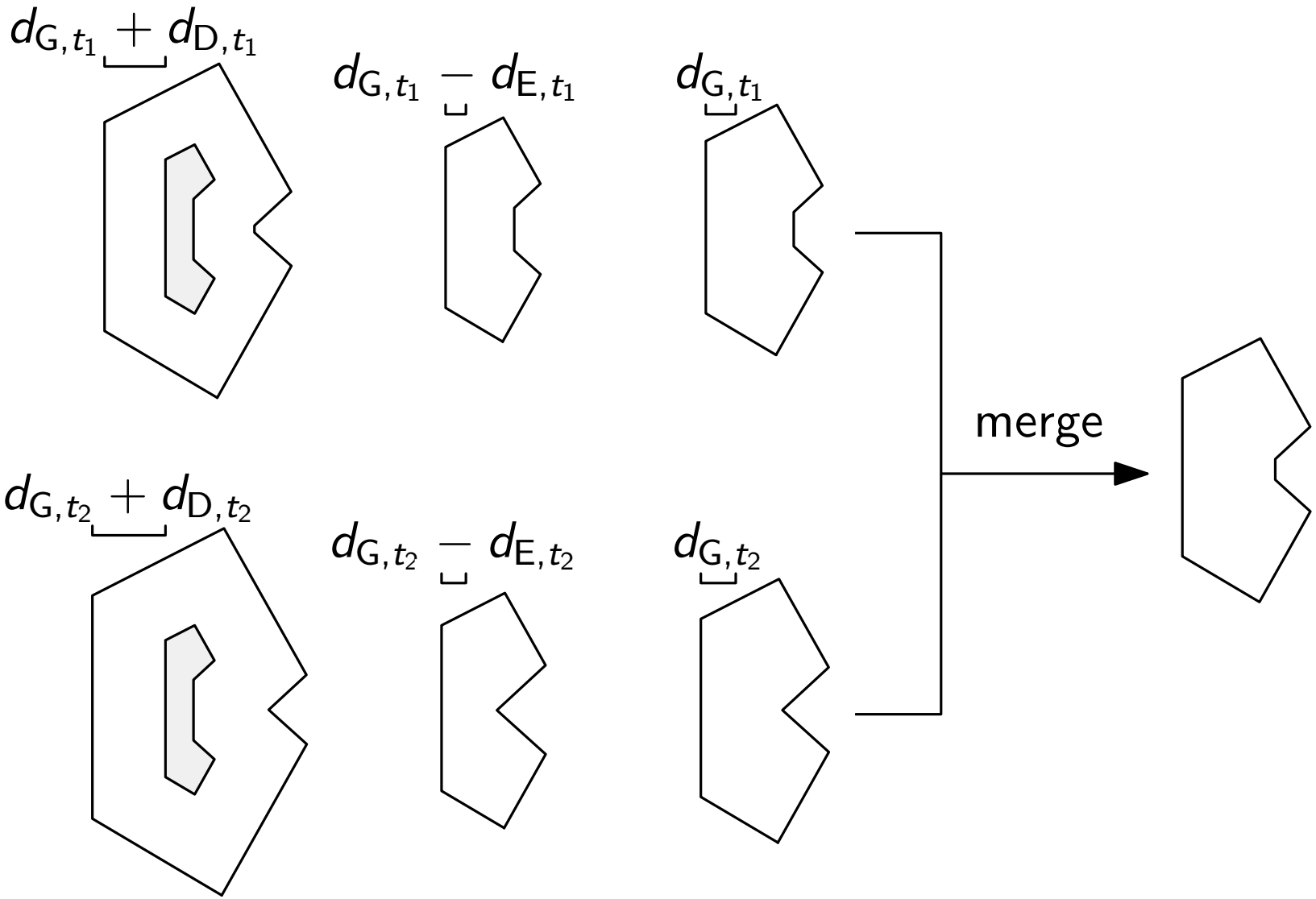
Producing Intermediate-scale maps

A building may shrink because of erosion



Producing Intermediate-scale maps

A building may shrink because of erosion



Producing Intermediate-scale maps

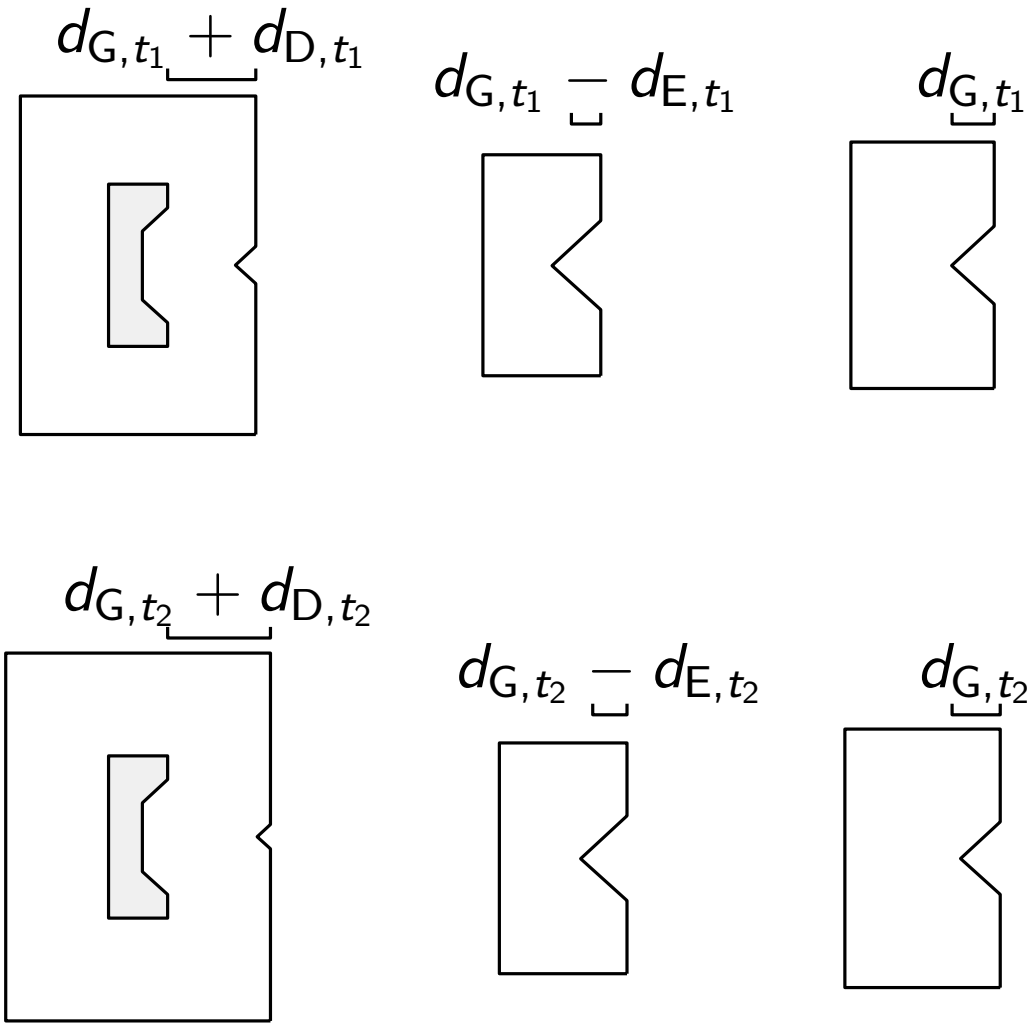
A building may shrink because of line simplification

Producing Intermediate-scale maps

A building may shrink because of line simplification

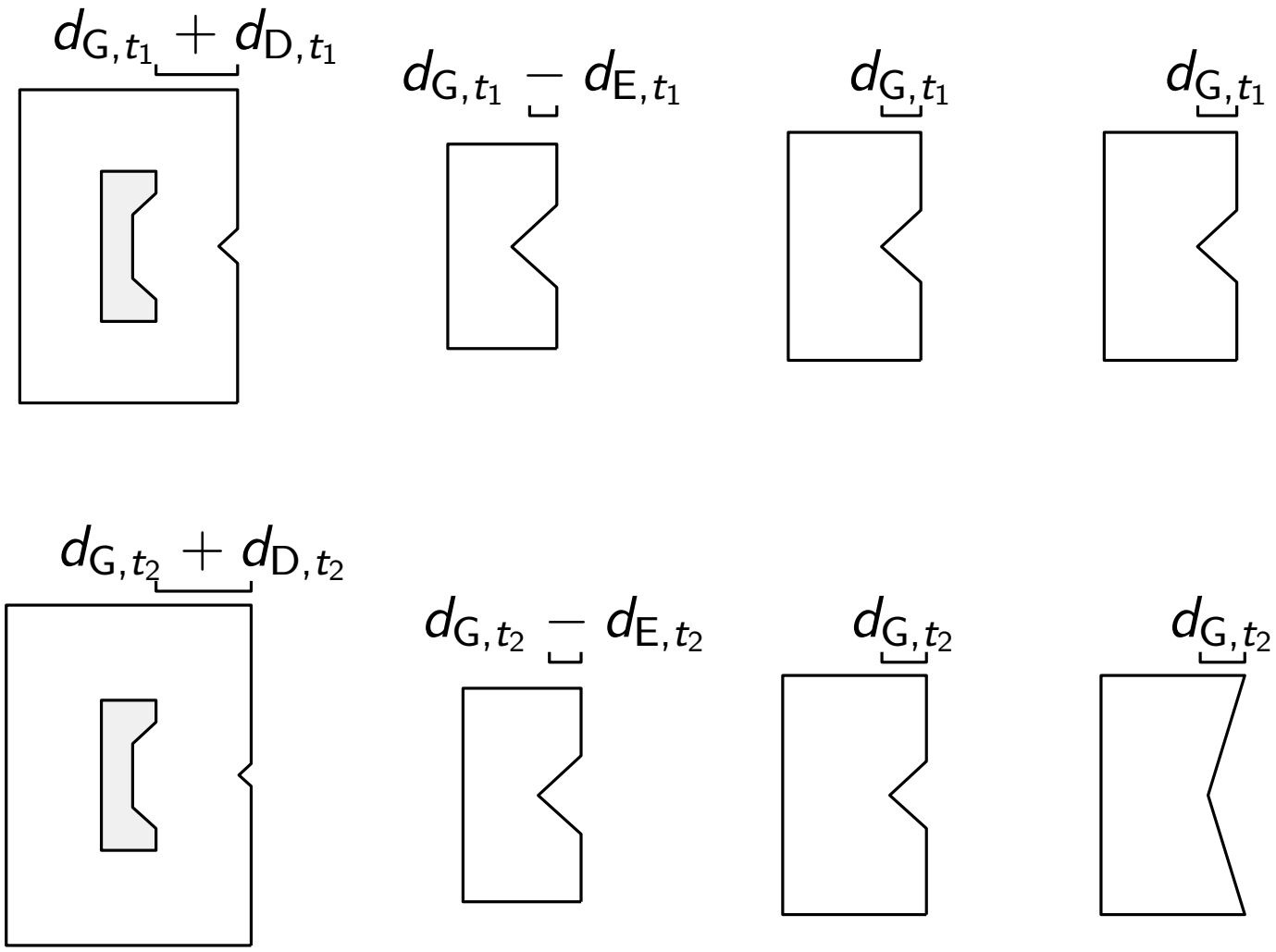
Producing Intermediate-scale maps

A building may shrink because of line simplification



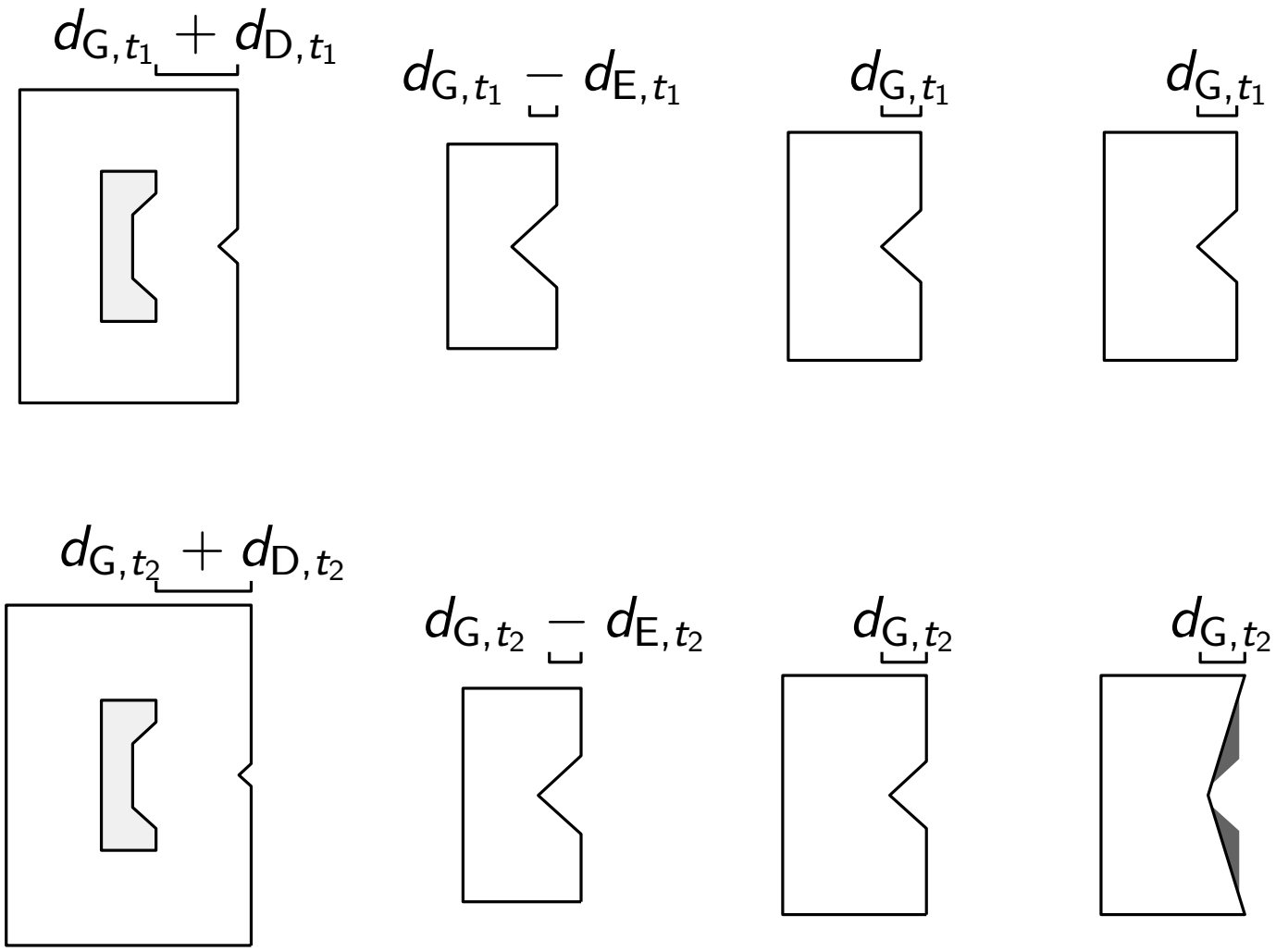
Producing Intermediate-scale maps

A building may shrink because of line simplification



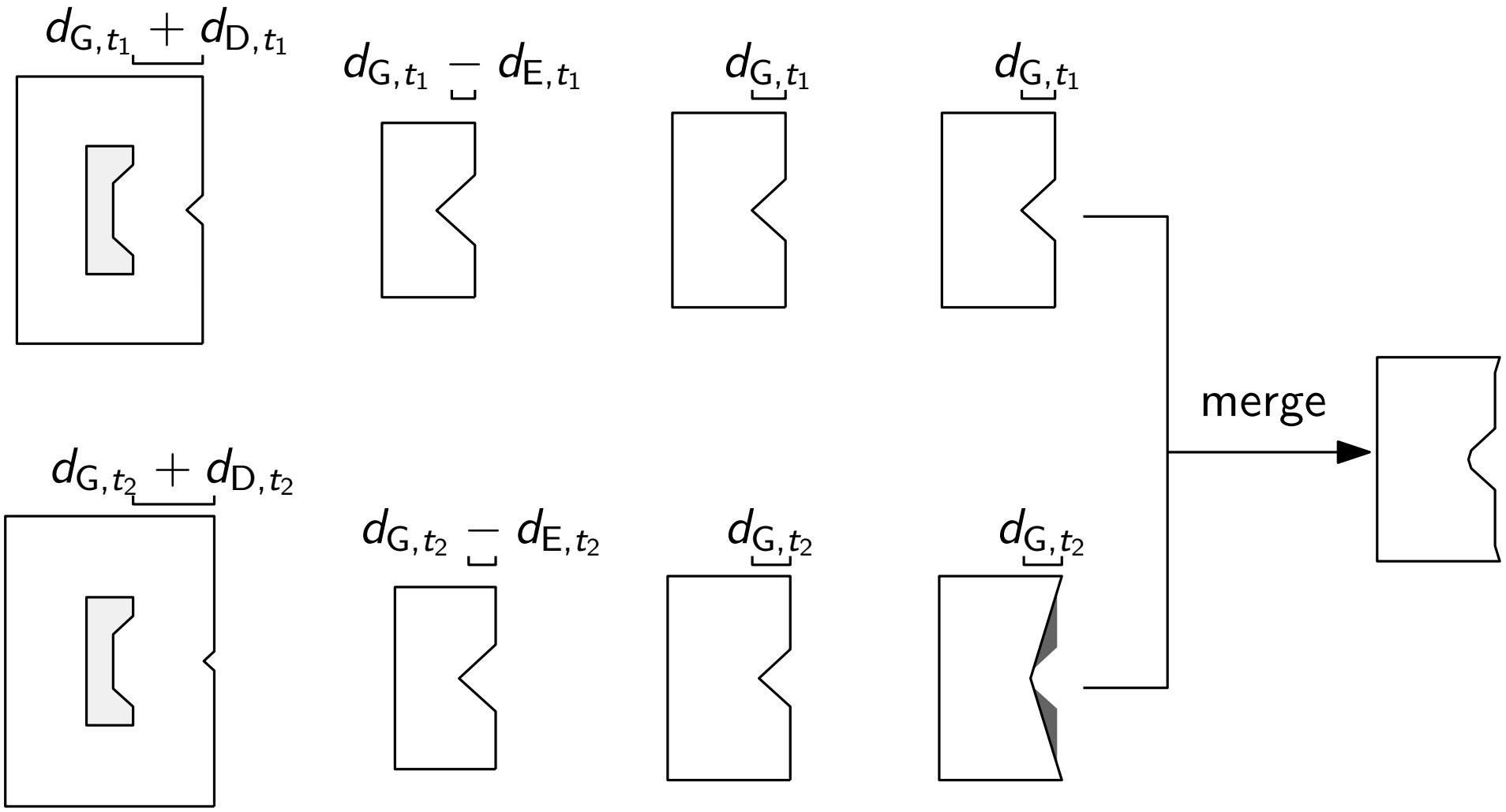
Producing Intermediate-scale maps

A building may shrink because of line simplification

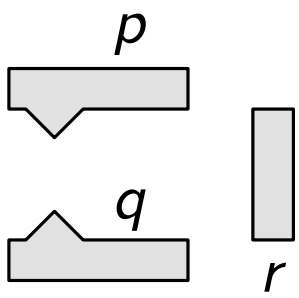


Producing Intermediate-scale maps

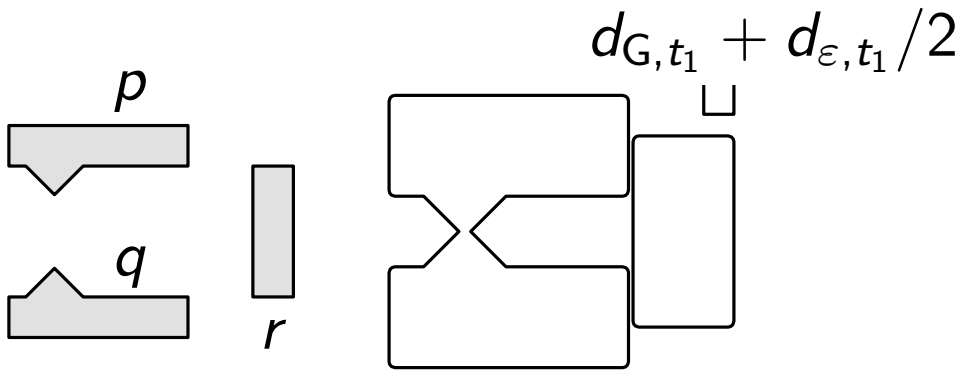
A building may shrink because of line simplification



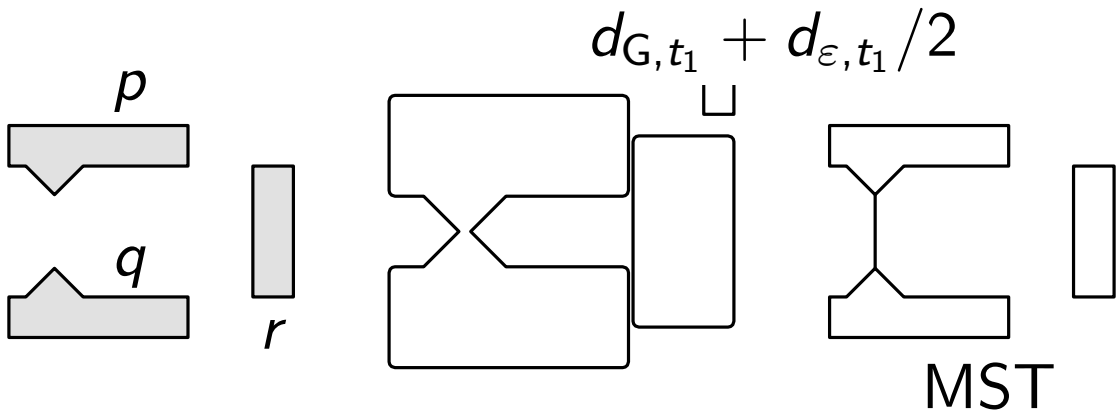
Merging avoiding shriking bridge



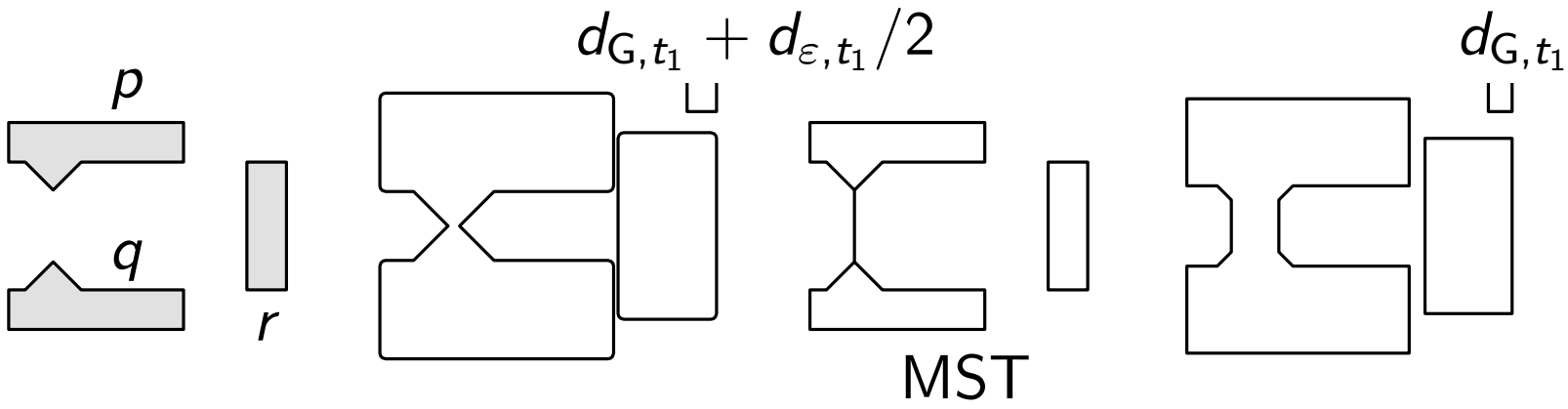
Merging avoiding shriking bridge



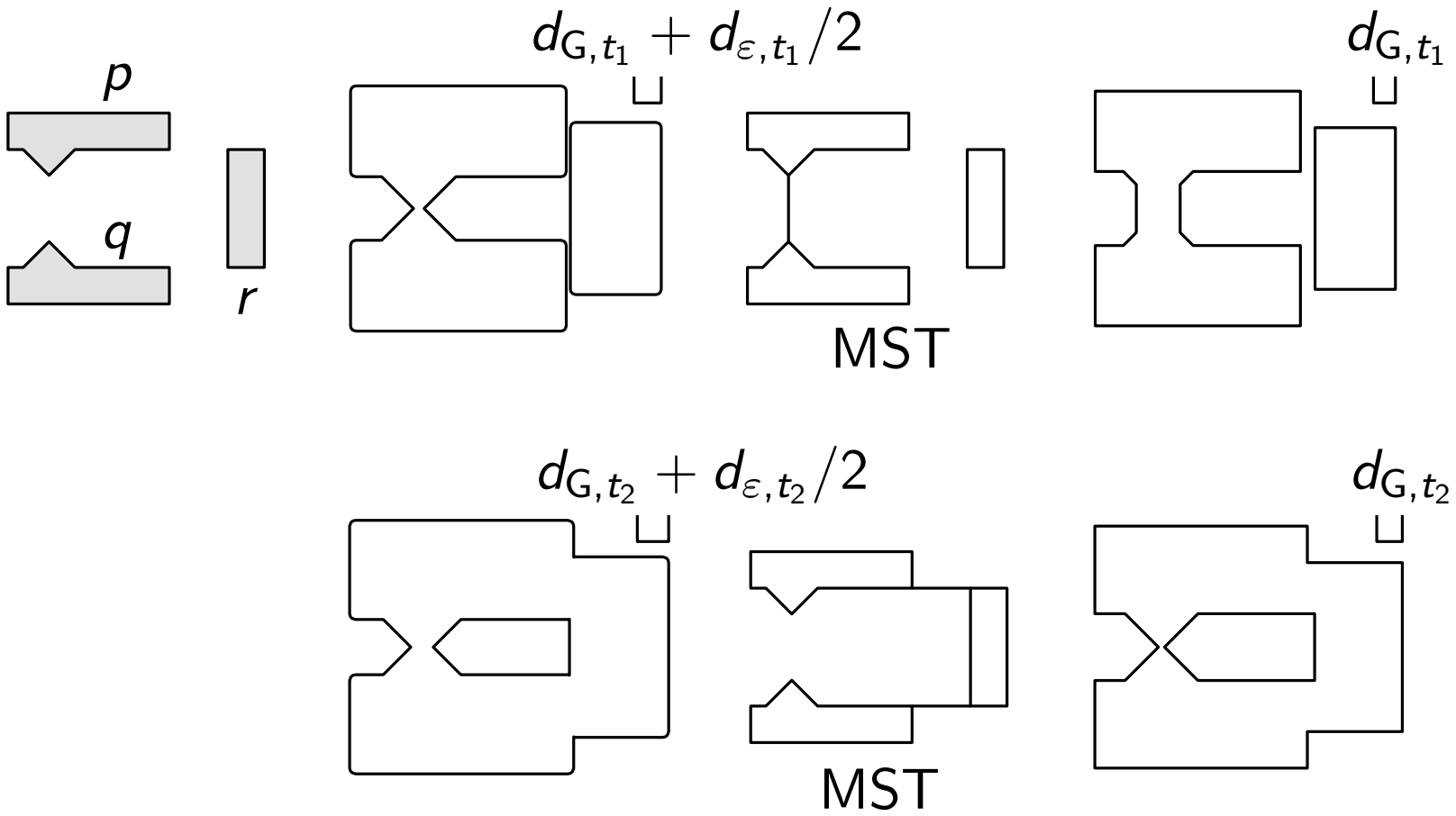
Merging avoiding shriking bridge



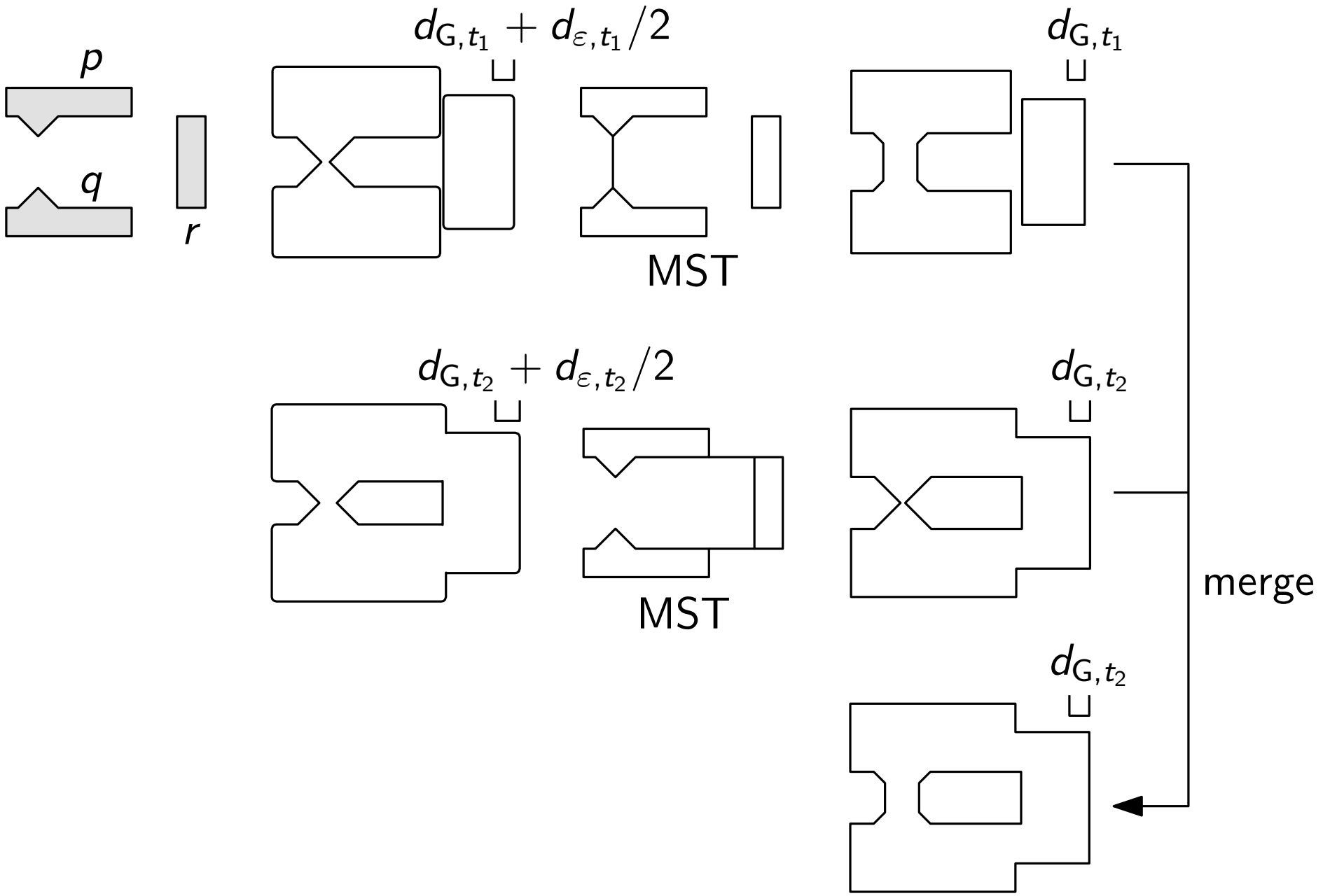
Merging avoiding shriking bridge



Merging avoiding shriking bridge



Merging avoiding shriking bridge



Clipping by Goal Shape

Using the goal shape, at time $t = 1$, to clip the intermediate-scale results

Clipping by Goal Shape

Using the goal shape, at time $t = 1$, to clip the intermediate-scale results

In this way, we avoid that an intermeidate-scale results may exceed the goal shapes

Eliminating Small Buildings and Holes

- For a group building that will be aggregated at time $t = 1$, we eliminate them if their total area at any time t is smaller than a_t.

$$a_t = a \cdot M_t^2, \text{ where } a = 0.16\,\text{mm}^2$$

Eliminating Small Buildings and Holes

- For a group building that will be aggregated at time $t = 1$, we eliminate them if their total area at any time t is smaller than a_t.

$$a_t = a \cdot M_t^2, \text{ where } a = 0.16\,\text{mm}^2$$

- Removing a hole if its area is smaller than $a_{\text{h},t}$

$$a_{\text{h},t} = a_{\text{h}} \cdot M_t^2, \text{ where } a_{\text{h}} = 8\,\text{mm}^2$$

Running Time

- Operations like growing, dilation, eorsion, merge, and clip cost time $O(n^2)$. We may need to iteratively do $O(n)$ times, which increases our running time to $O(n^3)$
- Our version of Imai–Iri simplification algorithm takes time $O(n^3)$.

Outline

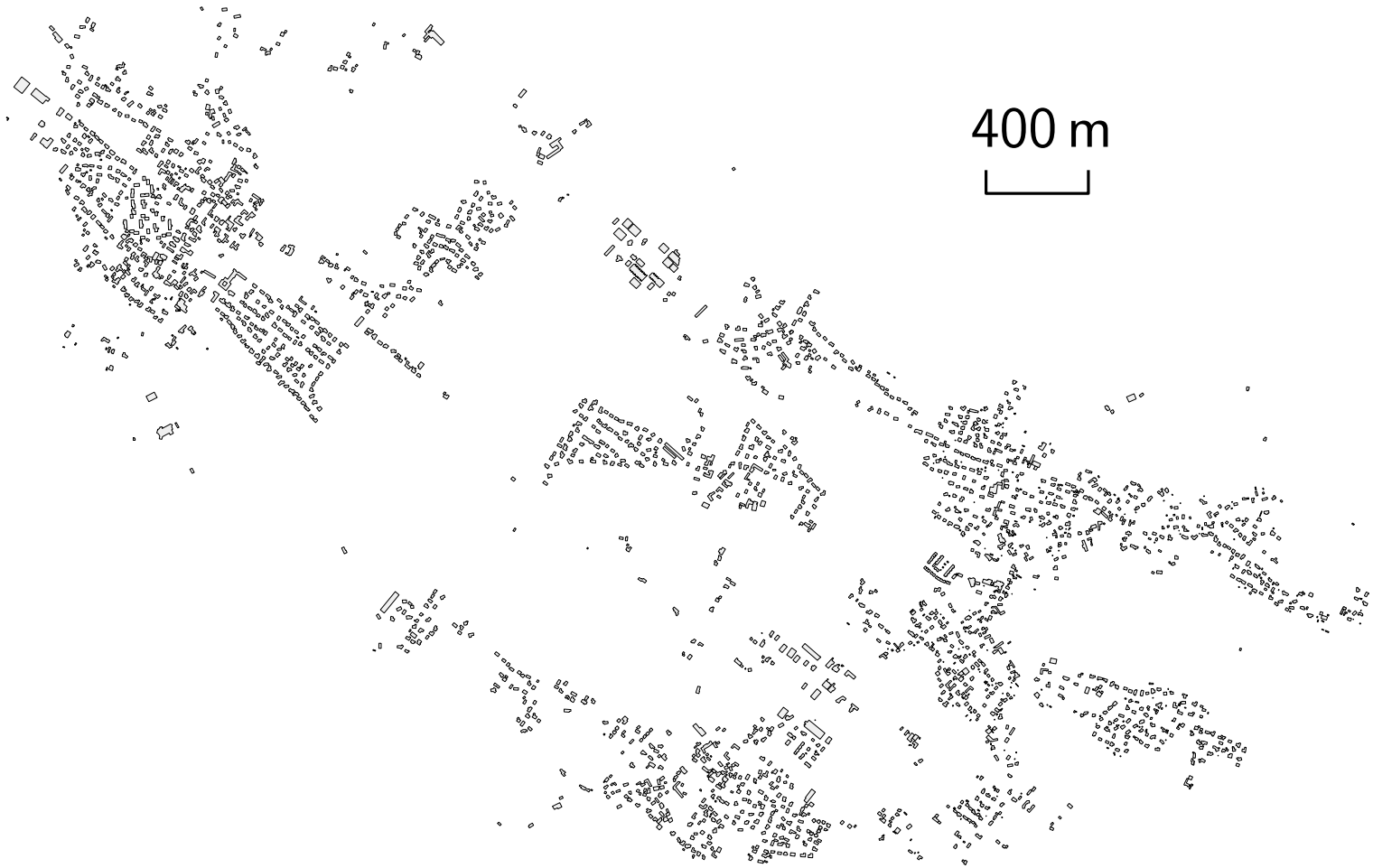
- Introduction
- Methodology
- Case Study
- Concluding Remarks

Case Study

Environment

- C# (using the .NET Framework 4.5)
- ArcObjects SDK 10.4.1
- Windows 7, 3.3 GHz dual core CPU, 8 GB RAM
- Time measure: Stopwatch (a class in C#)
- CLIPPER: buffering, dilation, erosion, and merge

Data



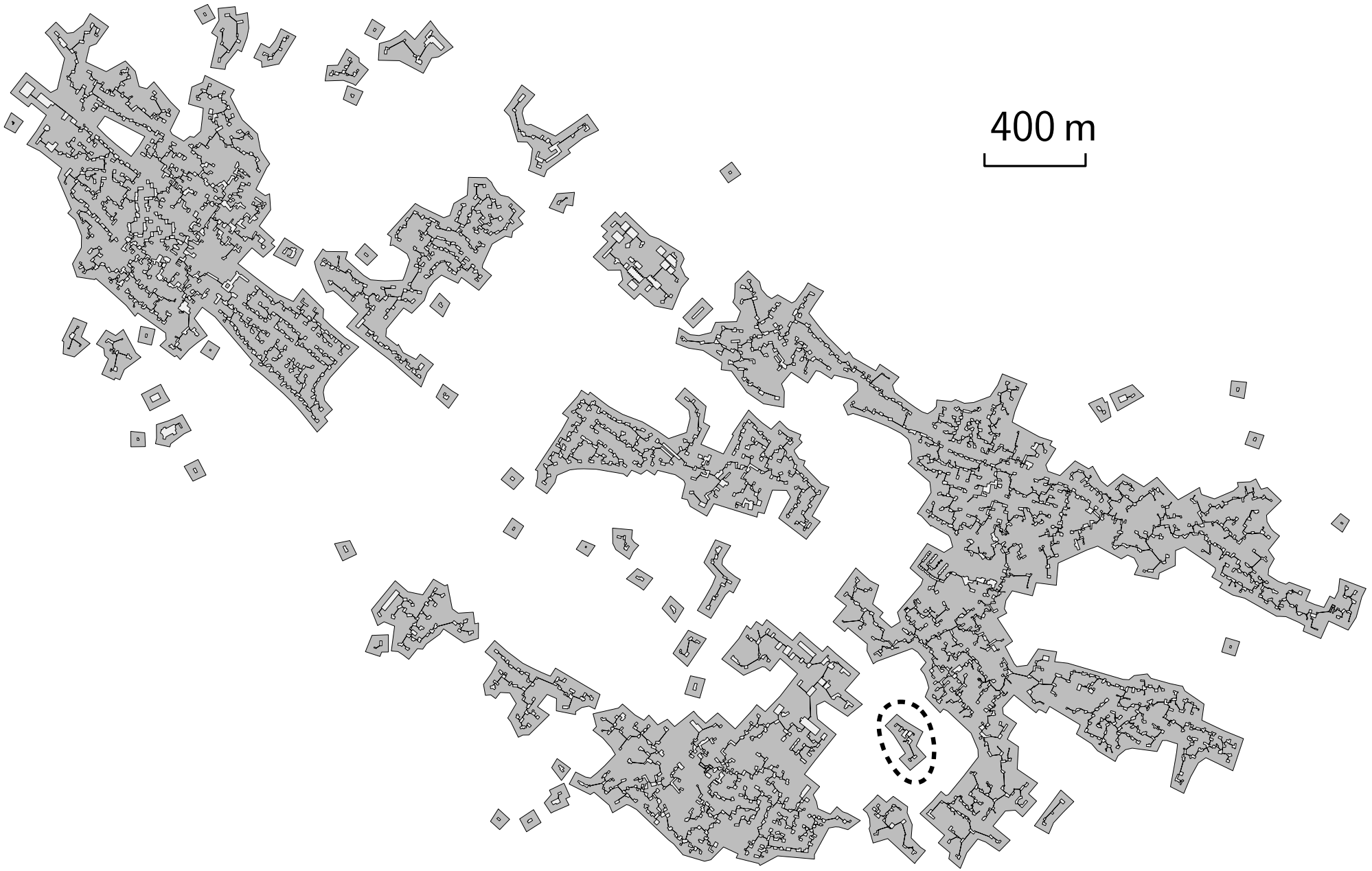
four towns in France,at scale 1 : 15,000,from IGN,
2,590 buildings, in total 19,255 edges,
we set $d_G = 25\,\mathrm{m}$, and thus $d_{\mathrm{D},t} = t \cdot 35\,\mathrm{m}$ and $d_{\mathrm{E},t} = t \cdot 7.5\,\mathrm{m}$

Result

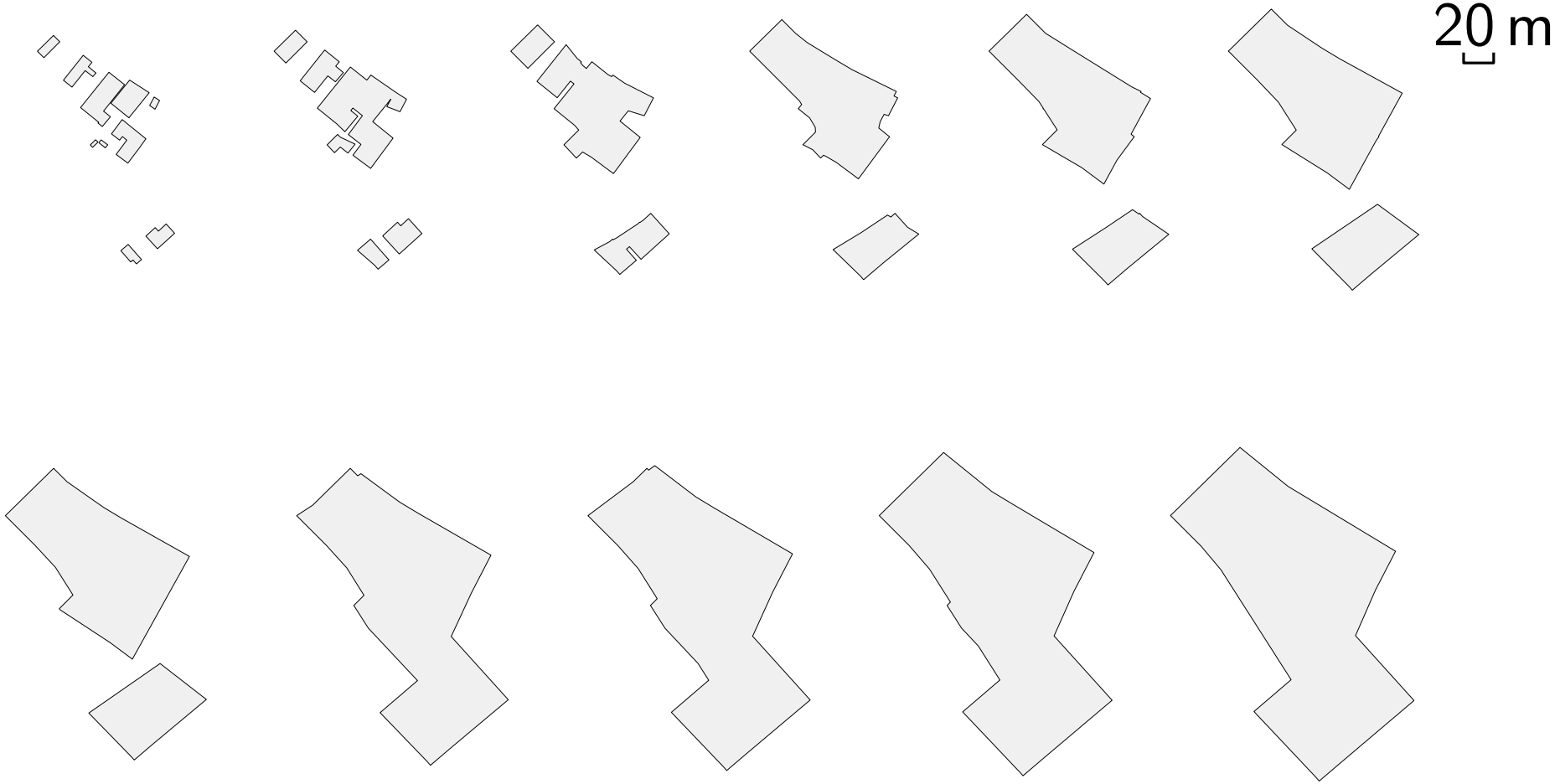
- 93.6 s for computing the goal shapes, where Imai–Iri algorithm simplifies 2,095 edges to 1,102 edges

Result

- 93.6 s for computing the goal shapes, where Imai–Iri algorithm simplifies 2,095 edges to 1,102 edges
- 668.2 s for computing a sequence of 10 maps


400 m

A sequence of maps



Outline

- Our Example Problem
- Methodology
- Case Study
- Concluding Remarks

Concluding Remarks

Advantages of our method:

- The buildings grow continuously and are simplified.
- Right angles of buildings are preserved during growing
- Distances between buildings are larger than a specified threshold.

Concluding Remarks

Advantages of our method:

- The buildings grow continuously and are simplified.
- Right angles of buildings are preserved during growing
- Distances between buildings are larger than a specified threshold.

Open problems:

- For a given map and scale, how many buildings should be kept after generalization?
- Again, how much total area of buildings should be kept? What about the total number of edges?
- How to design a meaningful user study to evaluate results?

Concluding Remarks

Advantages of our method:

- The buildings grow continuously and are simplified.
- Right angles of buildings are preserved during growing
- Distances between buildings are larger than a specified threshold.

Open problems:

- For a given map and scale, how many buildings should be kept after generalization?
- Again, how much total area of buildings should be kept? What about the total number of edges?
- How to design a meaningful user study to evaluate results?

Thank you!

Looking for a postdoc position!