Verwaltung sehr großer volumetrischer Datensätze unter Verwendung der hierarchischen Lauflängenkodierung

Holger Gerth

Technische Universität Chemnitz
Fakultät für Informatik

1. Oktober 2007

Übersicht

Motivation

- moderne Aufnahmetechnologien für volumetrische 3D Daten ermöglichen sehr hohe Auflösungen

Motivation

- moderne Aufnahmetechnologien für volumetrische 3D Daten ermöglichen sehr hohe Auflösungen $\Longrightarrow$ benötigen daher sehr viel Speicherplatz

Motivation

- moderne Aufnahmetechnologien für volumetrische 3D Daten ermöglichen sehr hohe Auflösungen $\Longrightarrow$ benötigen daher sehr viel Speicherplatz
- nicht alle Voxel enthalten brauchbare Informationen

Motivation

- moderne Aufnahmetechnologien für volumetrische 3D Daten ermöglichen sehr hohe Auflösungen $\Longrightarrow$ benötigen daher sehr viel Speicherplatz
- nicht alle Voxel enthalten brauchbare Informationen
- oft werden die Daten nur in einer binarisierten Form benötigt

Ziel der Arbeit

Eine Datenstruktur entwerfen, die

- speichereffizient ist,

Ziel der Arbeit

Eine Datenstruktur entwerfen, die

- speichereffizient ist,
- einen schnellen Zugriff auf die Daten ermöglicht und

Ziel der Arbeit

Eine Datenstruktur entwerfen, die

- speichereffizient ist,
- einen schnellen Zugriff auf die Daten ermöglicht und
- eine effiziente Implementierung von Bildverarbeitungsalgorithmen ermöglicht.

Ziel der Arbeit

Eine Datenstruktur entwerfen, die

- speichereffizient ist,
- einen schnellen Zugriff auf die Daten ermöglicht und
- eine effiziente Implementierung von Bildverarbeitungsalgorithmen ermöglicht.

Grundlage

Hierarchische lauflängenkodierte Niveaumenge nach Housten u.a.

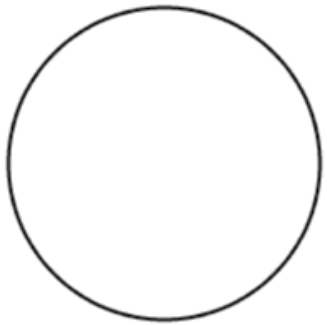
Niveaumenge (engl. Level Set)

- in einem n-dimensionalen Raum wird der $(n-1)$-dimensionale Rand Γ eines n-dimensionalen Objekts als Nullstellenmenge einer Hilsfunktion φ beschrieben
- positive Werte auf der einen Seite von Γ (außerhalb des Objekts)
- negative Werte auf der anderen Seite (innerhalb des Objekts)

Niveaumenge (engl. Level Set)

- in einem n-dimensionalen Raum wird der $(n-1)$-dimensionale Rand Γ eines n-dimensionalen Objekts als Nullstellenmenge einer Hilsfunktion φ beschrieben
- positive Werte auf der einen Seite von Γ (außerhalb des Objekts)
- negative Werte auf der anderen Seite (innerhalb des Objekts)

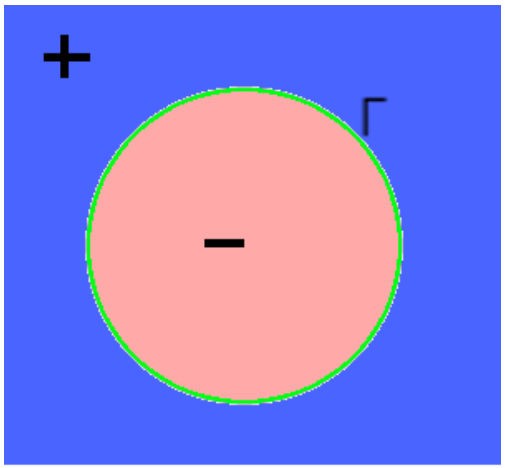
Beispiel: Kreis im 2D $\quad \Gamma \longrightarrow (x - x_M)^2 + (y - y_M)^2 = r^2$
φ sei vorzeichenbehaftete Abstandsfunktion



Niveaumenge (engl. Level Set)

- in einem n-dimensionalen Raum wird der $(n-1)$-dimensionale Rand Γ eines n-dimensionalen Objekts als Nullstellenmenge einer Hilsfunktion φ beschrieben
- positive Werte auf der einen Seite von Γ (außerhalb des Objekts)
- negative Werte auf der anderen Seite (innerhalb des Objekts)

Beispiel: Kreis im 2D $\quad \Gamma \longrightarrow (x - x_M)^2 + (y - y_M)^2 = r^2$
φ sei vorzeichenbehaftete Abstandsfunktion



Lauflängenkodierung

- Serie aufeinanderfolgender Daten der selben Art wird komprimiert gespeichert

Beispiel:

aaaaaaaazzzzzzBBBBBkkkkkhhh

Lauflängenkodierung

- Serie aufeinanderfolgender Daten der selben Art wird komprimiert gespeichert
- es wird nur Typ, Startpunkt und Anzahl der Daten (Länge des Laufs) gespeichert

Beispiel:

aaaaaaaazzzzzzBBBBBkkkkkhhh

Lauflängenkodierung

- Serie aufeinanderfolgender Daten der selben Art wird komprimiert gespeichert
- es wird nur Typ, Startpunkt und Anzahl der Daten (Länge des Laufs) gespeichert

Beispiel:

aaaaaaaazzzzzzBBBBBkkkkkhhh

Lauflängenkodierung

- Serie aufeinanderfolgender Daten der selben Art wird komprimiert gespeichert
- es wird nur Typ, Startpunkt und Anzahl der Daten (Länge des Laufs) gespeichert

Beispiel:

aaaaaaaazzzzzzBBBBBkkkkkhhh

a8z6B5k4h3

Aufbau der H-LLK Niveaumenge

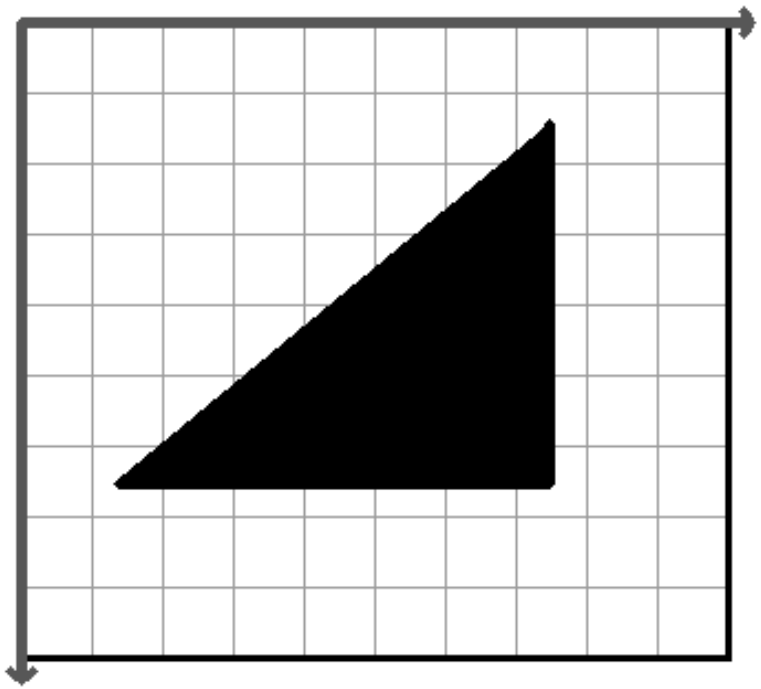
- grundlegender Lösungsvorschlag: dimensionsweise Anwendung der Lauflängenkodierung, beginnend mit der niedrigsten Dimension

Aufbau der H-LLK Niveaumenge

- grundlegender Lösungsvorschlag: dimensionsweise Anwendung der Lauflängenkodierung, beginnend mit der niedrigsten Dimension
- Klassifizierung der Voxel in positiv, negativ und definiert

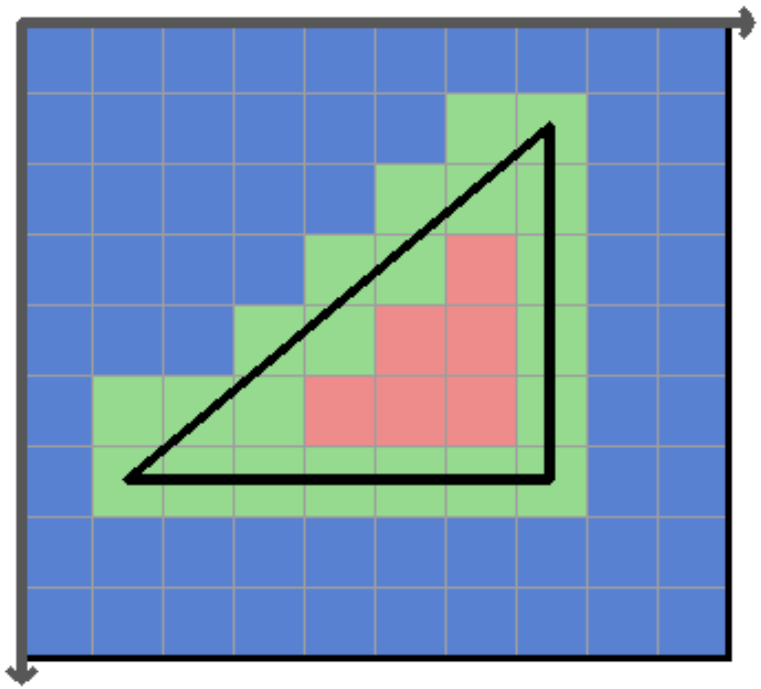
Aufbau der H-LLK Niveaumenge

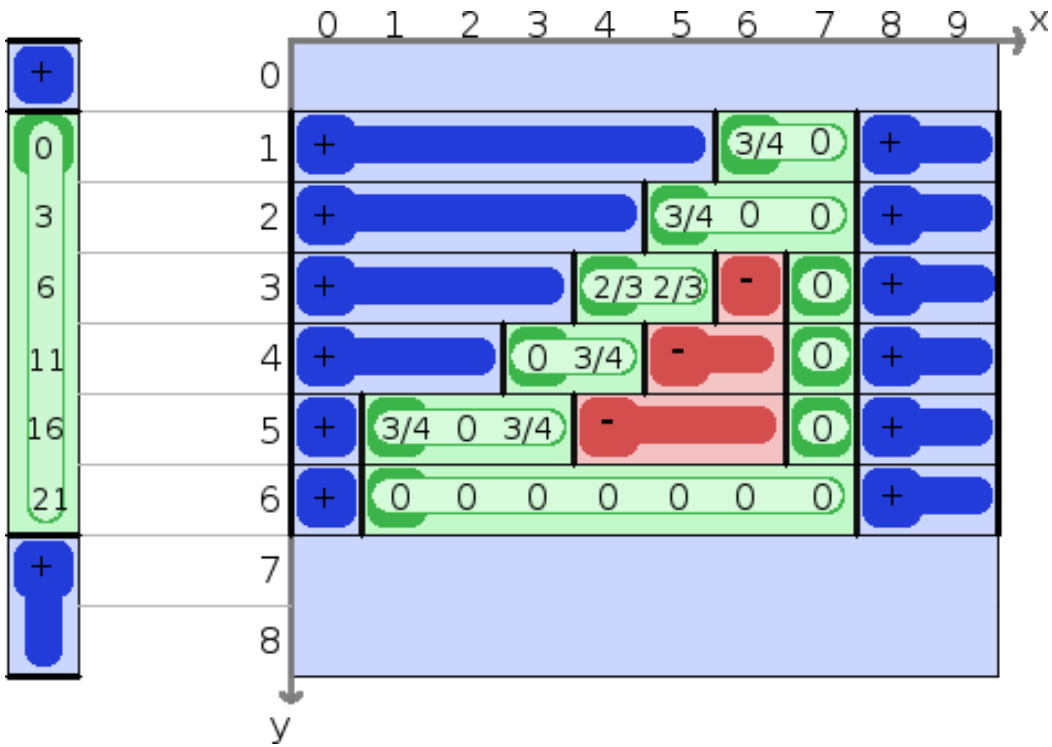
- grundlegender Lösungsvorschlag: dimensionsweise Anwendung der Lauflängenkodierung, beginnend mit der niedrigsten Dimension
- Klassifizierung der Voxel in positiv, negativ und definiert

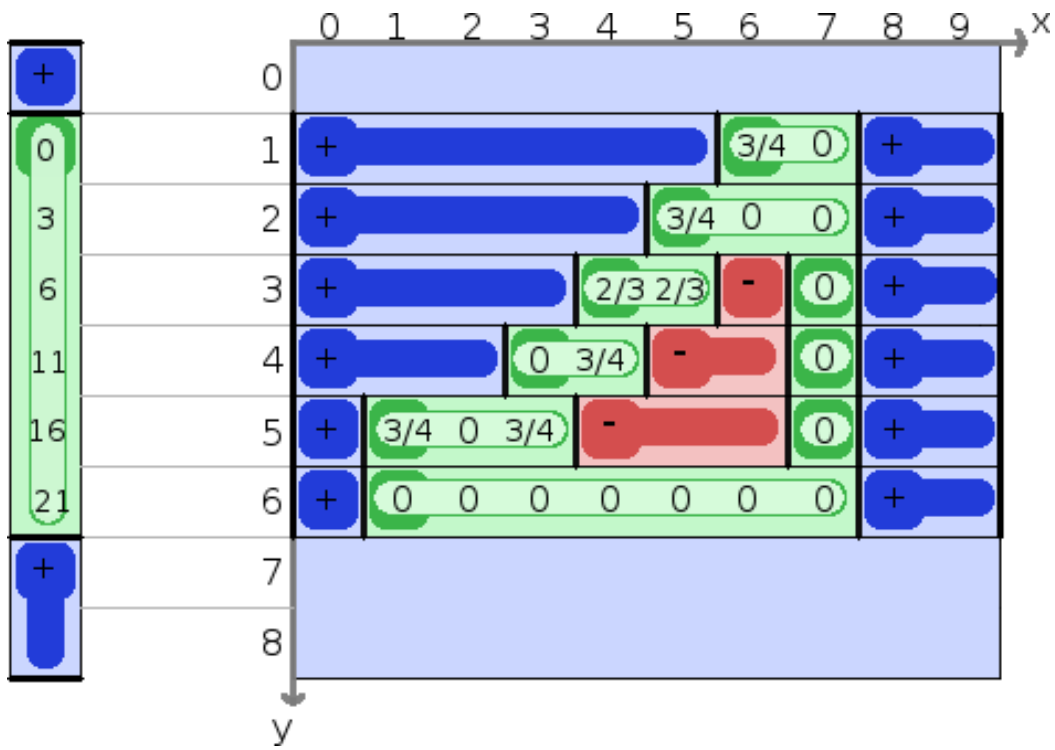


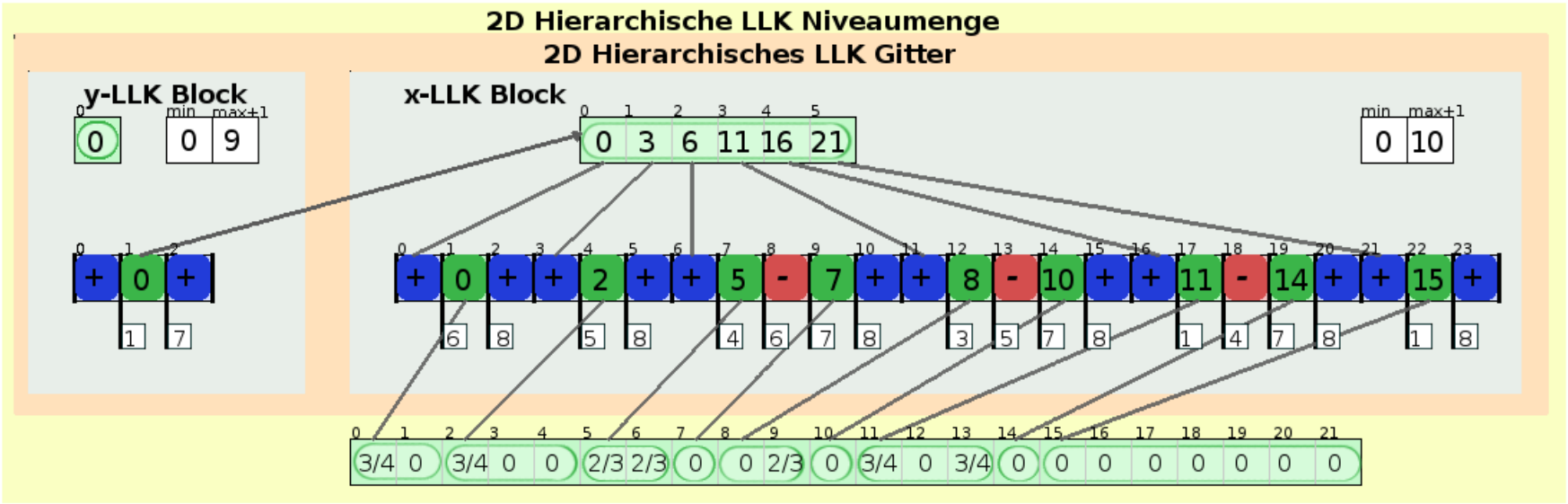
Aufbau der H-LLK Niveaumenge

- grundlegender Lösungsvorschlag: dimensionsweise Anwendung der Lauflängenkodierung, beginnend mit der niedrigsten Dimension
- Klassifizierung der Voxel in positiv, negativ und definiert




0 1 2 3 4 5 6 7 8 9 x
0
1
2
3
4
5
6
7
8
y
+
0
3
6
11
16
21
+
3/4 0
3/4 0 0
2/3 2/3
-
0
0 3/4
3/4 0 3/4
0 0 0 0 0 0 0


0 1 2 3 4 5 6 7 8 9 x
0 3 6 11 16 21
0 1 2 3 4 5 6 7 8 y
3/4 0
3/4 0 0
2/3 2/3
0 3/4
3/4 0 3/4
0 0 0 0 0 0 0


2D Hierarchische LLK Niveaumenge
2D Hierarchisches LLK Gitter
y-LLK Block
min max+1
0 9
x-LLK Block
0 3 6 11 16 21
min max+1
0 10
Start-Indizes & Ausdehng.
Läufe
Unterbrechg
Werte
3/4 0 3/4 0 0 2/3 2/3 0 0 2/3 0 3/4 0 3/4 0 0 0 0 0 0 0 0

Änderungen an der Datenstruktur

- kein Werte-Array, da Daten binarisiert

Änderungen an der Datenstruktur

- kein Werte-Array, da Daten binarisiert
- Klassifizierung der Läufe über deren Indizes, -1 für negativ, MaxInt für positiv

Änderungen an der Datenstruktur

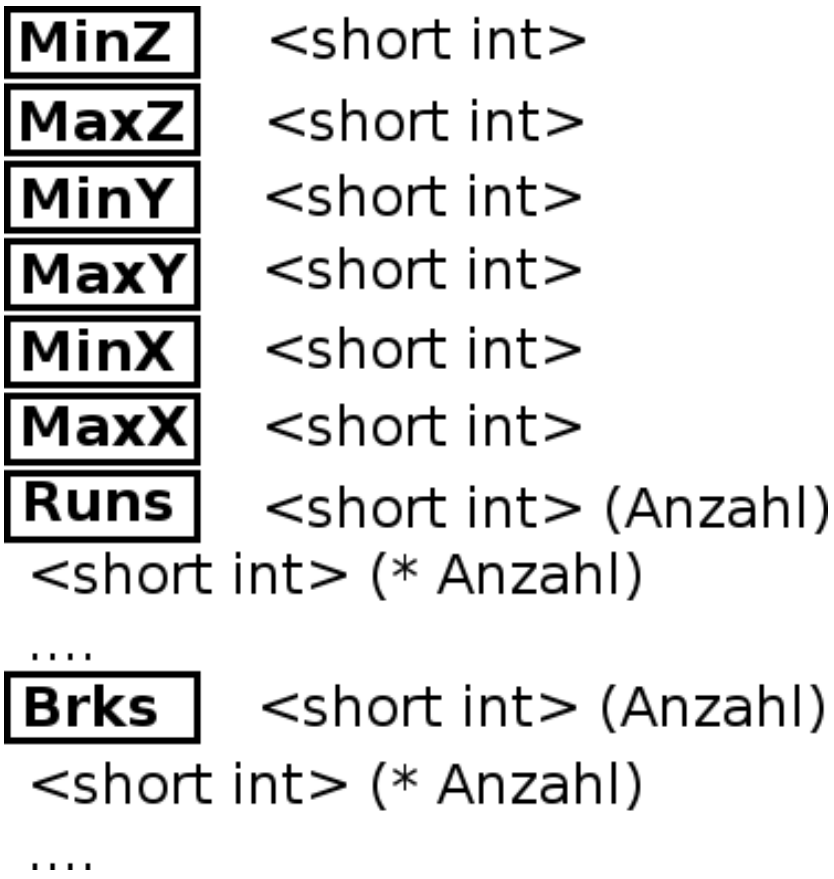
- kein Werte-Array, da Daten binarisiert
- Klassifizierung der Läufe über deren Indizes, -1 für negativ, MaxInt für positiv
- Dateiformat zur Speicherung auf Datenträgern

Änderungen an der Datenstruktur

- kein Werte-Array, da Daten binarisiert
- Klassifizierung der Läufe über deren Indizes, -1 für negativ, MaxInt für positiv
- Dateiformat zur Speicherung auf Datenträgern
- Aufteilung des Datensatzes in Teilstücke

Aufbau der Dateien

Abbildung: Hauptdatei



Aufbau der Dateien

Abbildung: Hauptdatei

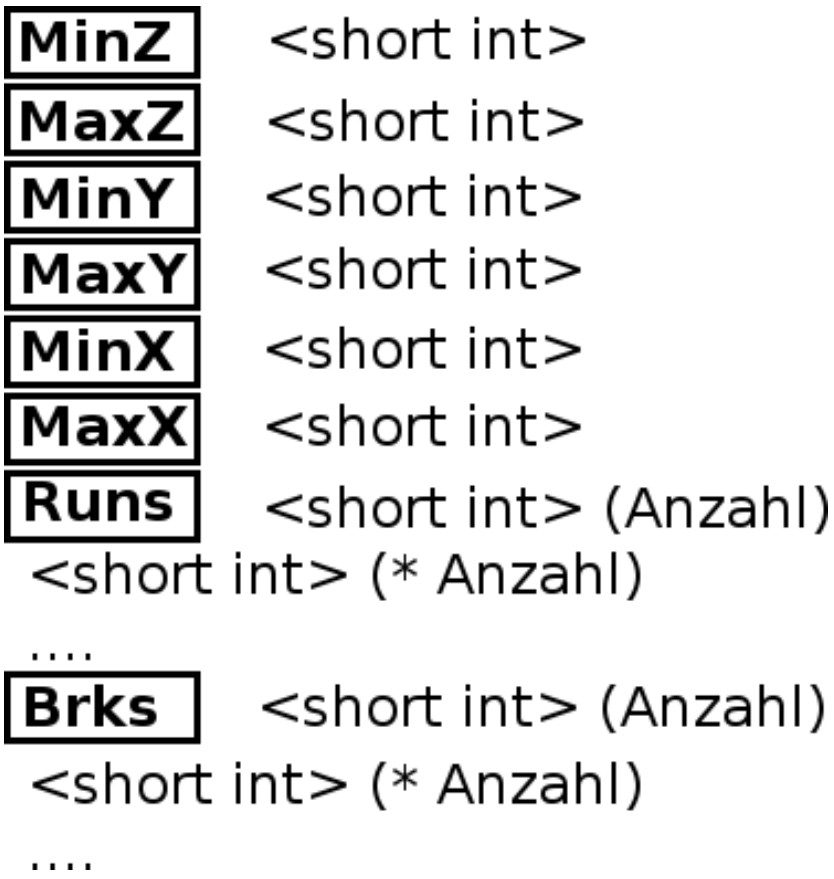
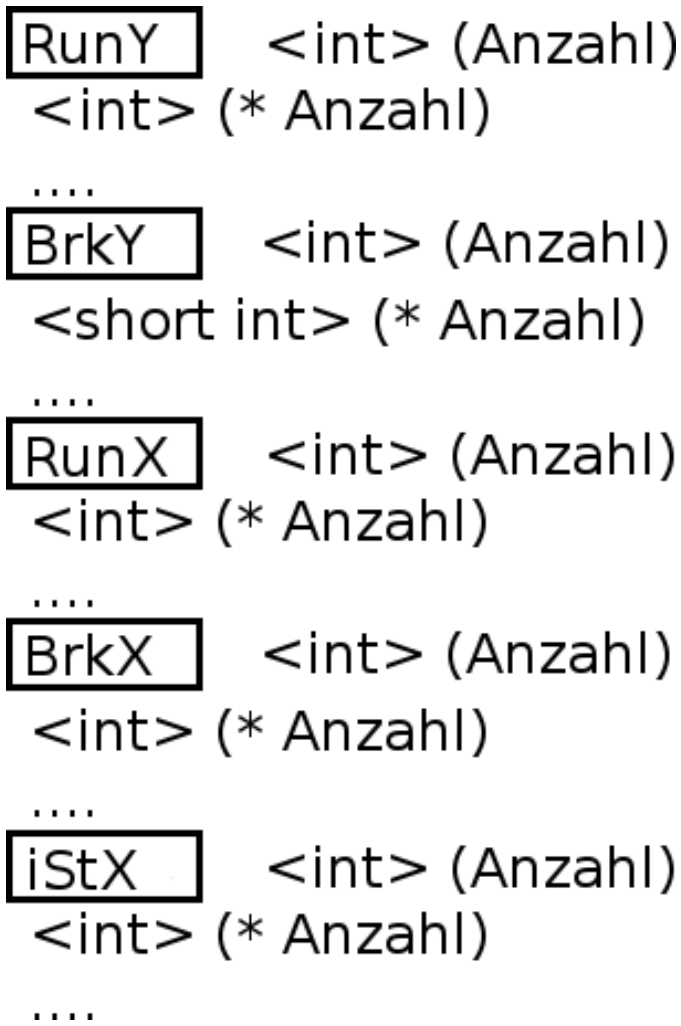


Abbildung: Segment-Datei



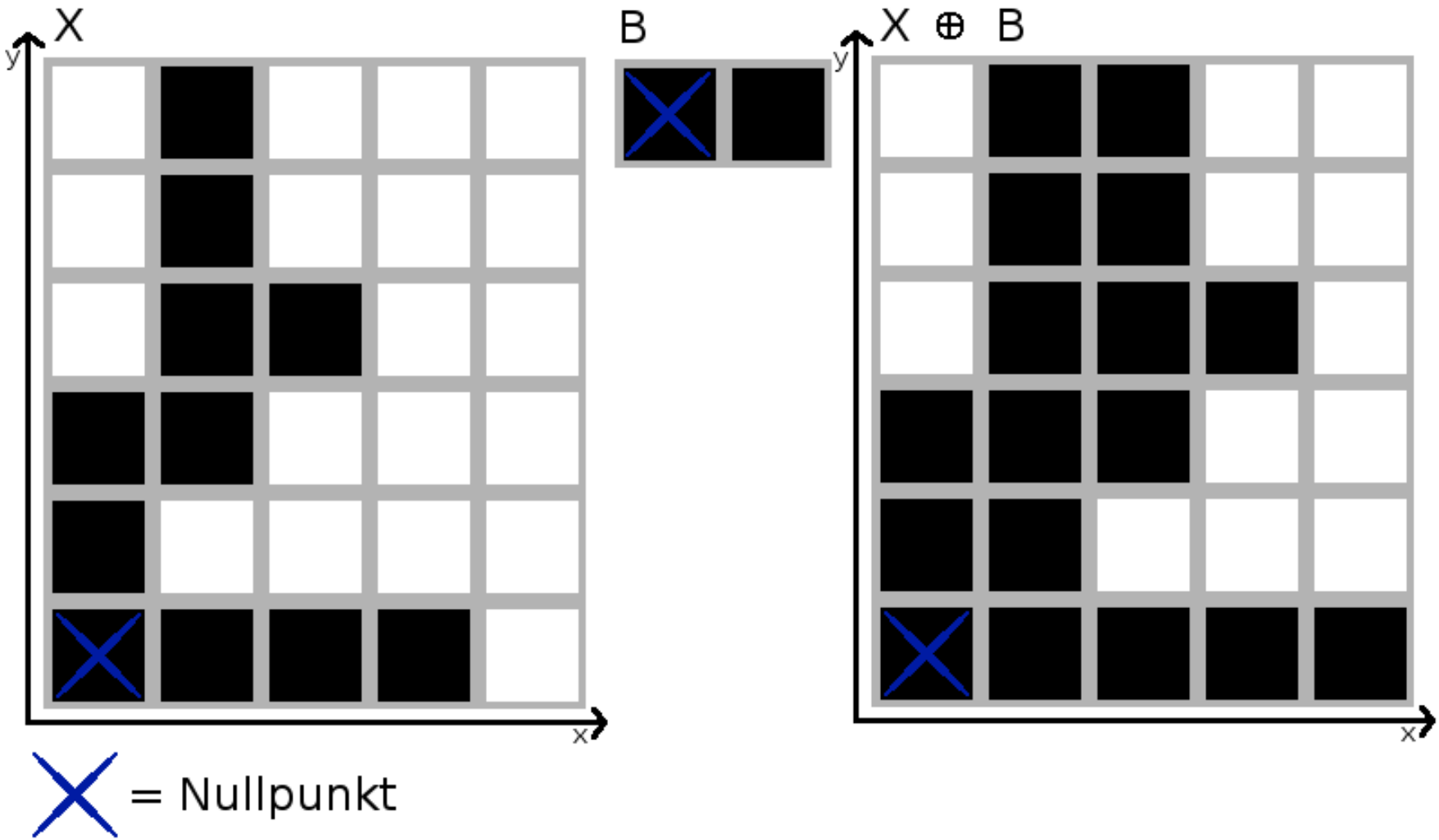
Übersicht

- Morphologische Operatoren
  - Dilation
  - Erosion
  - Opening
  - Closing

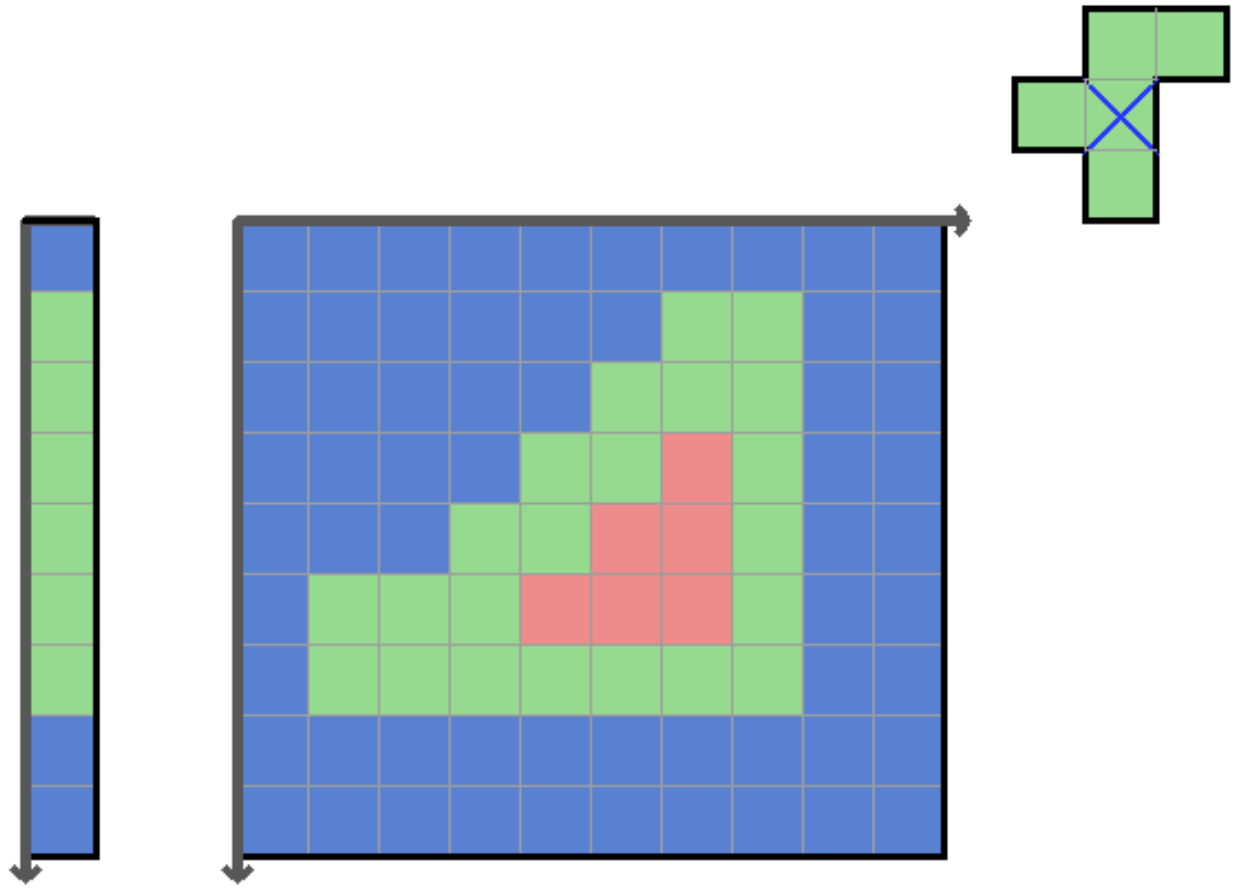
Übersicht

- Morphologische Operatoren
  - Dilation
  - Erosion
  - Opening
  - Closing
- Skelettierung / Ausdünnung

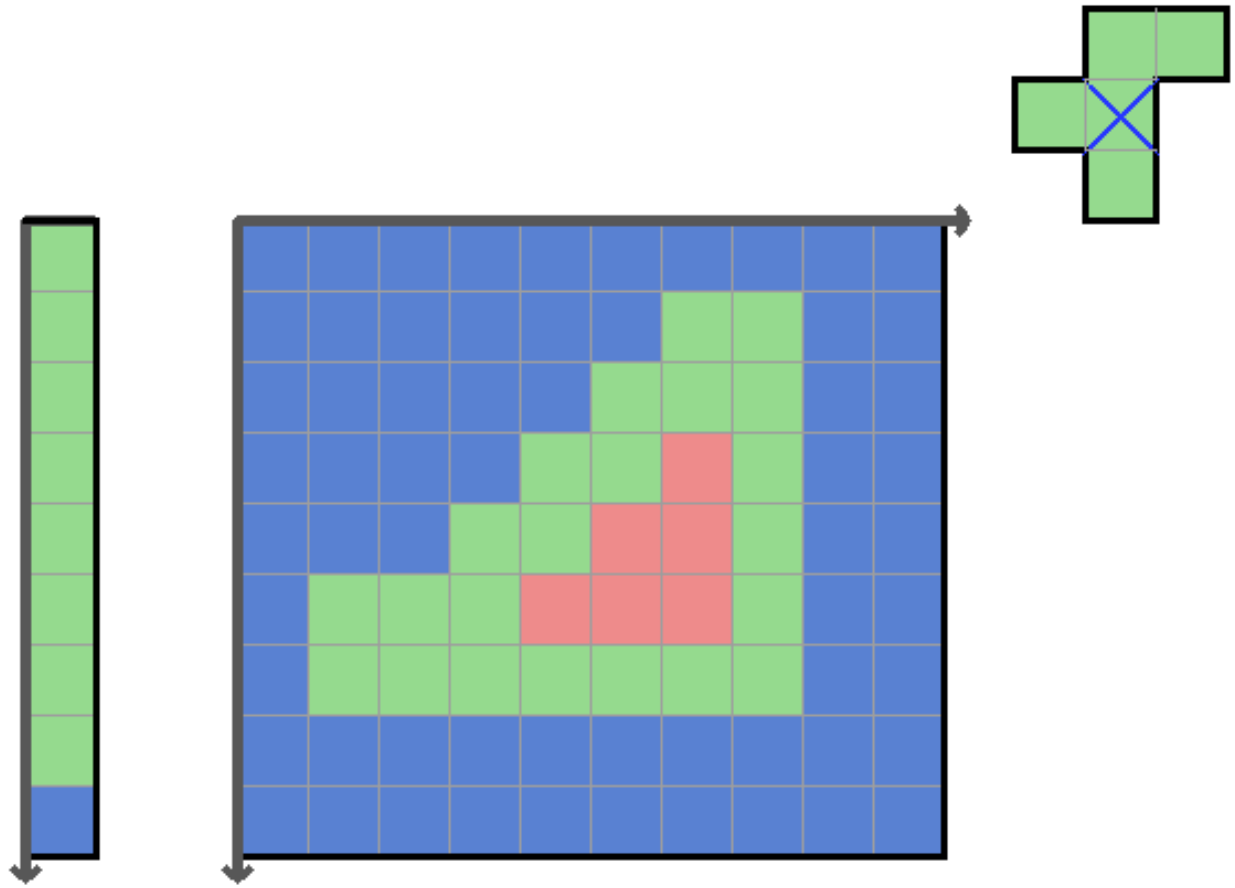
Dilation



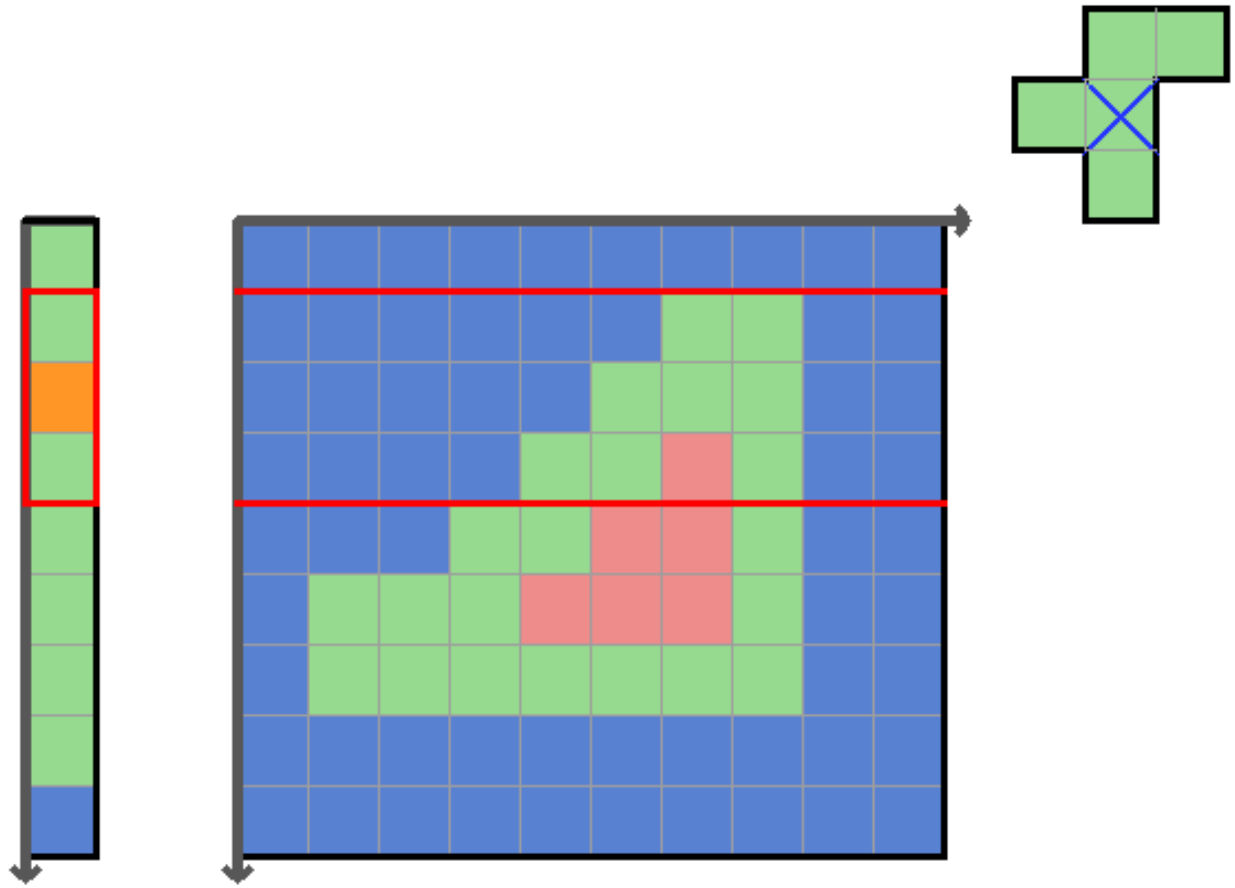
Dilation



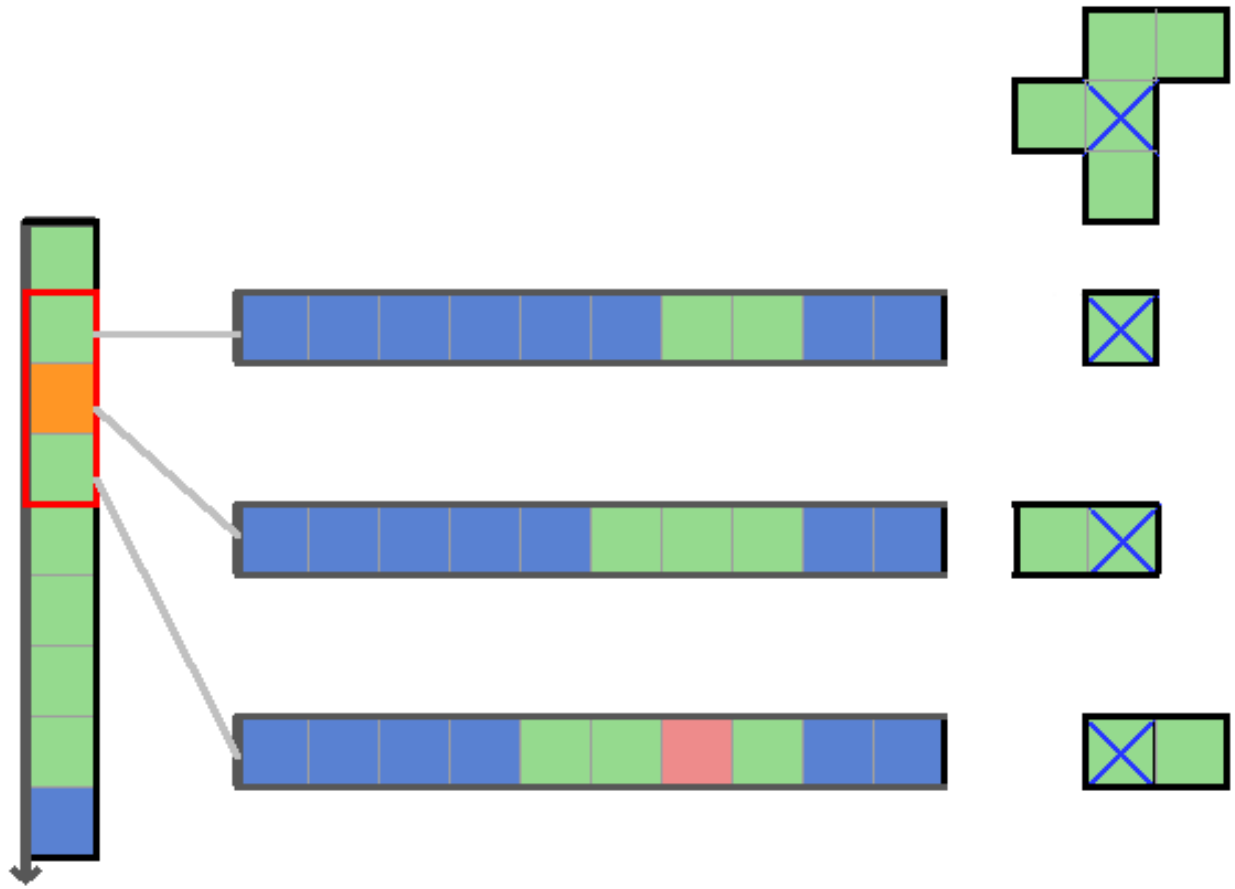
Dilation



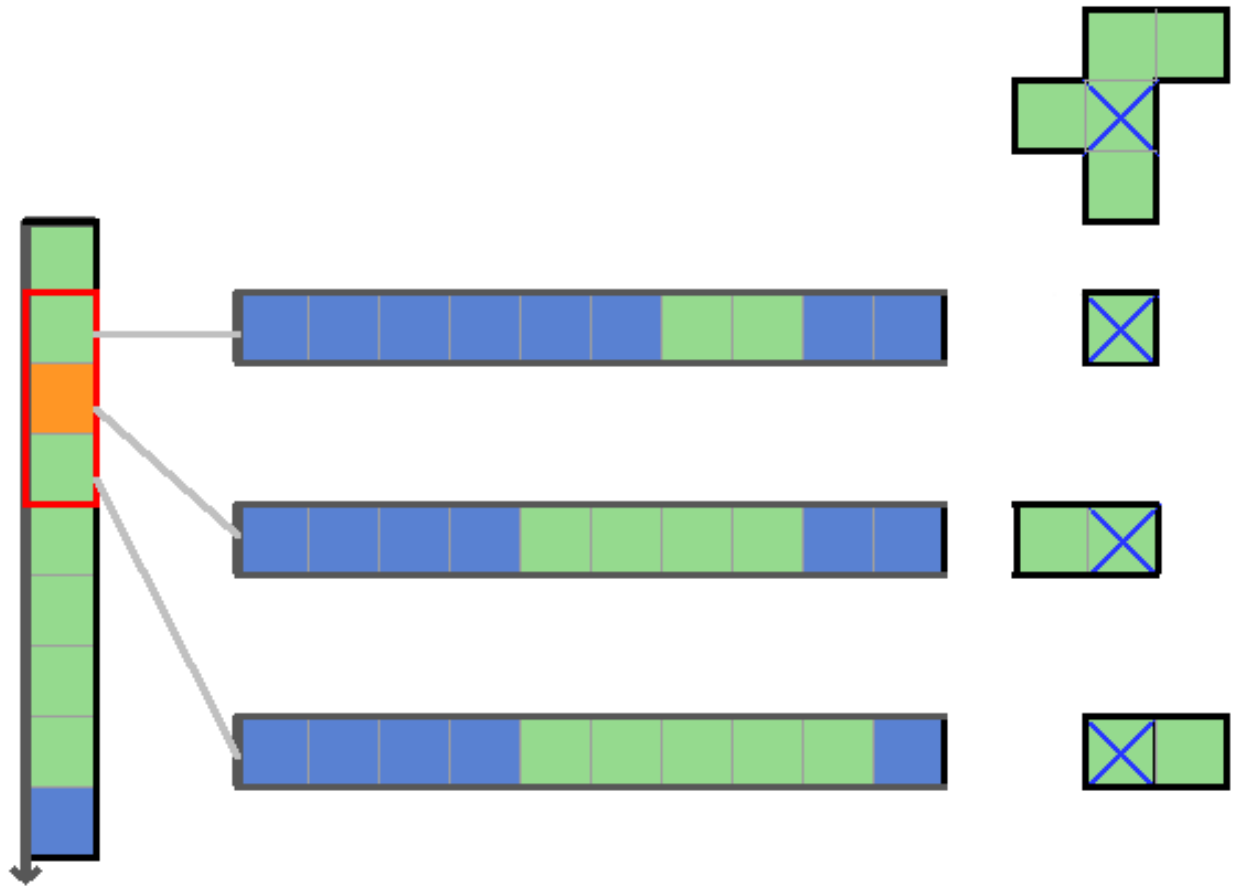
Dilation



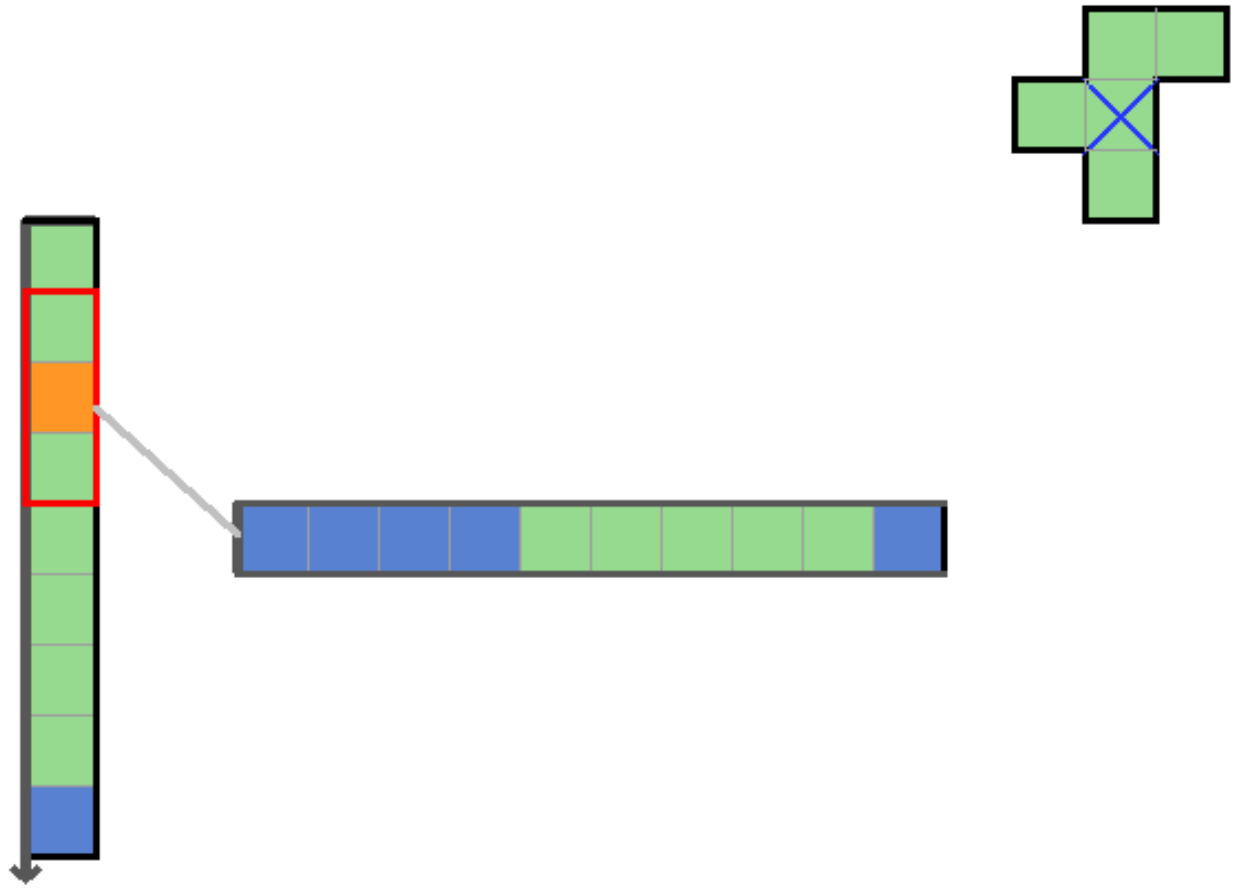
Dilation



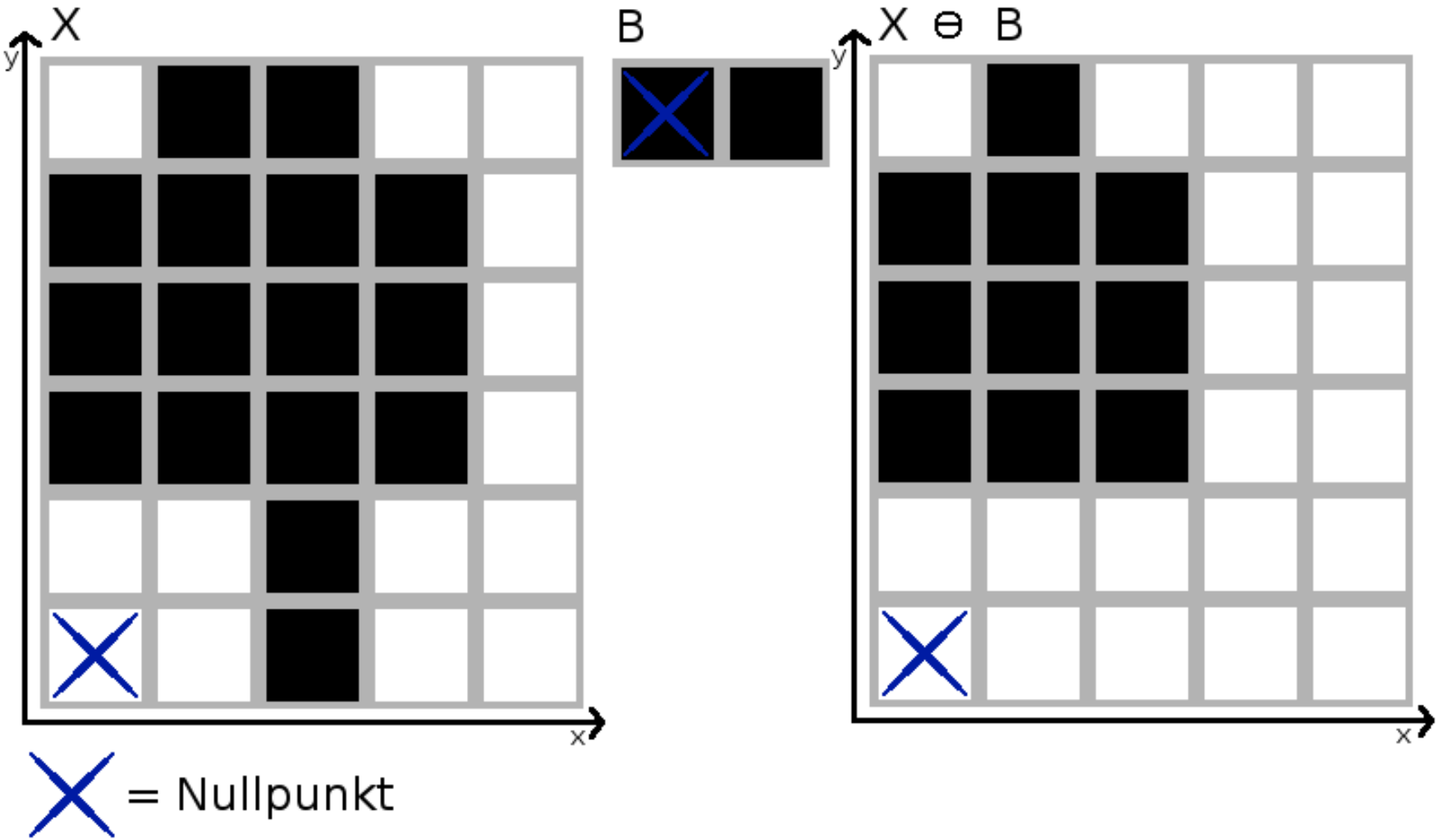
Dilation



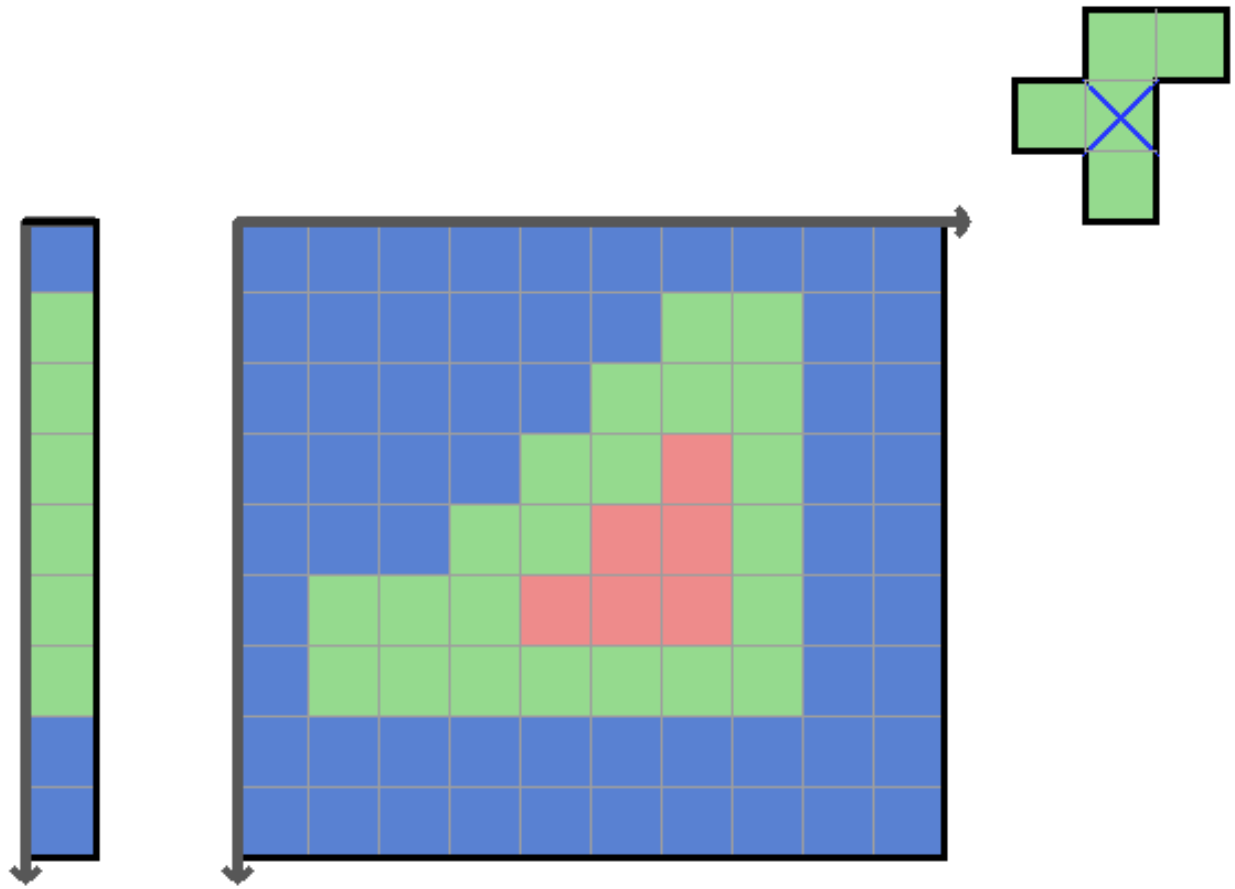
Dilation



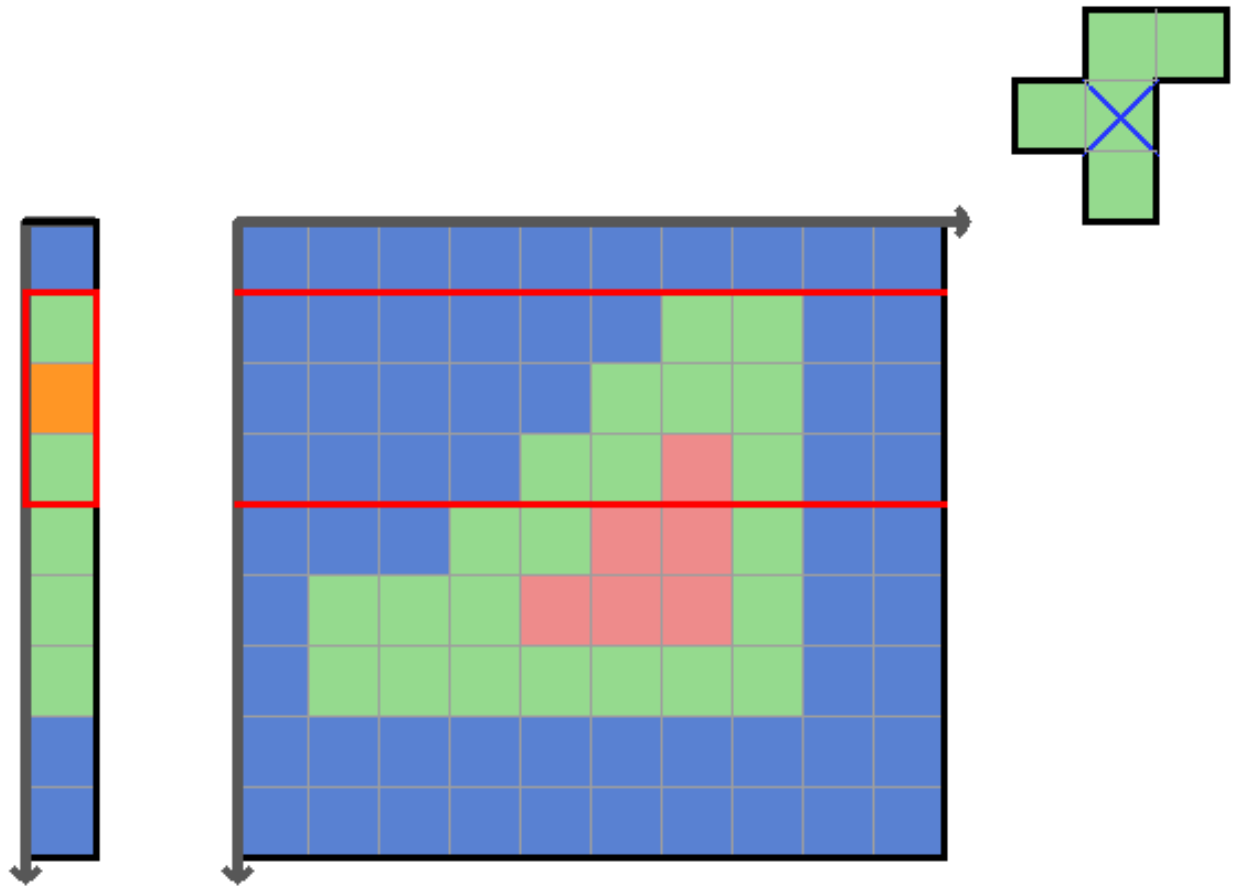
Erosion



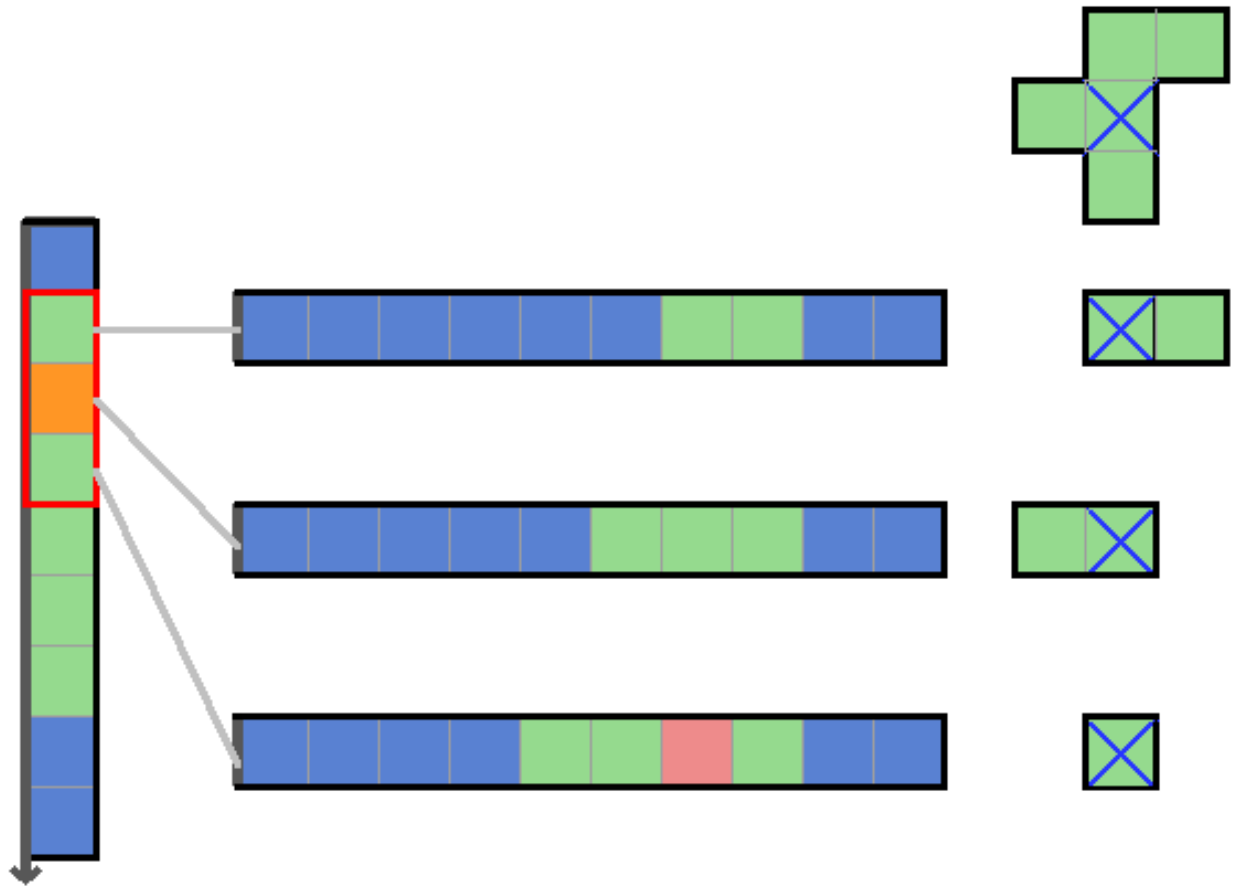
Erosion



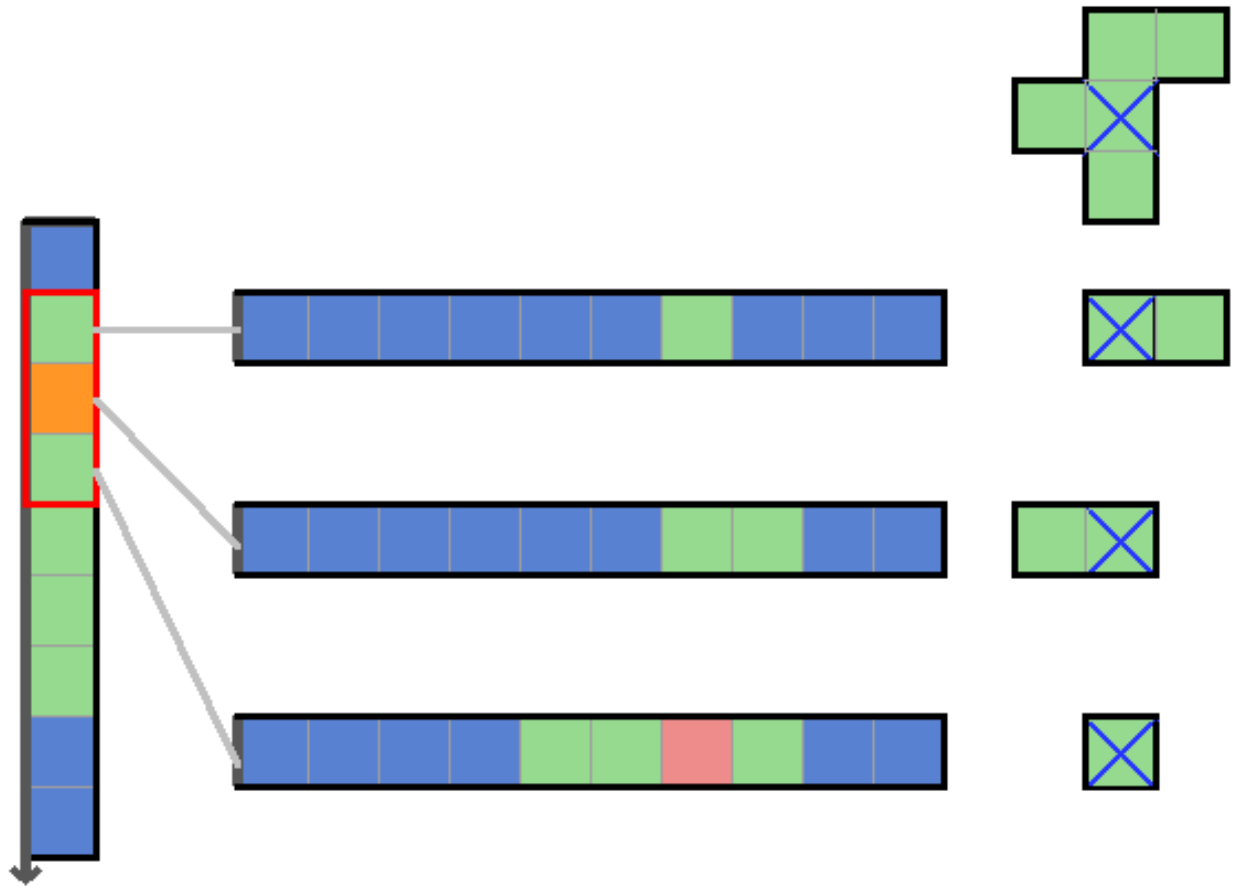
Erosion



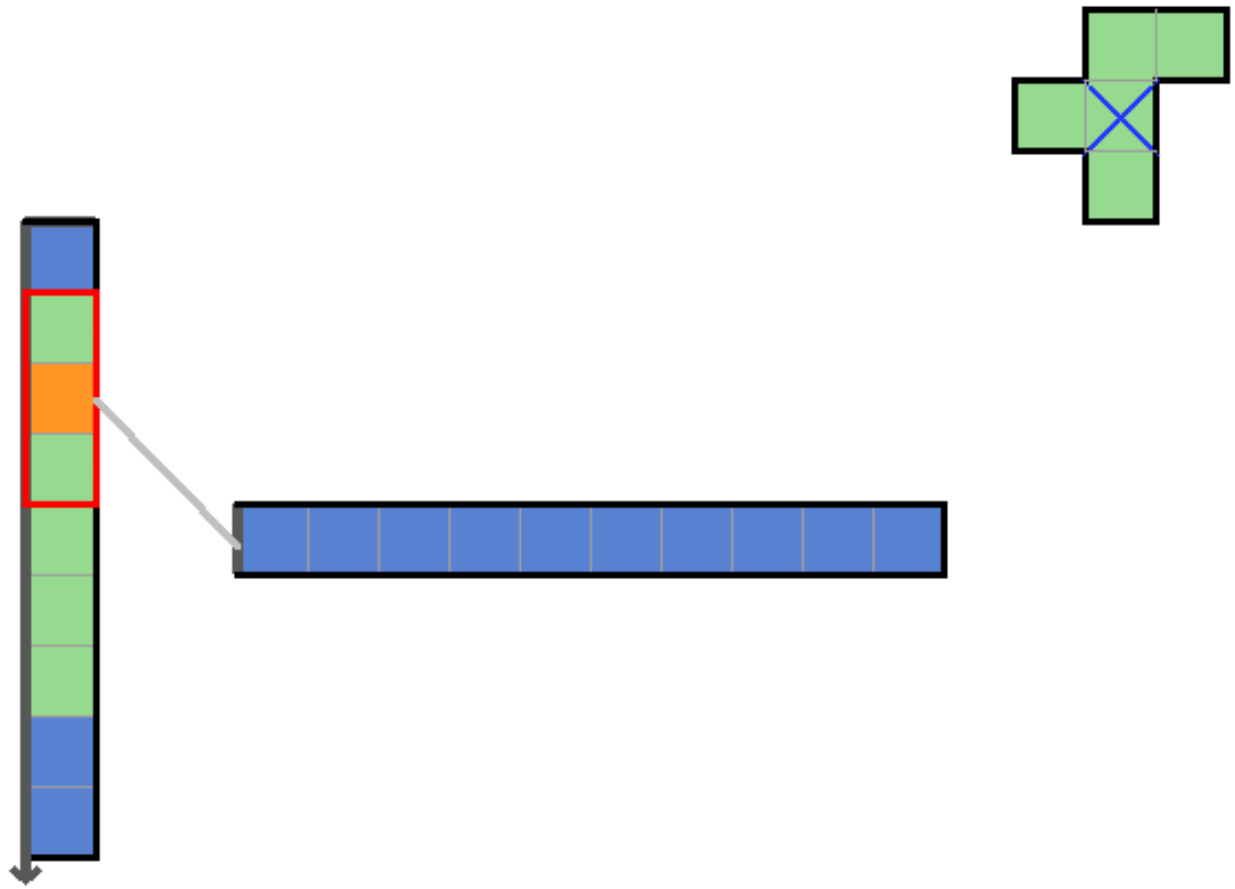
Erosion



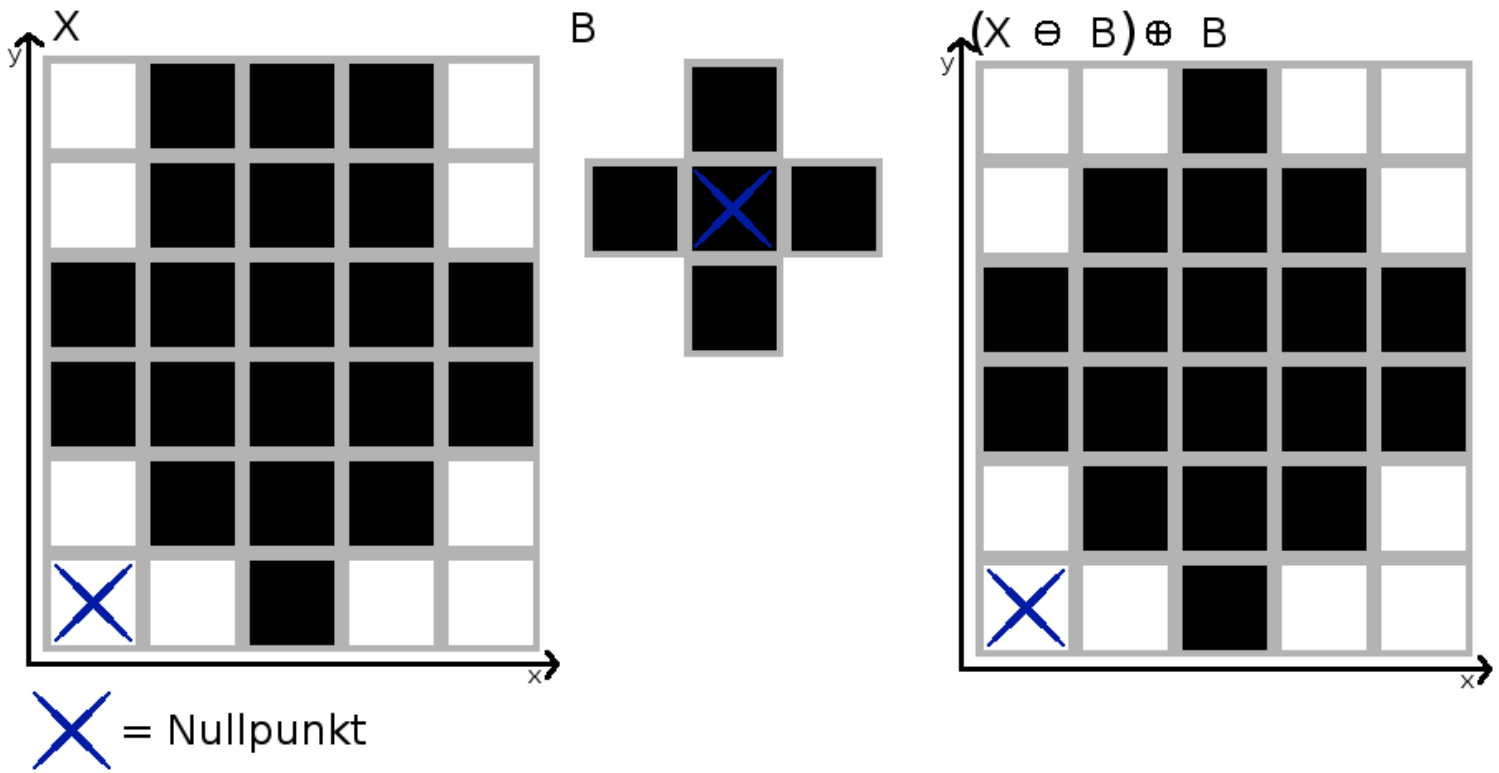
Erosion



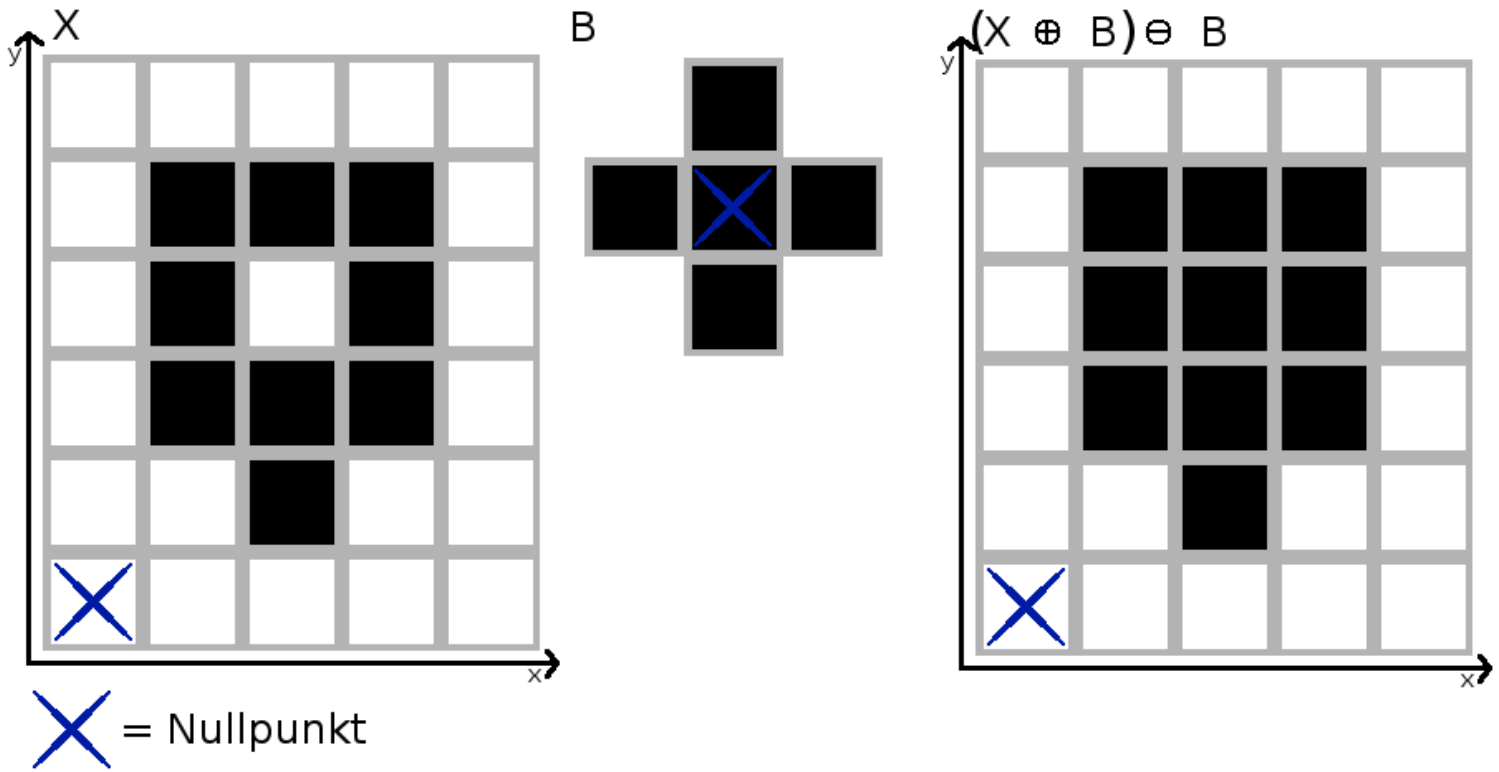
Erosion



Opening

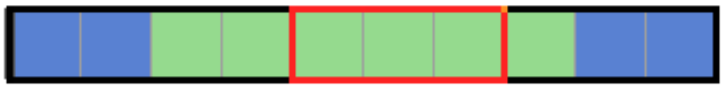


Closing



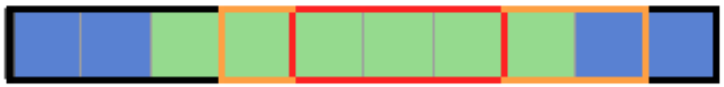
Morphologische Operationen auf Teilstücken des Datensatzes

- Aufteilung der Berechnung auf Teilstücke des Datensatzes um Speicherbedarf zu minimieren



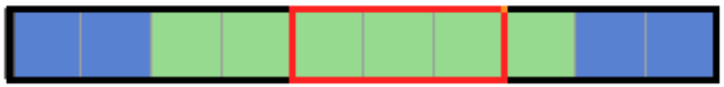
Morphologische Operationen auf Teilstücken des Datensatzes

- Aufteilung der Berechnung auf Teilstücke des Datensatzes um Speicherbedarf zu minimieren
- zum zu bearbeitenden Teilstück werden ausreichend große Übergangsbereiche zu den benachbarten Teilstücken geladen



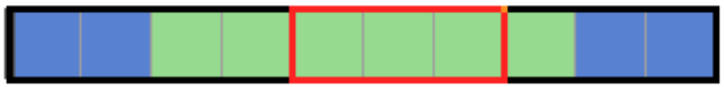
Morphologische Operationen auf Teilstücken des Datensatzes

- Aufteilung der Berechnung auf Teilstücke des Datensatzes um Speicherbedarf zu minimieren
- zum zu bearbeitenden Teilstück werden ausreichend große Übergangsbereiche zu den benachbarten Teilstücken geladen
- Operation wird auf kompletten geladenen Bereich ausgeführt, anschließend wird nur das Teilstück zurückgeschrieben



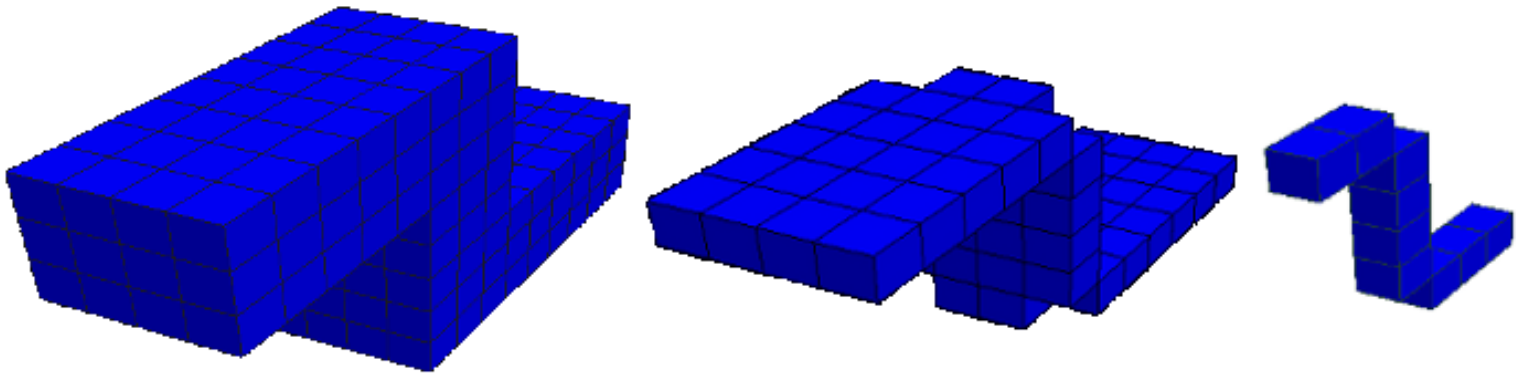
Morphologische Operationen auf Teilstücken des Datensatzes

- Aufteilung der Berechnung auf Teilstücke des Datensatzes um Speicherbedarf zu minimieren
- zum zu bearbeitenden Teilstück werden ausreichend große Übergangsbereiche zu den benachbarten Teilstücken geladen
- Operation wird auf kompletten geladenen Bereich ausgeführt, anschließend wird nur das Teilstück zurückgeschrieben
- vor der Operation wird ein Backup des Datensatzes erzeugt



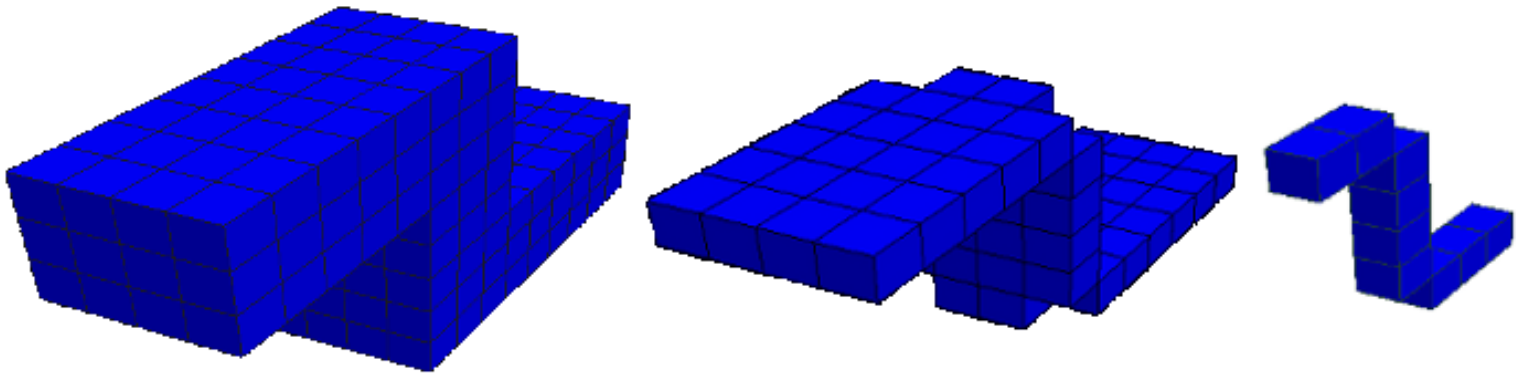
Skelettierung / Ausdünnung

- Erzeugung linienhafter Objekte aus flächenhaften



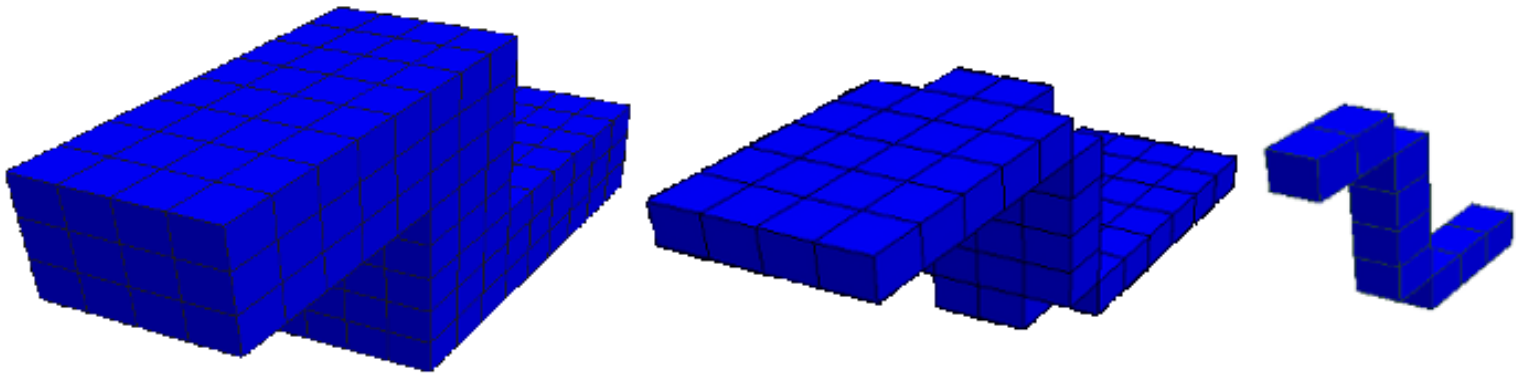
Skelettierung / Ausdünnung

- Erzeugung linienhafter Objekte aus flächenhaften
- z.B. für Formbeschreibung komplexer Objekte, Kurvendarstellung und Trennung von Objekten



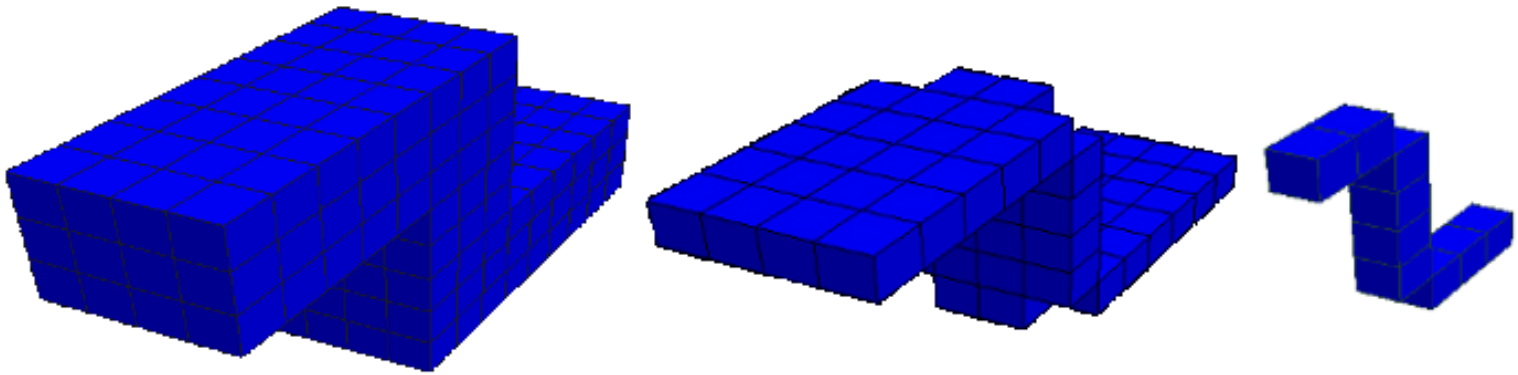
Skelettierung / Ausdünnung

- Erzeugung linienhafter Objekte aus flächenhaften
- z.B. für Formbeschreibung komplexer Objekte, Kurvendarstellung und Trennung von Objekten
- parallele und sequenzielle Skeltettierungsalgorithmen



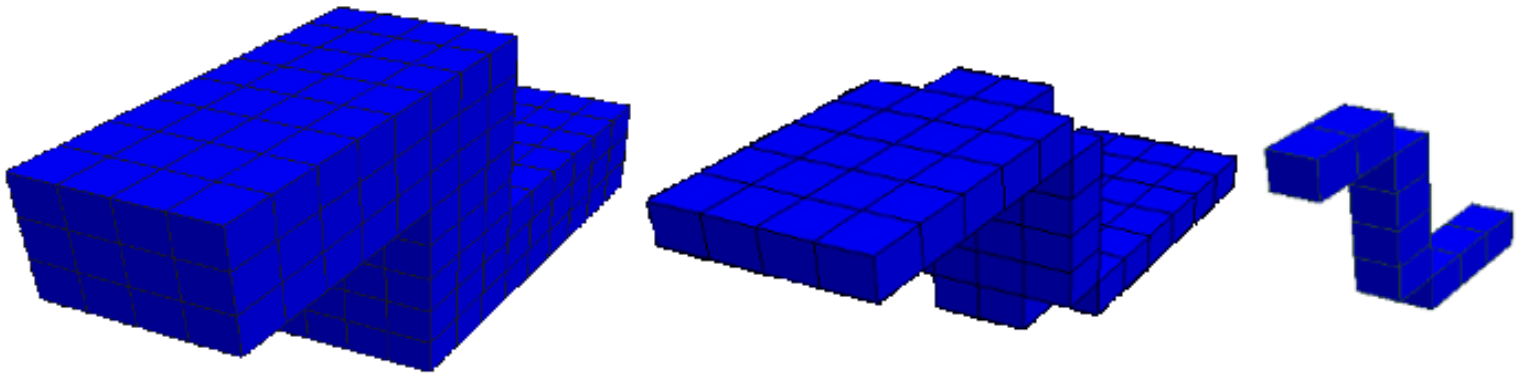
Skelettierung / Ausdünnung

- Erzeugung linienhafter Objekte aus flächenhaften
- z.B. für Formbeschreibung komplexer Objekte, Kurvendarstellung und Trennung von Objekten
- parallele und sequenzielle Skeltettierungsalgorithmen
- zwei Arten von Skeletten im 3D:
  - mediale Fläche
  - mediale Achse



Skelettierung / Ausdünnung

- Erzeugung linienhafter Objekte aus flächenhaften
- z.B. für Formbeschreibung komplexer Objekte, Kurvendarstellung und Trennung von Objekten
- parallele und sequenzielle Skeltettierungsalgorithmen
- zwei Arten von Skeletten im 3D:
  - mediale Fläche
  - mediale Achse
- Skelette sind topologisch äquivalent zum Objekt



Topologie

Die Beschreibung eines grafischen Objekts über diverse Charakteristika, welche invariant gegenüber Homöomorphismen (bijektive, stetige Abbildungen zwischen zwei Objekten wie Dehnen, Stauchen, Verbiegen oder Verzerren) sind.

Topologie

Die Beschreibung eines grafischen Objekts über diverse Charakteristika, welche invariant gegenüber Homöomorphismen (bijektive, stetige Abbildungen zwischen zwei Objekten wie Dehnen, Stauchen, Verbiegen oder Verzerren) sind.

Topologische Eigenschaften

- Anzahl der Verbundskomponenten (c)

Topologie

Die Beschreibung eines grafischen Objekts über diverse Charakteristika, welche invariant gegenüber Homöomorphismen (bijektive, stetige Abbildungen zwischen zwei Objekten wie Dehnen, Stauchen, Verbiegen oder Verzerren) sind.

Topologische Eigenschaften

- Anzahl der Verbundskomponenten (c)
- Anzahl der Löcher (h)

Topologie

Die Beschreibung eines grafischen Objekts über diverse Charakteristika, welche invariant gegenüber Homöomorphismen (bijektive, stetige Abbildungen zwischen zwei Objekten wie Dehnen, Stauchen, Verbiegen oder Verzerren) sind.

Topologische Eigenschaften

- Anzahl der Verbundskomponenten (c)
- Anzahl der Löcher (h)
- Geschlecht des Objekts (g)

Topologie

Die Beschreibung eines grafischen Objekts über diverse Charakteristika, welche invariant gegenüber Homöomorphismen (bijektive, stetige Abbildungen zwischen zwei Objekten wie Dehnen, Stauchen, Verbiegen oder Verzerren) sind.

Topologische Eigenschaften

- Anzahl der Verbundskomponenten (c)
- Anzahl der Löcher (h)
- Geschlecht des Objekts (g)

Euler-Charakteristik

$$\chi(X) = c + h - g$$

Simple Punkte

Sei $I \subset \mathbb{Z}^3$ ein 3-dimensionales Binärbild. Ein Punkt $x \in I$ wird als **simpler Punkt** bezeichnet, wenn sein Löschen die Eulerzahl von I nicht verändert.

Simple Punkte

Sei $I \subset \mathbb{Z}^3$ ein 3-dimensionales Binärbild. Ein Punkt $x \in I$ wird als **simpler Punkt** bezeichnet, wenn sein Löschen die Eulerzahl von I nicht verändert.

Allgemeines Vorgehen für die Ausdünnung

- Skelettierungsalgorithmen bestehen aus dem iterativen Entfernen von simplen Punkten

Simple Punkte

Sei $I \subset \mathbb{Z}^3$ ein 3-dimensionales Binärbild. Ein Punkt $x \in I$ wird als **simpler Punkt** bezeichnet, wenn sein Löschen die Eulerzahl von I nicht verändert.

Allgemeines Vorgehen für die Ausdünnung

- Skelettierungsalgorithmen bestehen aus dem iterativen Entfernen von simplen Punkten
- Euler-Charakteristik ist globale Eigenschaft

Simple Punkte

Sei $I \subset \mathbb{Z}^3$ ein 3-dimensionales Binärbild. Ein Punkt $x \in I$ wird als **simpler Punkt** bezeichnet, wenn sein Löschen die Eulerzahl von I nicht verändert.

Allgemeines Vorgehen für die Ausdünnung

- Skelettierungsalgorithmen bestehen aus dem iterativen Entfernen von simplen Punkten
- Euler-Charakteristik ist globale Eigenschaft
- benötigen Kriterium, welches sich über kleine Nachbarschaften überprüfen läßt

Simple Punkte

Sei $I \subset \mathbb{Z}^3$ ein 3-dimensionales Binärbild. Ein Punkt $x \in I$ wird als **simpler Punkt** bezeichnet, wenn sein Löschen die Eulerzahl von I nicht verändert.

Allgemeines Vorgehen für die Ausdünnung

- Skelettierungsalgorithmen bestehen aus dem iterativen Entfernen von simplen Punkten
- Euler-Charakteristik ist globale Eigenschaft
- benötigen Kriterium, welches sich über kleine Nachbarschaften überprüfen läßt
- Änderung der Eulerzahl läßt sich durch Zählen der Verbundskomponenten in kleinen Nachbarschaften bestimmen

Kleine Nachbarschaften

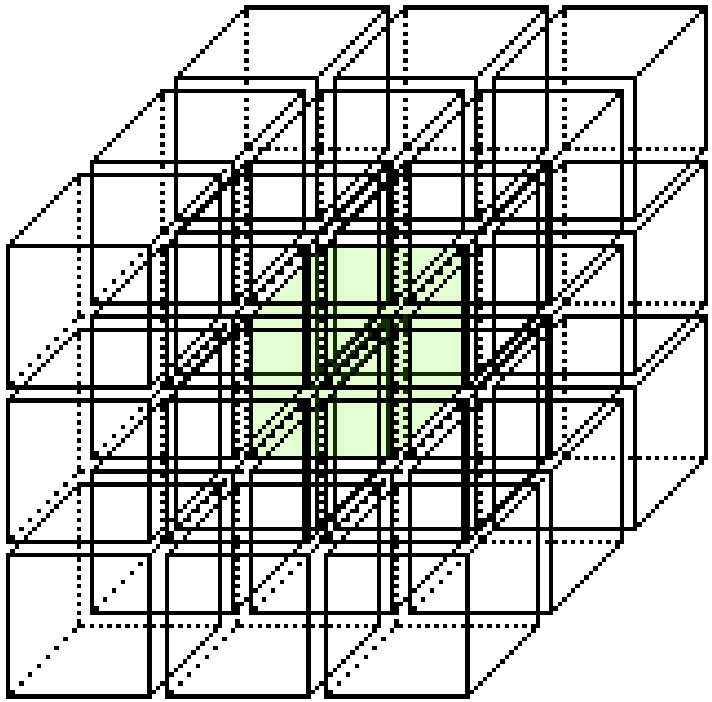
- N_6-Nachbarschaft

Kleine Nachbarschaften

- N_6-Nachbarschaft
- N_{26}-Nachbarschaft

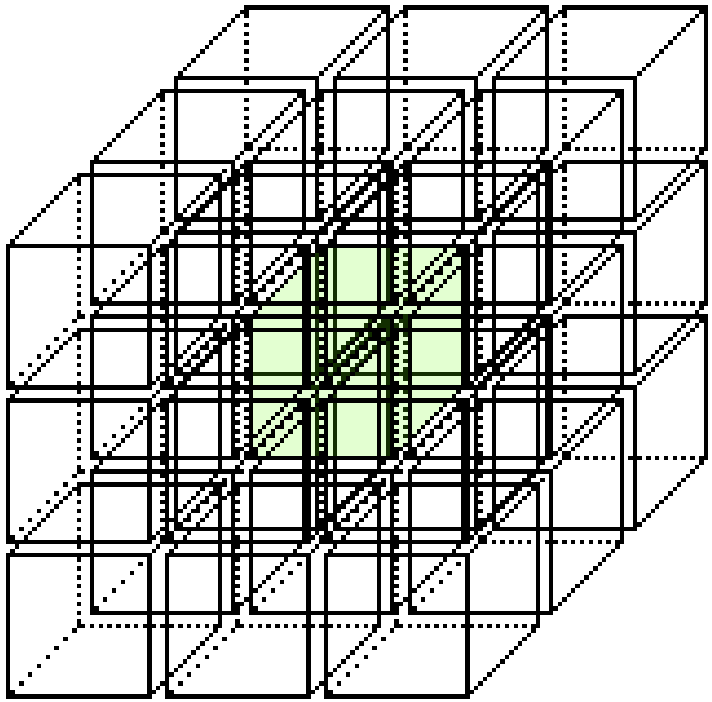
Kleine Nachbarschaften

- N_6-Nachbarschaft
- N_{26}-Nachbarschaft



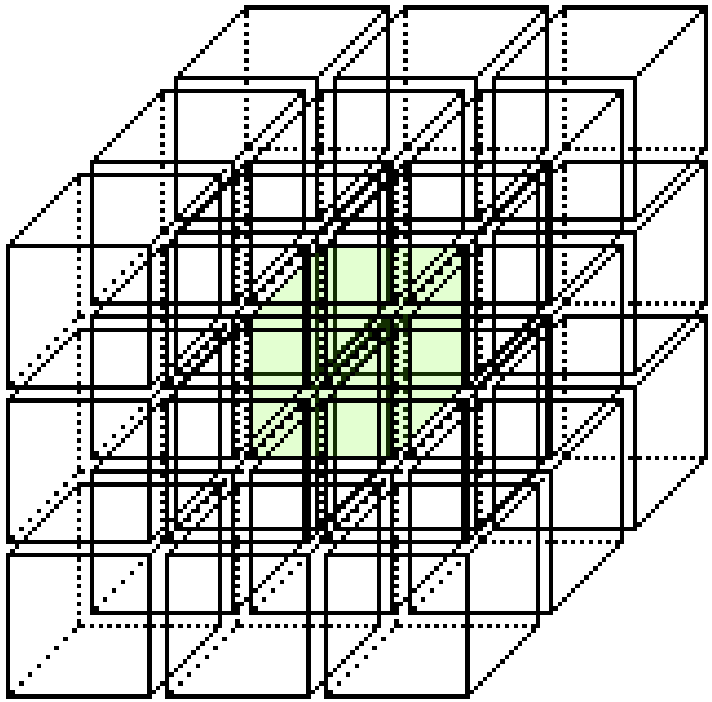
Kleine Nachbarschaften

- N_6-Nachbarschaft: alle Voxel, die genau eine Fläche mit dem betrachteten Voxel gemeinsam haben
- N_{26}-Nachbarschaft



Kleine Nachbarschaften

- N_6-Nachbarschaft: alle Voxel, die genau eine Fläche mit dem betrachteten Voxel gemeinsam haben
- N_{26}-Nachbarschaft: alle Voxel, die einen Punkt, eine Kante oder eine Fläche mit dem betrachteten Voxel gemeinsam haben



Simple Punkte

Ein Punkt ist simpel, wenn die Anzahl der Verbundskomponenten in seiner N_{26} Nachbarschaft für den Vordergrund und Hintergrund jeweils genau 1 ist.

Simple Punkte

Ein Punkt ist simpel, wenn die Anzahl der Verbundskomponenten in seiner N_{26} Nachbarschaft für den Vordergrund und Hintergrund jeweils genau 1 ist.

Probleme beim Entfernen simpler Punkte

- Erhaltung der Topologie stellt Minimalanforderung für Skelettierung dar

Simple Punkte

Ein Punkt ist simpel, wenn die Anzahl der Verbundskomponenten in seiner N_{26} Nachbarschaft für den Vordergrund und Hintergrund jeweils genau 1 ist.

Probleme beim Entfernen simpler Punkte

- Erhaltung der Topologie stellt Minimalanforderung für Skelettierung dar
- paralles Löschen von simplen Punkten kann das Objekt verschwinden lassen

Simple Punkte

Ein Punkt ist simpel, wenn die Anzahl der Verbundskomponenten in seiner N_{26} Nachbarschaft für den Vordergrund und Hintergrund jeweils genau 1 ist.

Probleme beim Entfernen simpler Punkte

- Erhaltung der Topologie stellt Minimalanforderung für Skelettierung dar
- paralles Löschen von simplen Punkten kann das Objekt verschwinden lassen
- sequenzielles Löschen von simplen Punkten kann das Objekt bis auf einen Voxel degenerieren lassen

Simple Punkte

Ein Punkt ist simpel, wenn die Anzahl der Verbundskomponenten in seiner N_{26} Nachbarschaft für den Vordergrund und Hintergrund jeweils genau 1 ist.

Probleme beim Entfernen simpler Punkte

- Erhaltung der Topologie stellt Minimalanforderung für Skelettierung dar
- paralles Löschen von simplen Punkten kann das Objekt verschwinden lassen
- sequenzielles Löschen von simplen Punkten kann das Objekt bis auf einen Voxel degenerieren lassen
- grundlegende Gestalt des Objekts muss ebenfalls erhalten bleiben

Nebenbedingungen

Finale Punkte (Kurvenendpunkte)

Ein simpler Punkt $x \in I$ wird als **finaler Punkt** bezeichnet, wenn die Anzahl n an Vordergrundvoxeln in seiner N_{26}-Nachbarschaft kleiner als 2 ist.

Nebenbedingungen

Finale Punkte (Kurvenendpunkte)

Ein simpler Punkt $x \in I$ wird als **finaler Punkt** bezeichnet, wenn die Anzahl n an Vordergrundvoxeln in seiner N_{26}-Nachbarschaft kleiner als 2 ist.

Gerichtete Randpunkte

Sei $I \subset \mathbb{Z}^3$ ein 3-dimensionales Binärbild und $p \in I$ ein Vordergrundvoxel, dann wird p als **gerichteter Randpunkt** der Richtung N(orden), O(sten), W(esten), S(üden), Ob(en), U(nten) bezeichnet, wenn das zu p adjazente Voxel in Richtung N,O,W,S,Ob,U zum Hintergrund gehört.

Skelettierung nach Tsao und Fu

- Ermittlung der Koordinaten aller definierten Voxel des Objekts
- Iteration über alle sechs Richtungen
  - Iteration über alle gefundenen Voxel
    - ★ überprüfe, ob der aktuelle Voxel bereits gelöscht
    - ★ wenn nein, überprüfe, ob der aktuelle Voxel ein Randpunkt der aktuellen Richtung ist
    - ★ wenn ja, überprüfe, ob der Voxel nicht final aber simpel ist und die "Simplizizät der Ebene"-Bedingung erfüllt
    - ★ wenn ja, füge Voxel zu den zu löschenden Punkten hinzu und markiere ihn als gelöscht
  - lösche die zu löschenden Voxel aus der Datenstruktur und fahre mit der nächsten Richtung fort
- wurden Voxel in einer Richtung gelöscht, starte von vorn

Simplizität in der Ebene

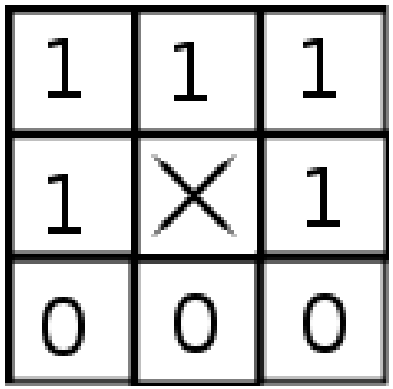
- stellt sicher, daß das entstehende Skelett “glatt” ist

Simplizität in der Ebene

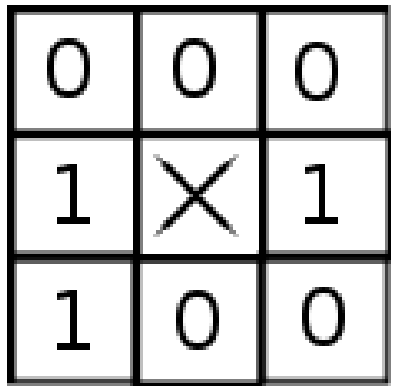
- stellt sicher, daß das entstehende Skelett “glatt” ist
- es werden die zwei zur aktuellen Richtung orthogonalen $3x3$ Ebenen ermittelt mit betrachteten Punkt im Zentrum

Simplizität in der Ebene

- stellt sicher, daß das entstehende Skelett “glatt” ist
- es werden die zwei zur aktuellen Richtung orthogonalen 3x3 Ebenen ermittelt mit betrachteten Punkt im Zentrum
- Bedingung gilt, wenn das Voxel simpel bezüglich beider Ebenen ist



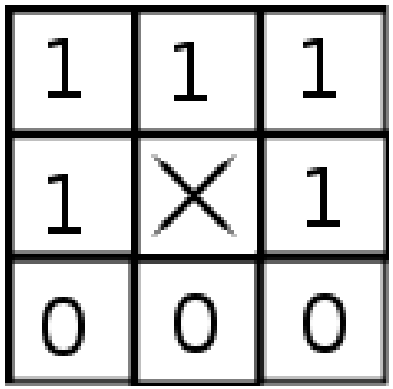
löschbar



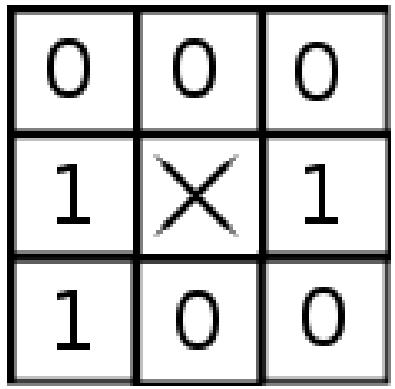
nicht löschbar

Simplizität in der Ebene

- stellt sicher, daß das entstehende Skelett “glatt” ist
- es werden die zwei zur aktuellen Richtung orthogonalen 3x3 Ebenen ermittelt mit betrachteten Punkt im Zentrum
- Bedingung gilt, wenn das Voxel simpel bezüglich beider Ebenen ist
- außerdem darf die Anzahl verbleibender Vordergrundvoxel nicht kleiner als 2 (mediale Fläche) bzw. 1 (mediale Achse) sein



löschbar



nicht löschbar

Löschen von Punkten aus der Datenstruktur

Probleme

- ein einzelner Punkt läßt sich nicht ohne weiteres aus der Datenstruktur entfernen

Löschen von Punkten aus der Datenstruktur

Probleme

- ein einzelner Punkt läßt sich nicht ohne weiteres aus der Datenstruktur entfernen
- alle Läufe und Unterbrechungen eines LLK Blocks in einem zusammenhängenden Array gespeichert $\rightarrow$ neu entstehende Läufe und Unterbrechungen können nicht einfach eingefügt werden

Löschen von Punkten aus der Datenstruktur

Probleme

- ein einzelner Punkt läßt sich nicht ohne weiteres aus der Datenstruktur entfernen
- alle Läufe und Unterbrechungen eines LLK Blocks in einem zusammenhängenden Array gespeichert $\rightarrow$ neu entstehende Läufe und Unterbrechungen können nicht einfach eingefügt werden
- Datenstruktur muß bei jedem Löschvorgang komplett neu aufgebaut werden

Löschen von Punkten aus der Datenstruktur

Probleme

- ein einzelner Punkt läßt sich nicht ohne weiteres aus der Datenstruktur entfernen
- alle Läufe und Unterbrechungen eines LLK Blocks in einem zusammenhängenden Array gespeichert → neu entstehende Läufe und Unterbrechungen können nicht einfach eingefügt werden
- Datenstruktur muß bei jedem Löschvorgang komplett neu aufgebaut werden

Lösungsansatz

- so viele Punkte wie möglich in einem Durchgang entfernen

Löschen von Punkten aus der Datenstruktur

Probleme

- ein einzelner Punkt läßt sich nicht ohne weiteres aus der Datenstruktur entfernen
- alle Läufe und Unterbrechungen eines LLK Blocks in einem zusammenhängenden Array gespeichert → neu entstehende Läufe und Unterbrechungen können nicht einfach eingefügt werden
- Datenstruktur muß bei jedem Löschvorgang komplett neu aufgebaut werden

Lösungsansatz

- so viele Punkte wie möglich in einem Durchgang entfernen
- Eingabe: Array mit nach ihren Koordinaten aufsteigendend sortierten Punkten (beginnend bei dimensional höchster Koordinate)

Löschen von Punkten aus der Datenstruktur

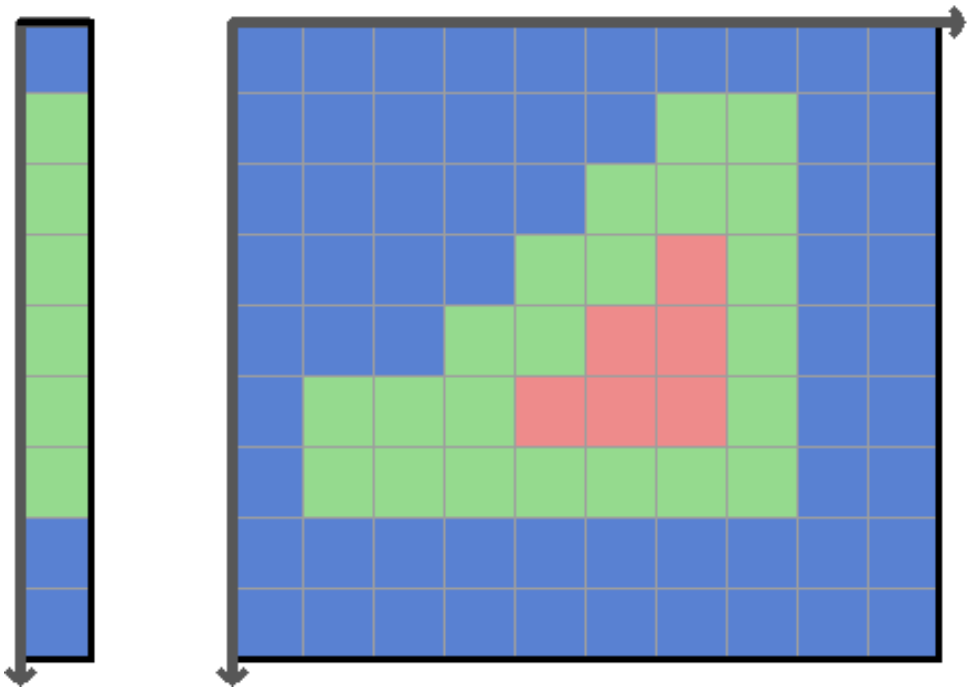
Probleme

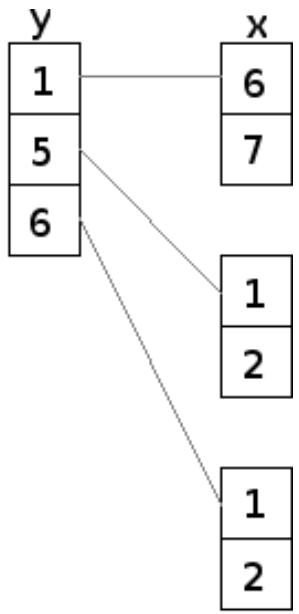
- ein einzelner Punkt läßt sich nicht ohne weiteres aus der Datenstruktur entfernen
- alle Läufe und Unterbrechungen eines LLK Blocks in einem zusammenhängenden Array gespeichert → neu entstehende Läufe und Unterbrechungen können nicht einfach eingefügt werden
- Datenstruktur muß bei jedem Löschvorgang komplett neu aufgebaut werden

Lösungsansatz

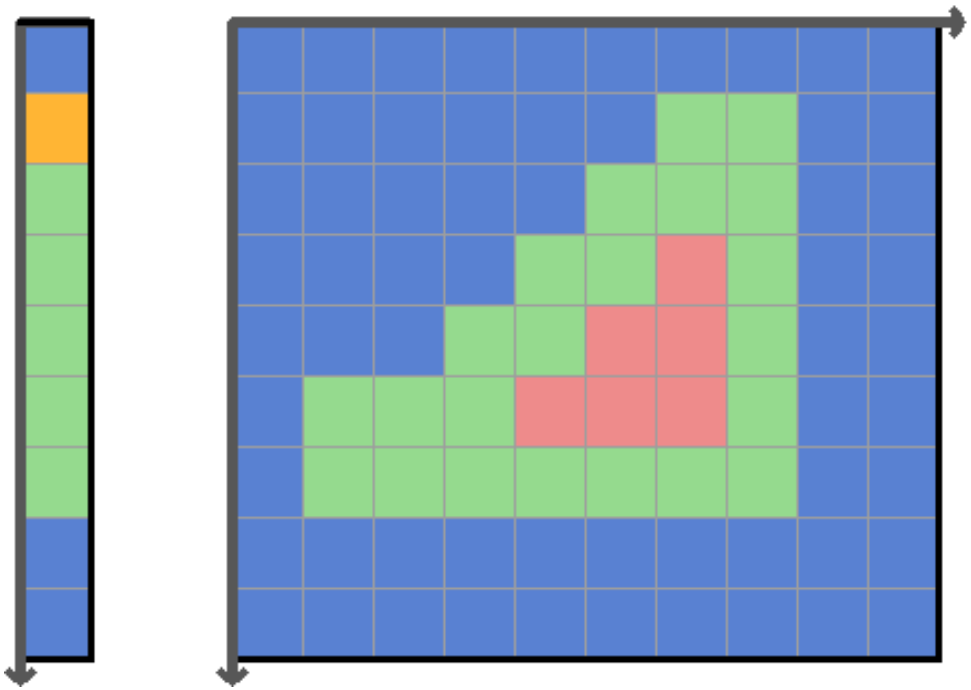
- so viele Punkte wie möglich in einem Durchgang entfernen
- Eingabe: Array mit nach ihren Koordinaten aufsteigendend sortierten Punkten (beginnend bei dimensional höchster Koordinate)
- Arrays der Datenstruktur werden neu erzeugt und Punkte dabei entfernt

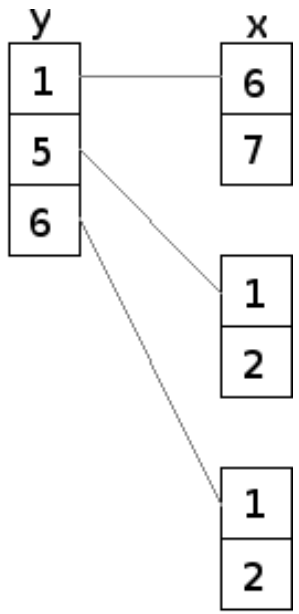
Löschen von Punkten aus der Datenstruktur



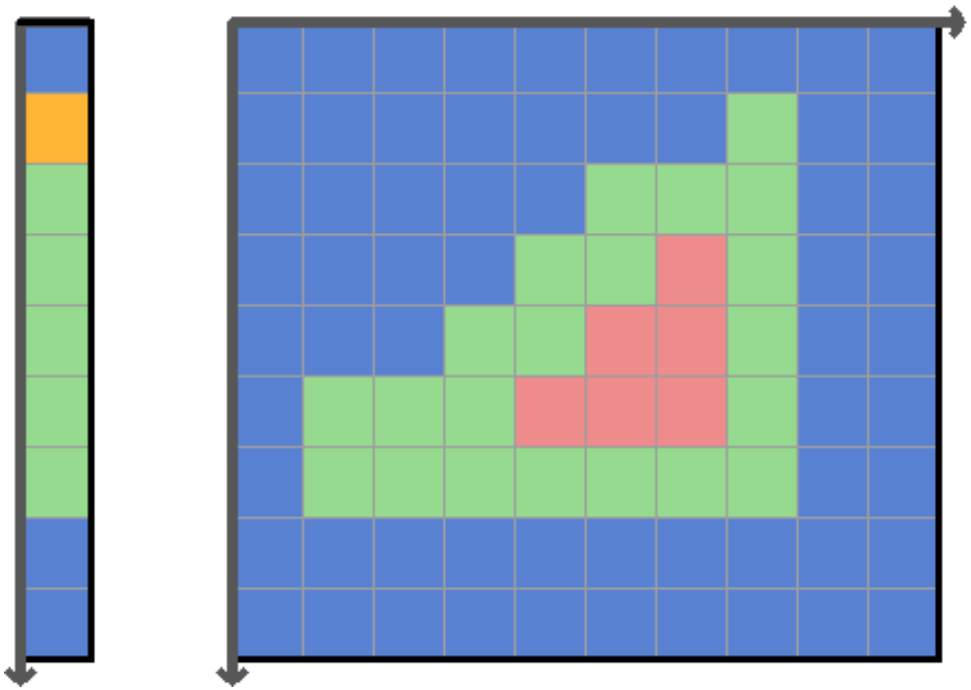


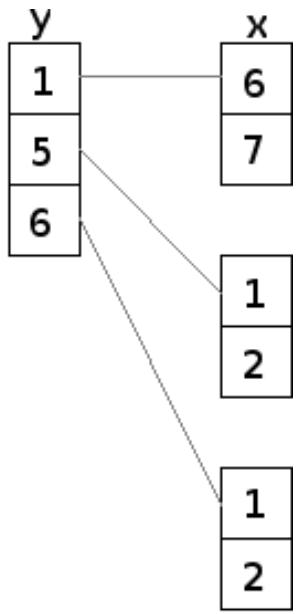
Löschen von Punkten aus der Datenstruktur



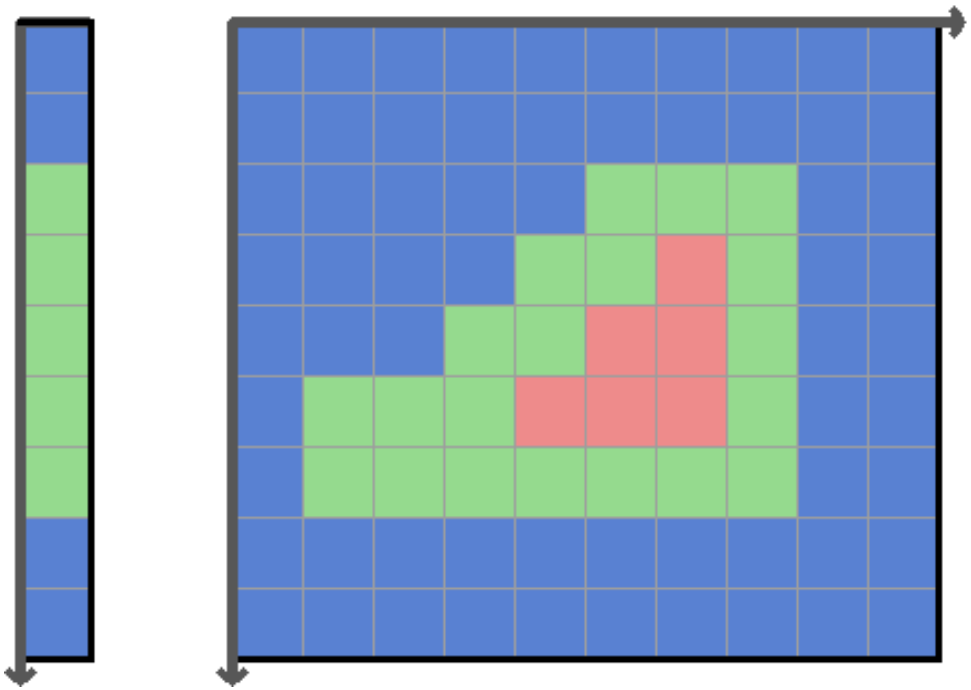


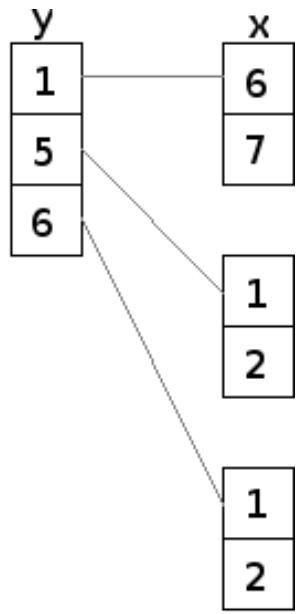
Löschen von Punkten aus der Datenstruktur



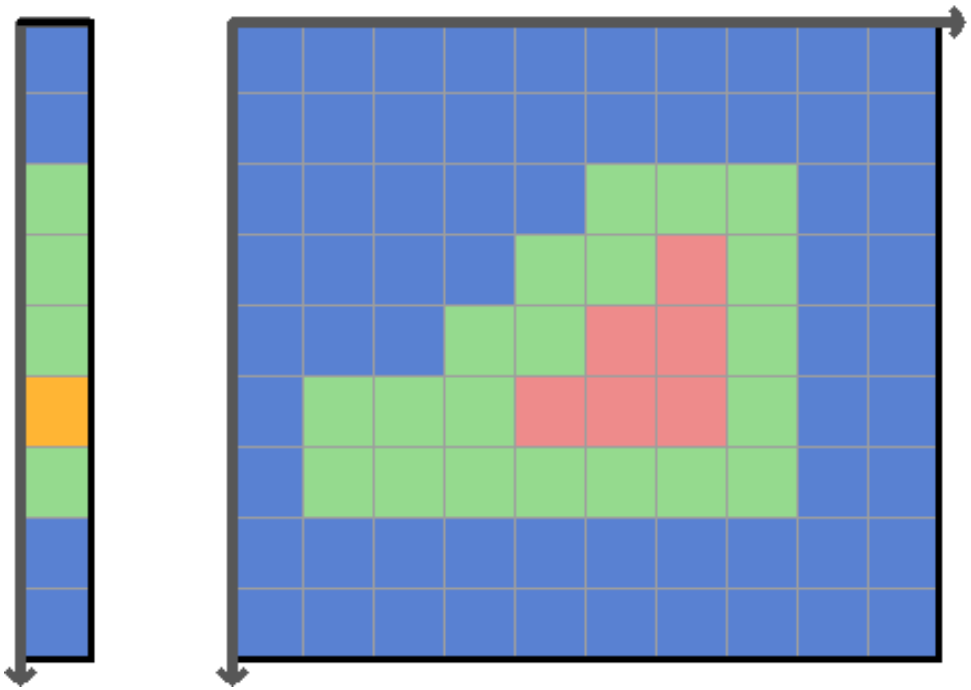


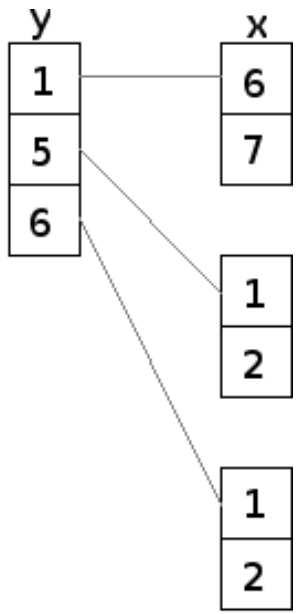
Löschen von Punkten aus der Datenstruktur



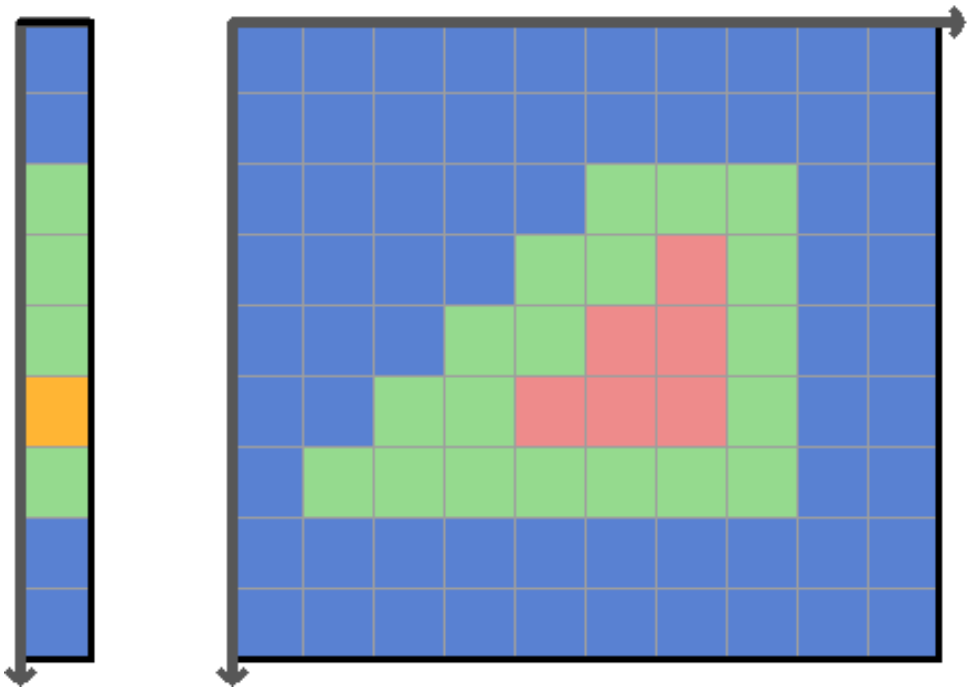


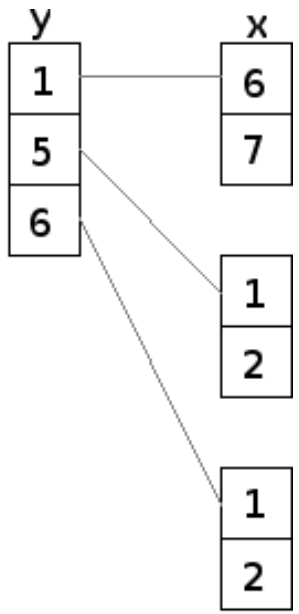
Löschen von Punkten aus der Datenstruktur



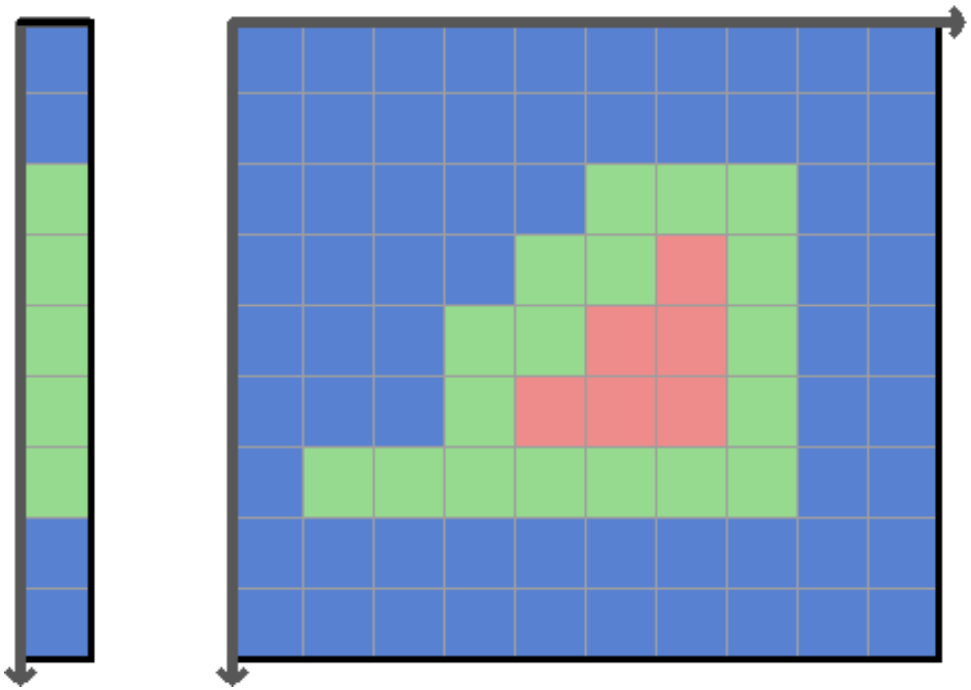


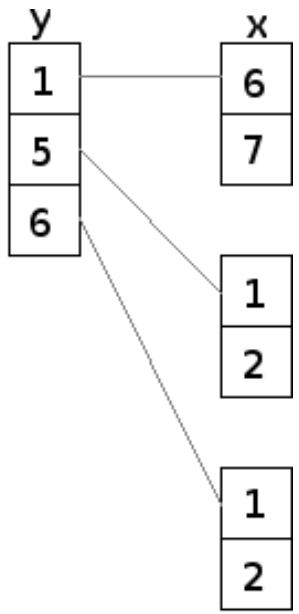
Löschen von Punkten aus der Datenstruktur



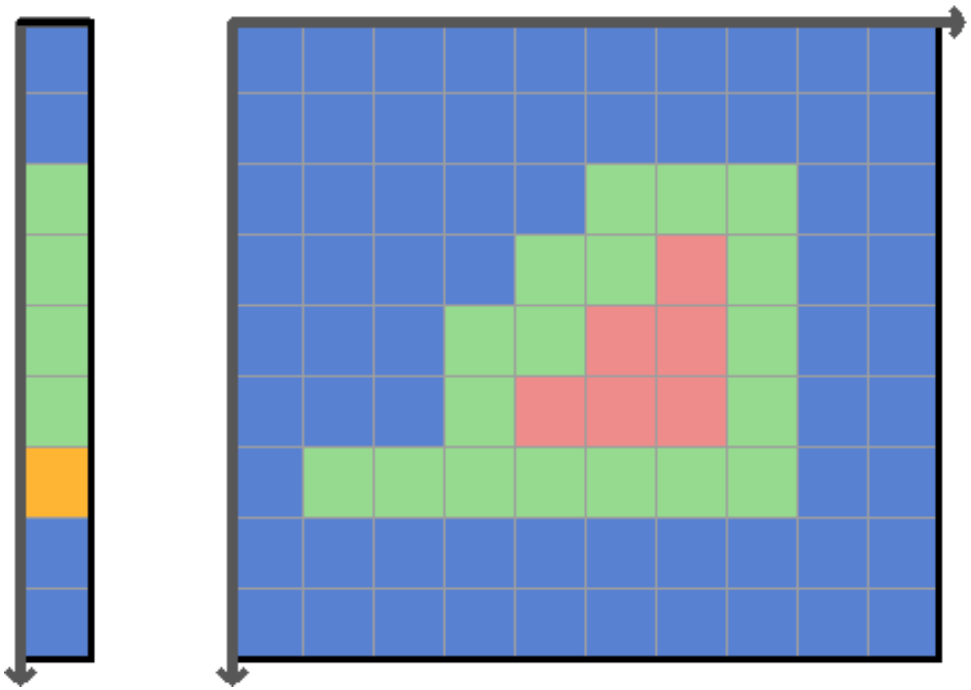


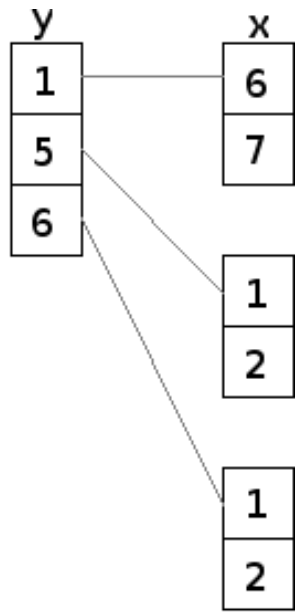
Löschen von Punkten aus der Datenstruktur



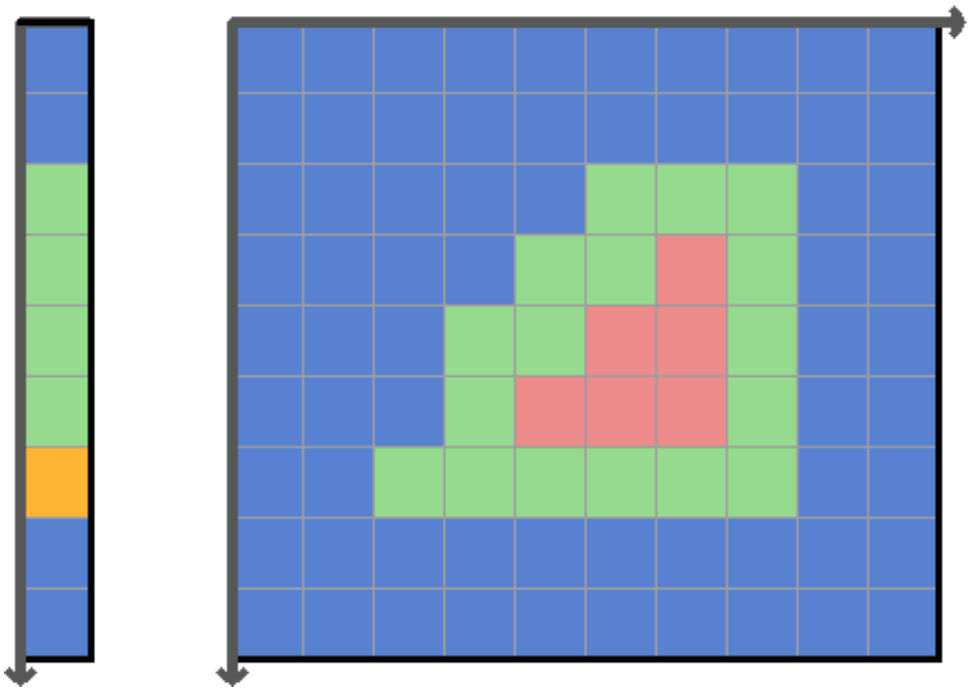


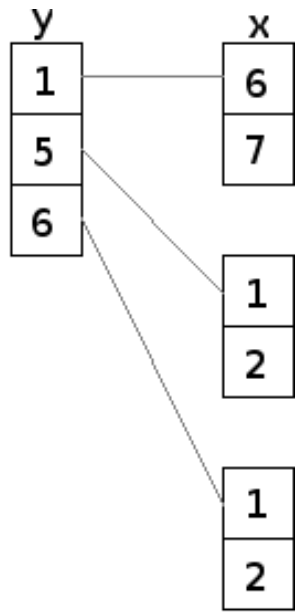
Löschen von Punkten aus der Datenstruktur



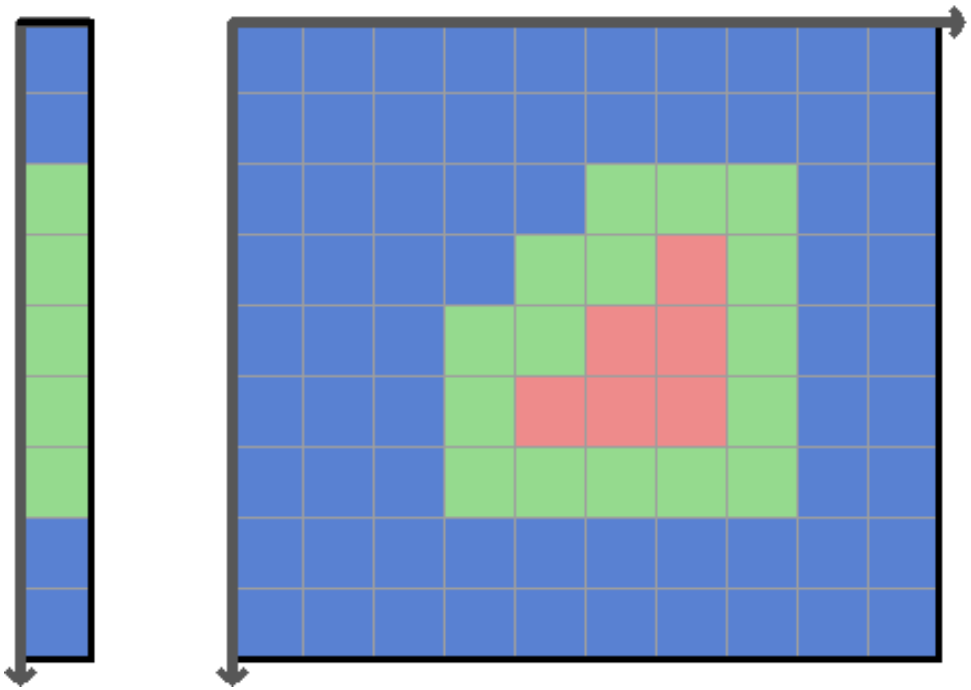


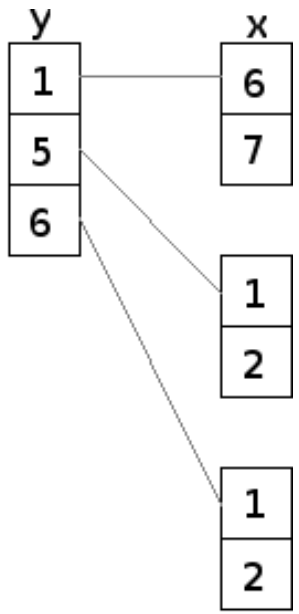
Löschen von Punkten aus der Datenstruktur



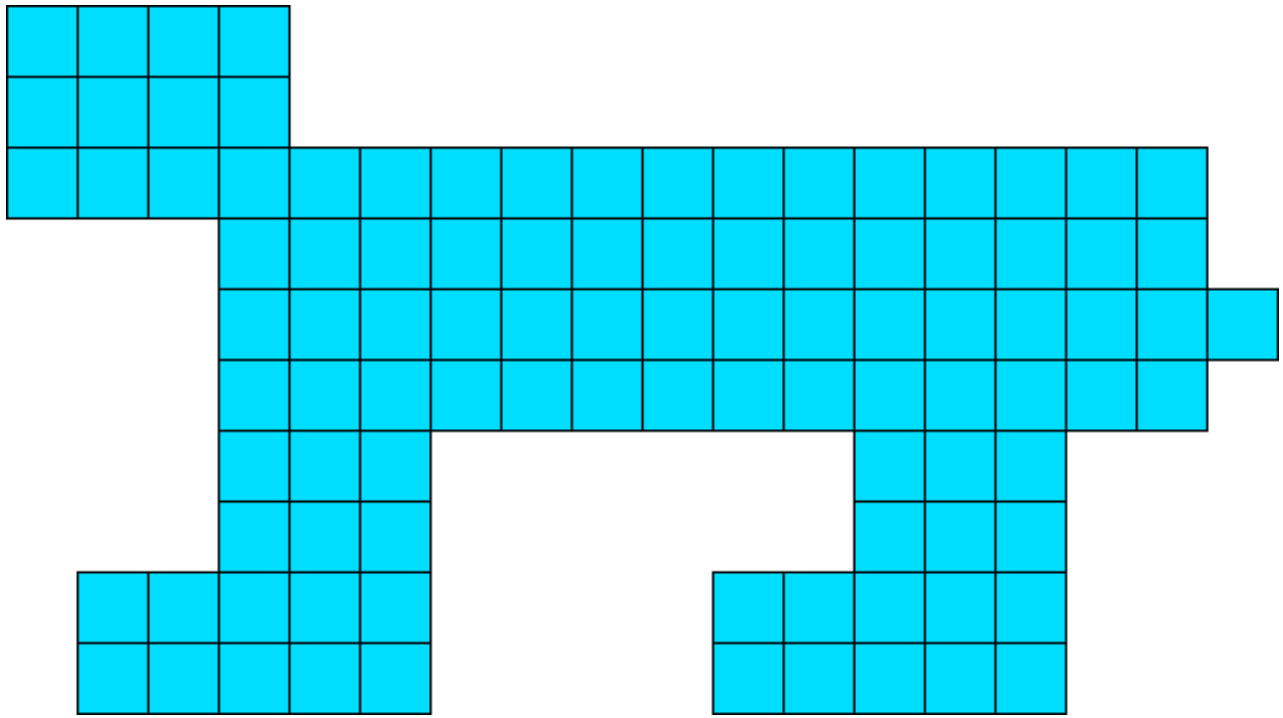


Löschen von Punkten aus der Datenstruktur

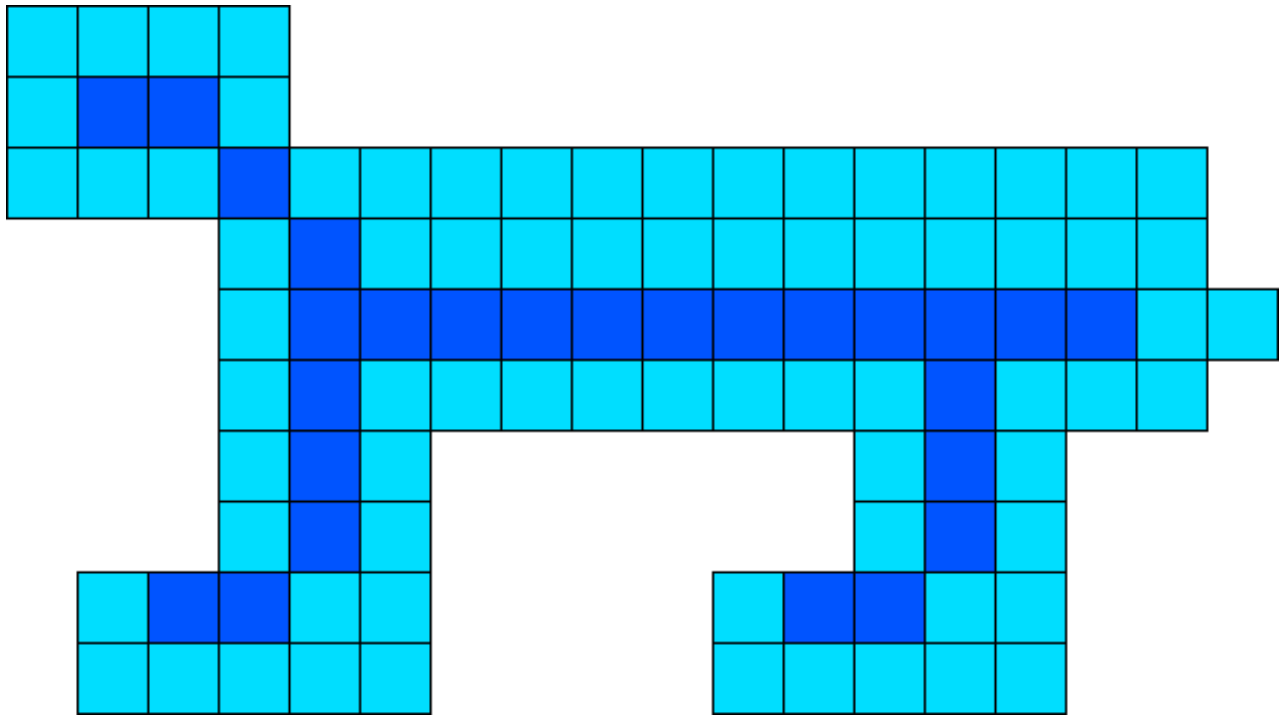




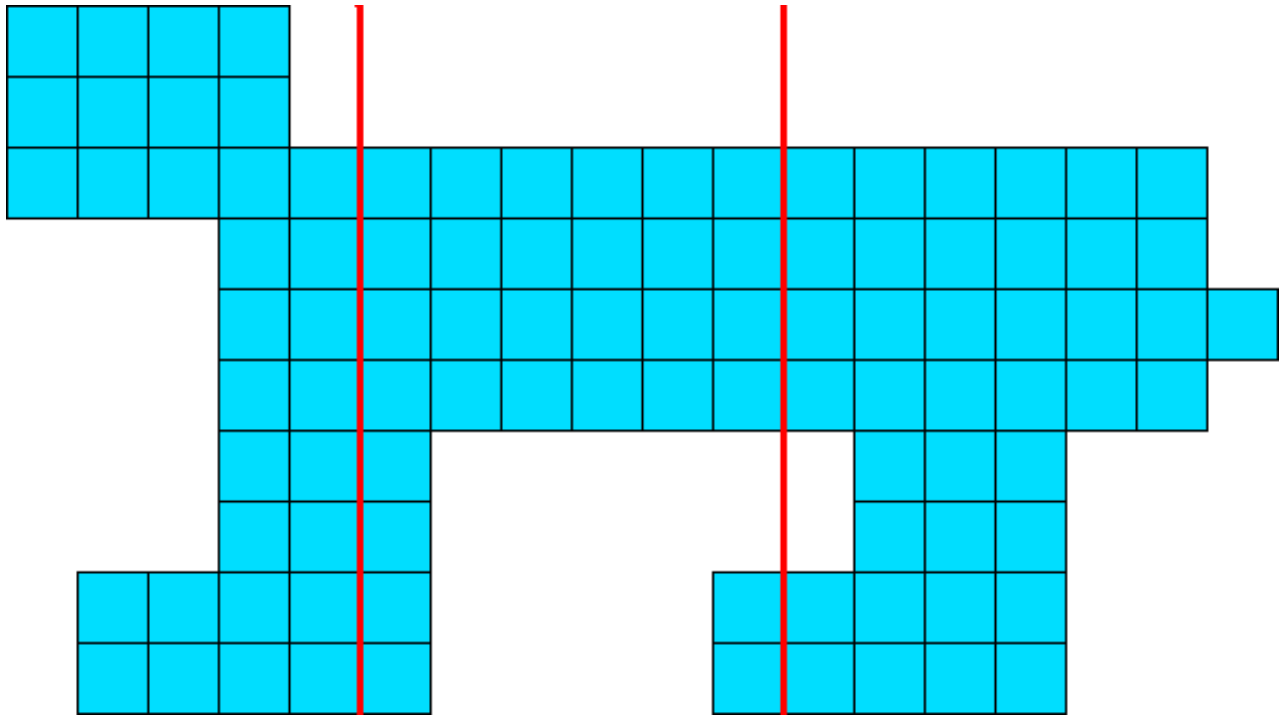
Partielle Skelettierung



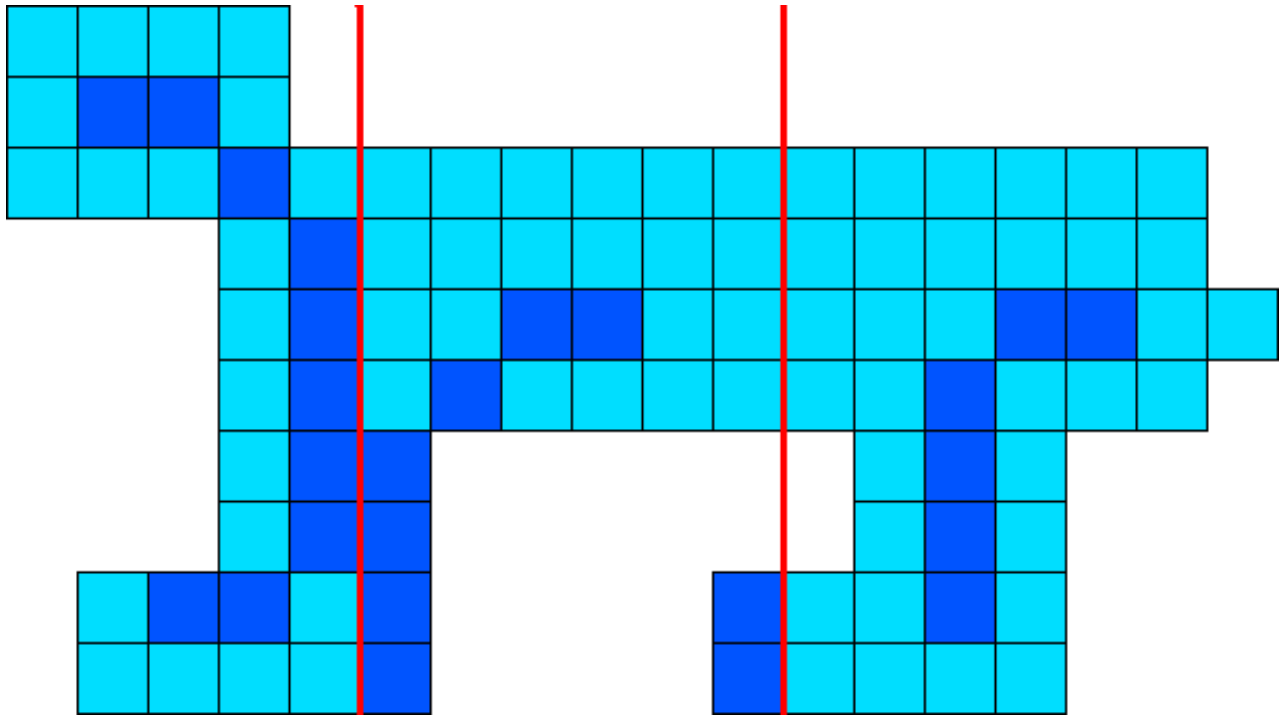
Partielle Skelettierung



Partielle Skelettierung



Partielle Skelettierung



Partielle Skelettierung

- 1. Problem: ohne Kenntnis des restlichen Objekts lassen sich keine gültigen Übergangsbedingungen zwischen zwei Teilstücken finden

Partielle Skelettierung

- 1. Problem: ohne Kenntnis des restlichen Objekts lassen sich keine gültigen Übergangsbedingungen zwischen zwei Teilstücken finden
- 2. Problem: bereits ein kompletter Ausdünnungszyklus auf einem Teilstück kann zur Deformation des Skeletts führen

Partielle Skelettierung

- 1. Problem: ohne Kenntnis des restlichen Objekts lassen sich keine gültigen Übergangsbedingungen zwischen zwei Teilstücken finden
- 2. Problem: bereits ein kompletter Ausdünnungszyklus auf einem Teilstück kann zur Deformation des Skeletts führen
- Iteration über alle sechs Richtungen

Partielle Skelettierung

- 1. Problem: ohne Kenntnis des restlichen Objekts lassen sich keine gültigen Übergangsbedingungen zwischen zwei Teilstücken finden
- 2. Problem: bereits ein kompletter Ausdünnungszyklus auf einem Teilstück kann zur Deformation des Skeletts führen
- Iteration über alle sechs Richtungen
- für jede Richtung wird über alle Teilstücke iteriert und Randpunkte der aktuellen Richtung gelöscht

Partielle Skelettierung

- 1. Problem: ohne Kenntnis des restlichen Objekts lassen sich keine gültigen Übergangsbedingungen zwischen zwei Teilstücken finden
- 2. Problem: bereits ein kompletter Ausdünnungszyklus auf einem Teilstück kann zur Deformation des Skeletts führen
- Iteration über alle sechs Richtungen
- für jede Richtung wird über alle Teilstücke iteriert und Randpunkte der aktuellen Richtung gelöscht
- falls in einem Teilstück einer Richtung Punkte gelöscht wurden, startet Algorithmus neu

Speicherbedarf

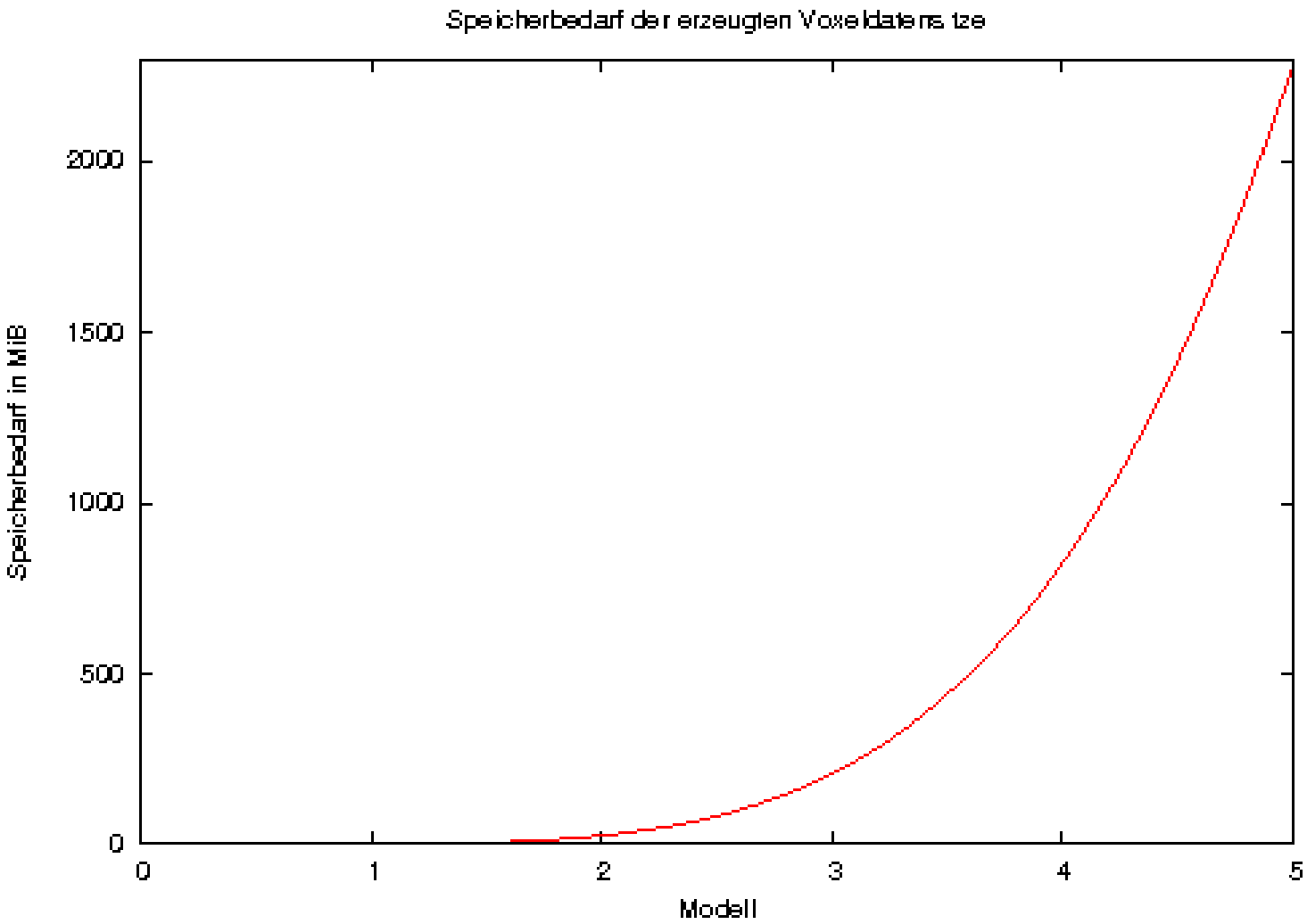


Abbildung: erzeugte Voxelmengen

Speicherbedarf

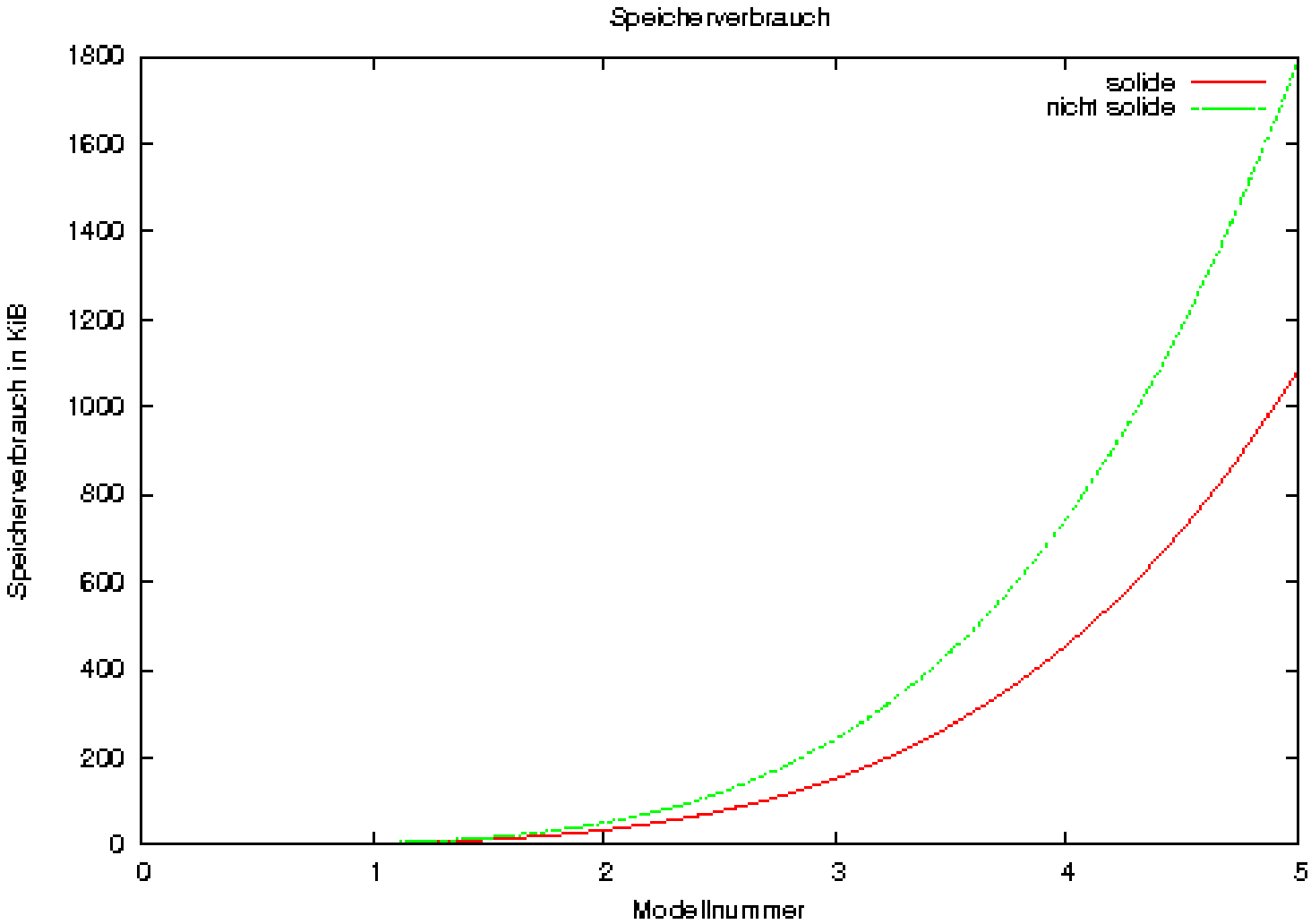
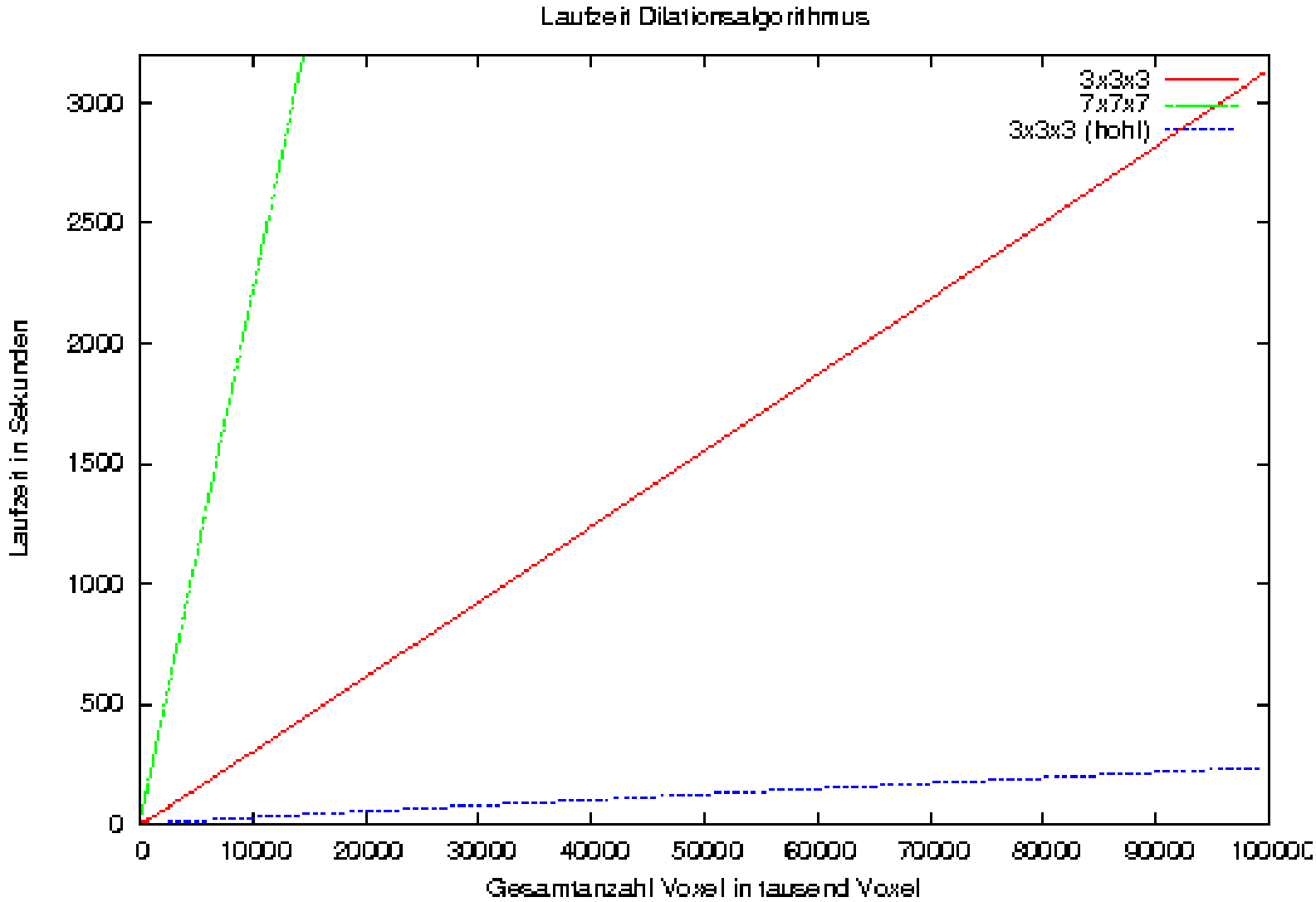
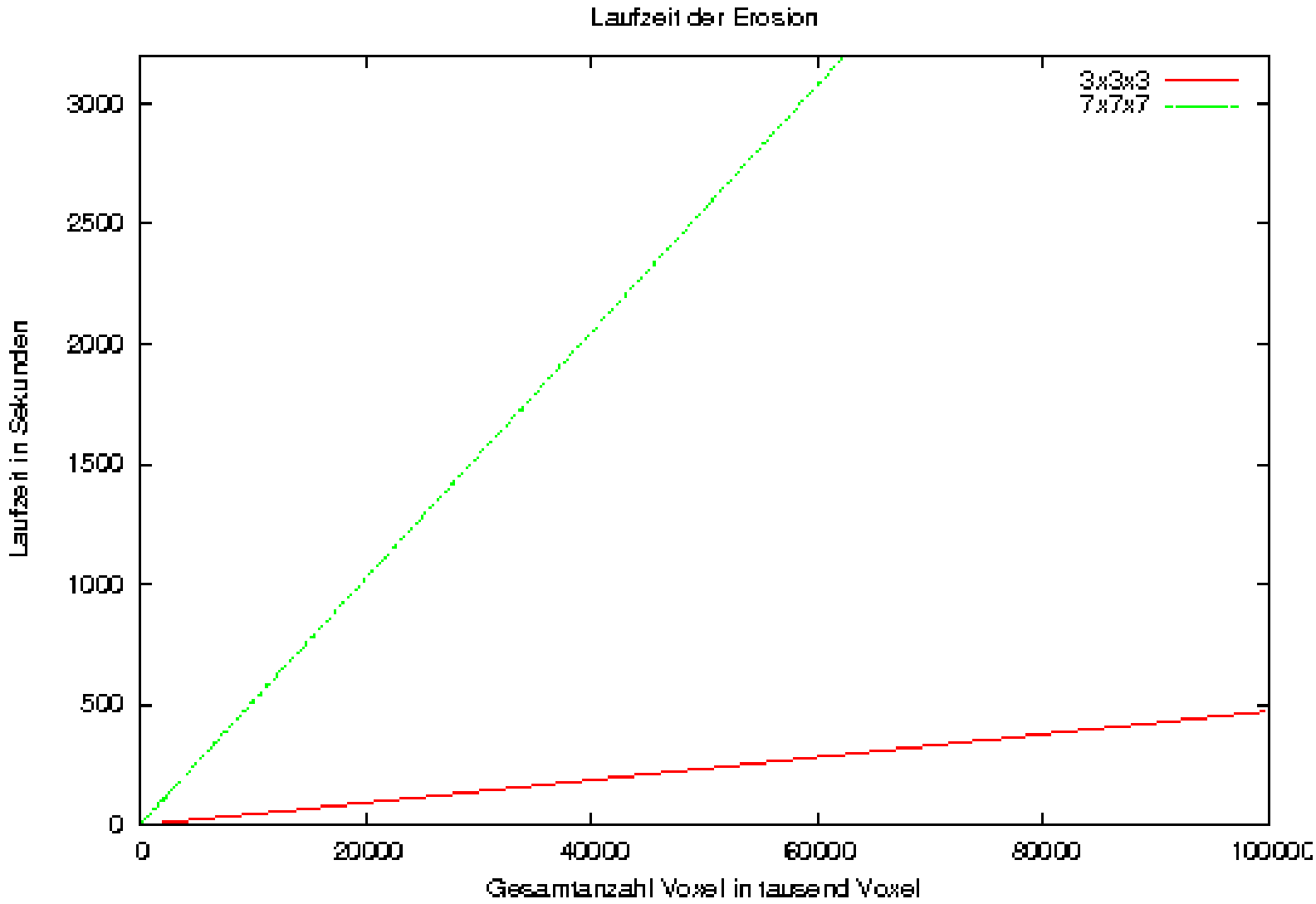


Abbildung: erweiterte H-LLK Datenstruktur

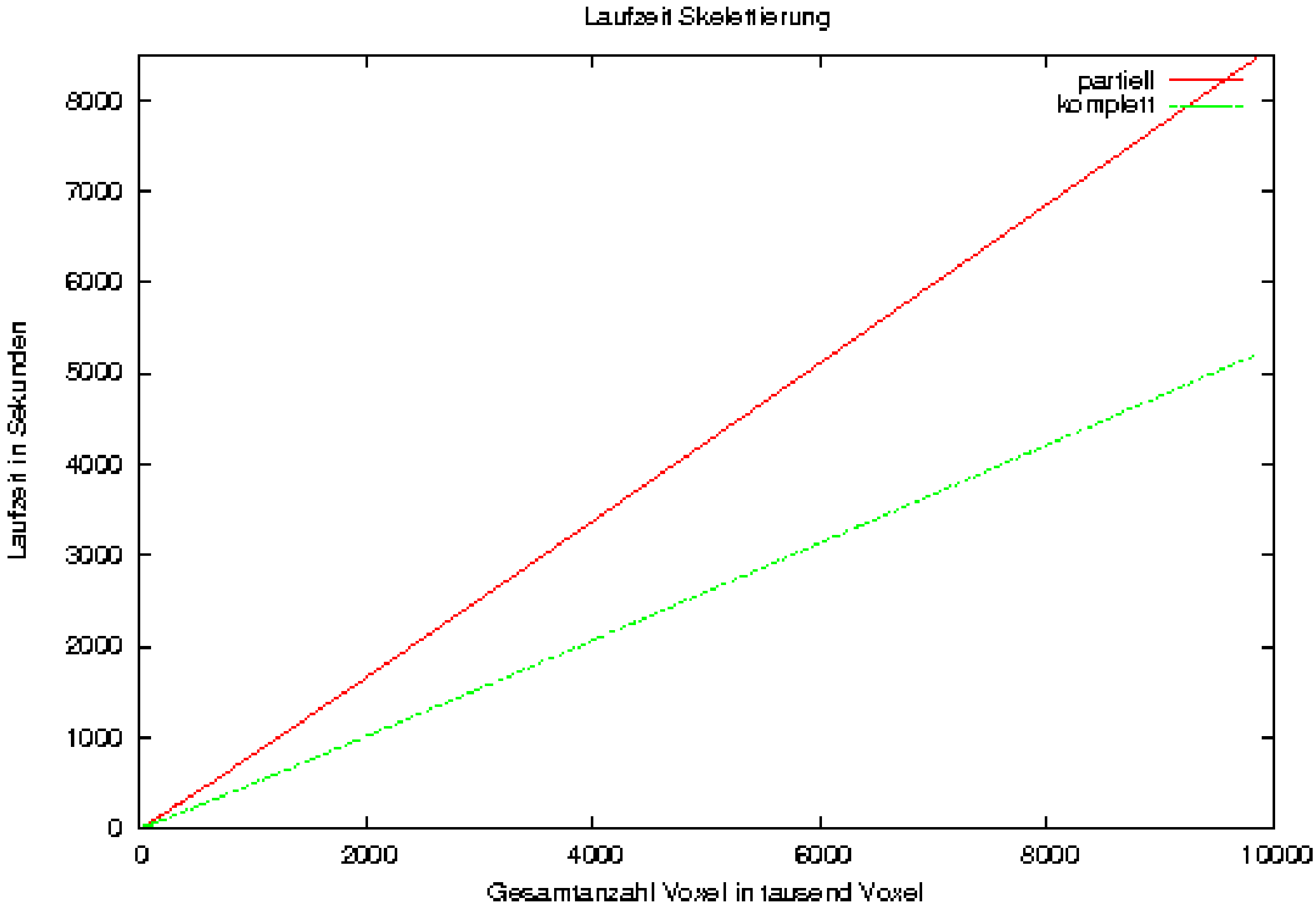
Dilation


Laufzeit Dilationsalgorithmus
3x3x3
7x7x7
3x3x3 (hohl)
Laufzeit in Sekunden
0
500
1000
1500
2000
2500
3000
0
10000
20000
30000
40000
50000
60000
70000
80000
90000
100000
Gesamtanzahl Voxel in tausend Voxel

Erosion


Laufzeit der Erosion
3x3x3
7x7x7
Laufzeit in Sekunden
0
500
1000
1500
2000
2500
3000
0
20000
40000
60000
80000
100000
Gesamtanzahl Voxel in tausend Voxel

Skelettierung



Zusammenfassung

- speichereffizient und schneller Zugriff auf die Daten

Zusammenfassung

- speichereffizient und schneller Zugriff auf die Daten
- Bildverarbeitungsaufgaben lassen sich auf der Datenstruktur lösen

Zusammenfassung

- speichereffizient und schneller Zugriff auf die Daten
- Bildverarbeitungsaufgaben lassen sich auf der Datenstruktur lösen
- sehr lange Laufzeiten der Algorithmen

Zusammenfassung

- speichereffizient und schneller Zugriff auf die Daten
- Bildverarbeitungsaufgaben lassen sich auf der Datenstruktur lösen
- sehr lange Laufzeiten der Algorithmen

Vielen Dank für Ihre Aufmerksamkeit