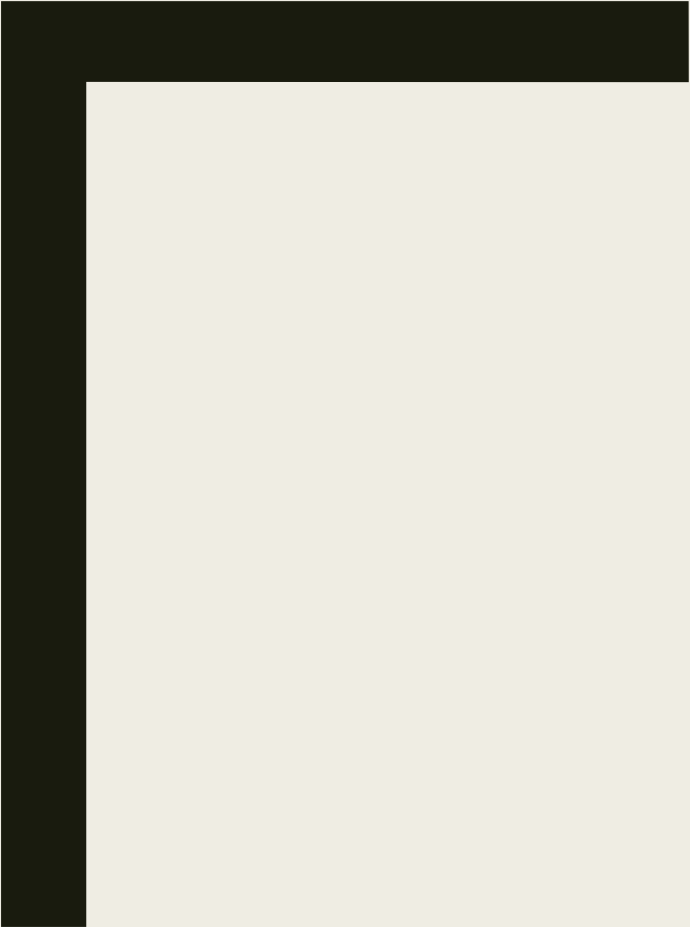



Wie können wir den Scann-Vorgang beschleunigen?



BWINF 36 RUNDE 2 AUFGABE 3

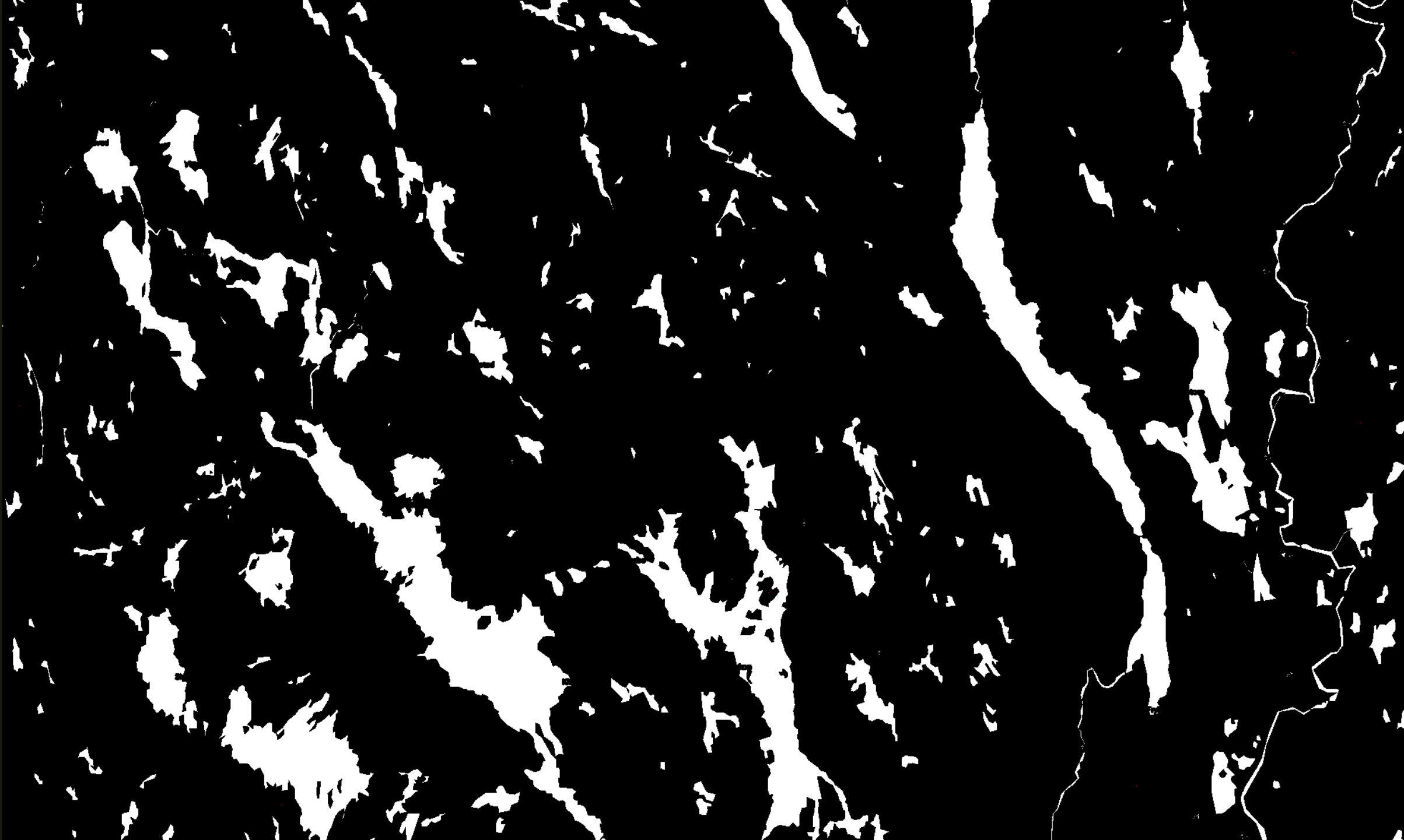
von Bennet Deboben
und Jonas Fritsch



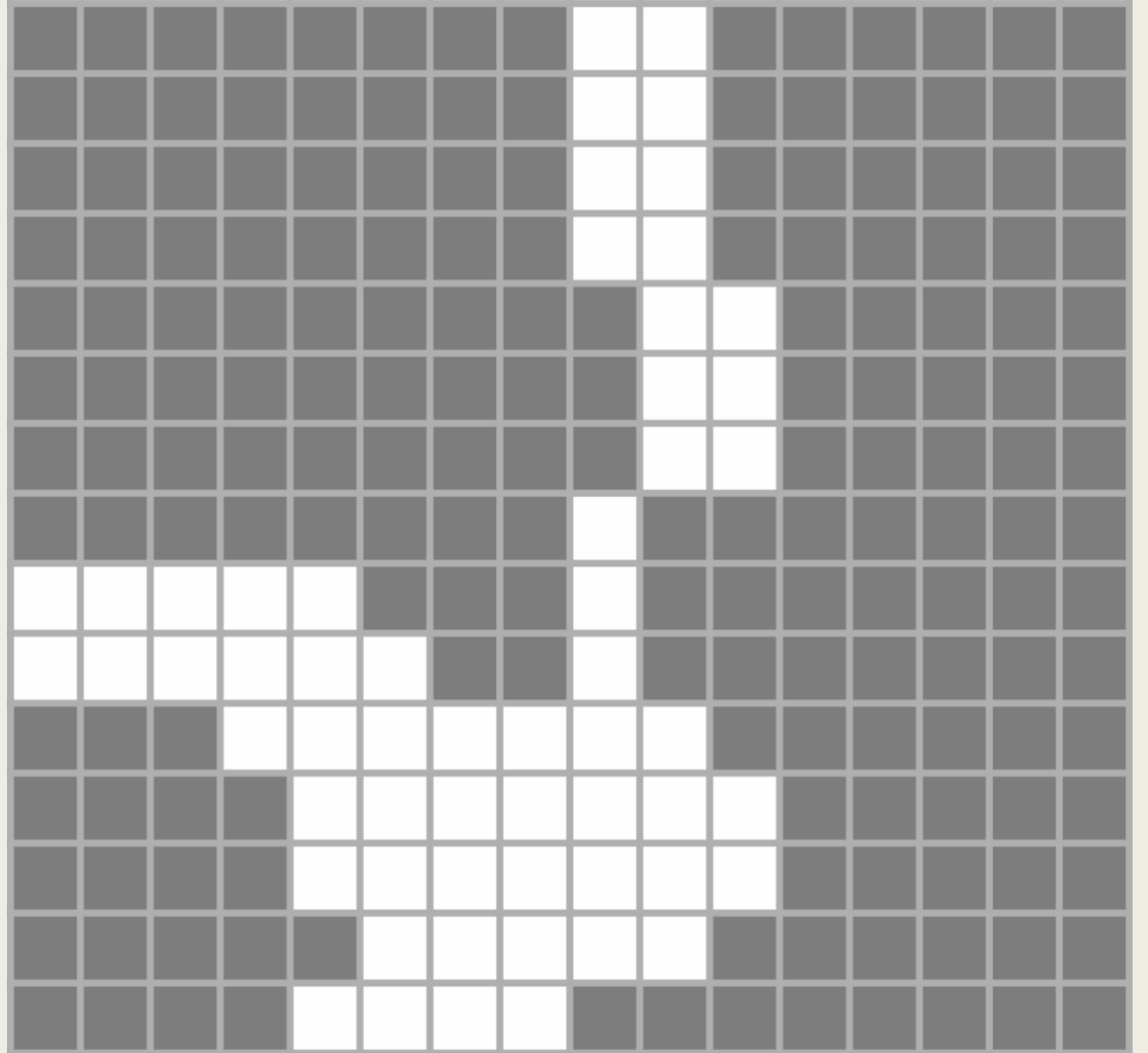
AUFGABENSTELLUNG

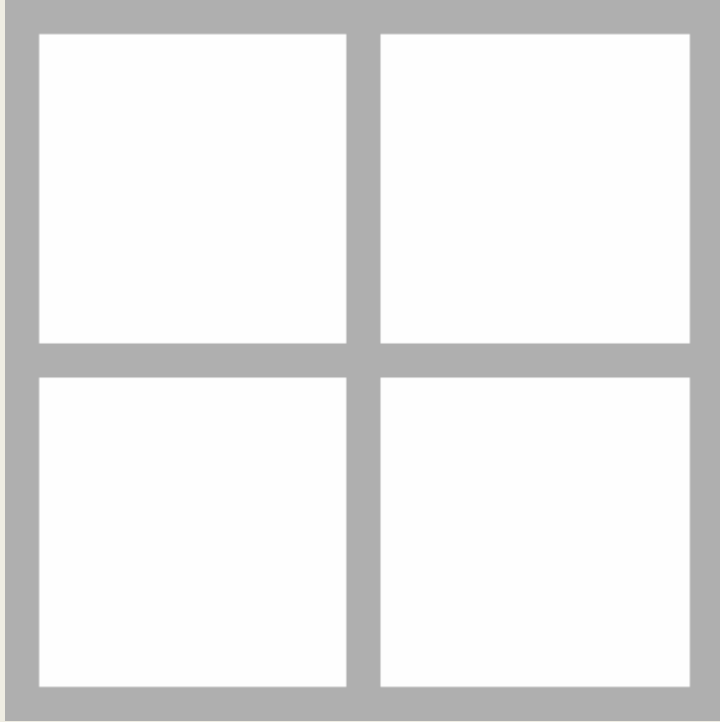




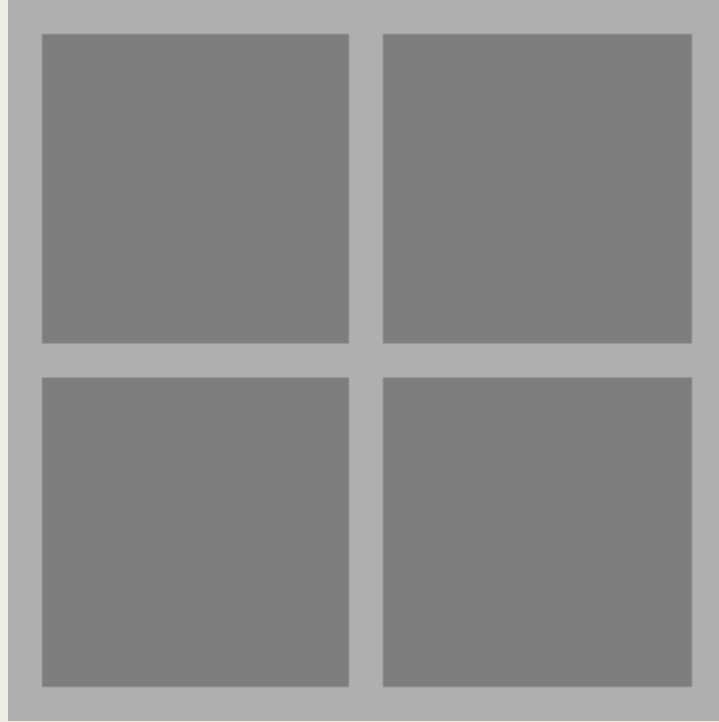


Fragen?

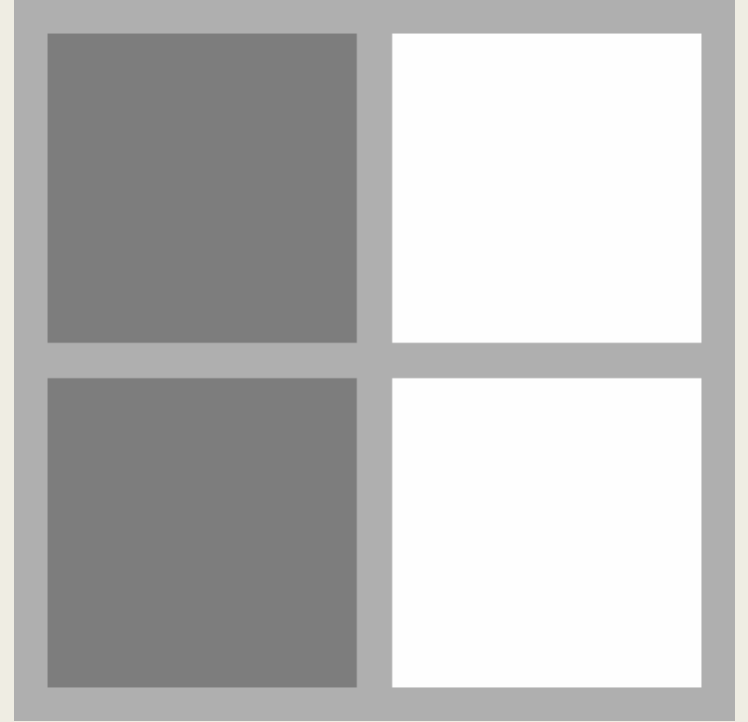




Wasser



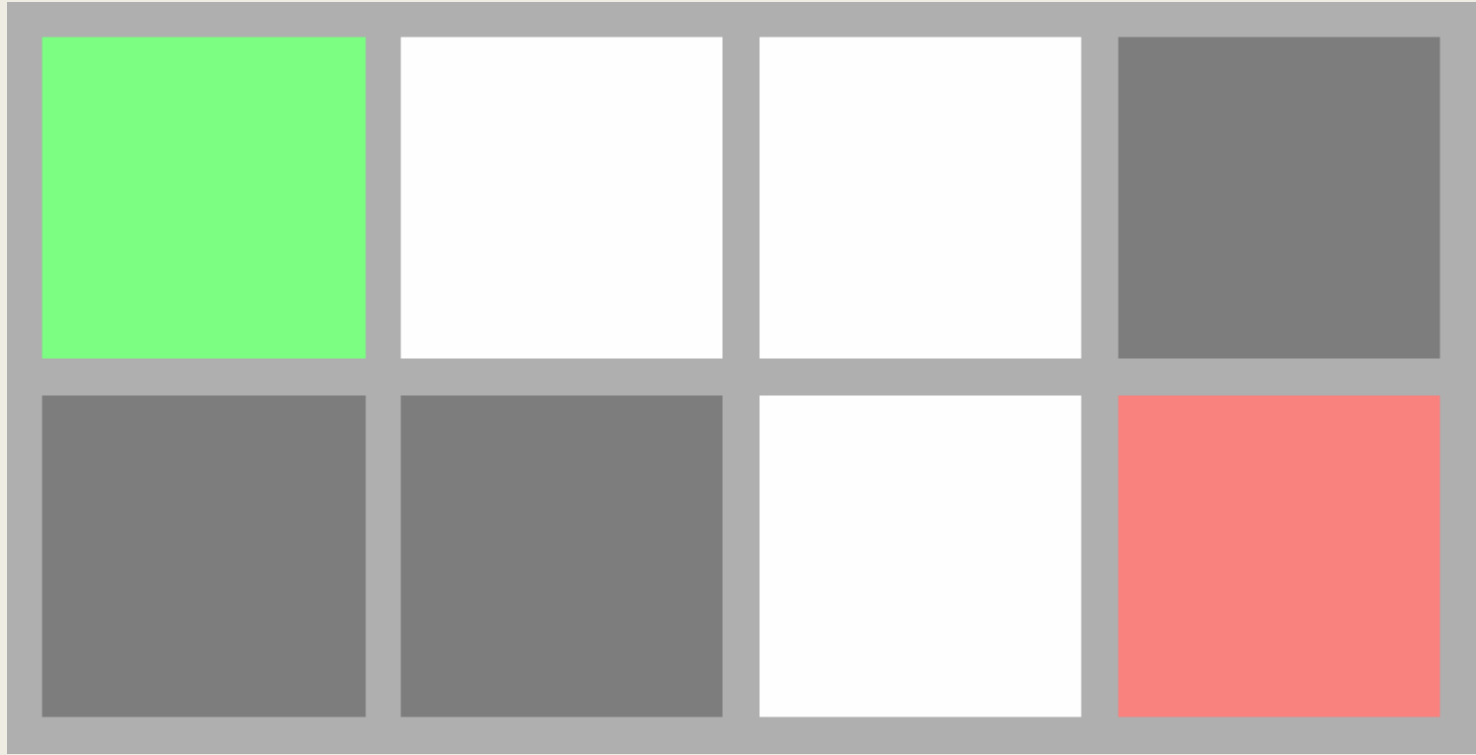
Land

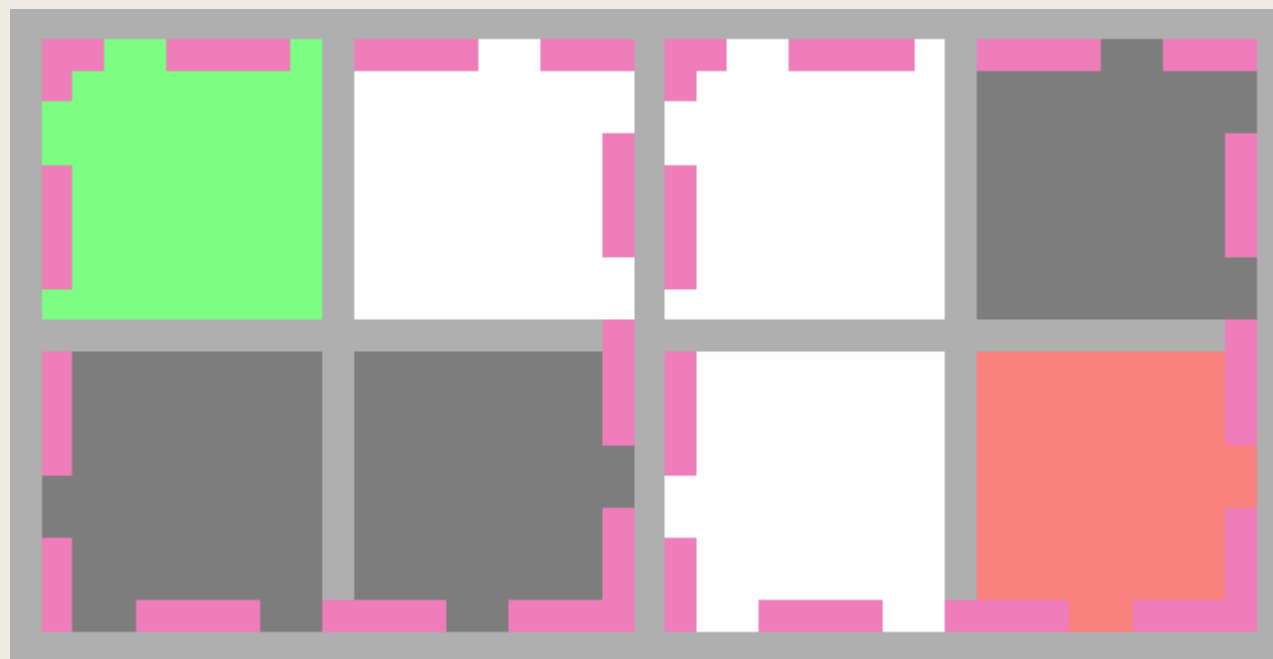
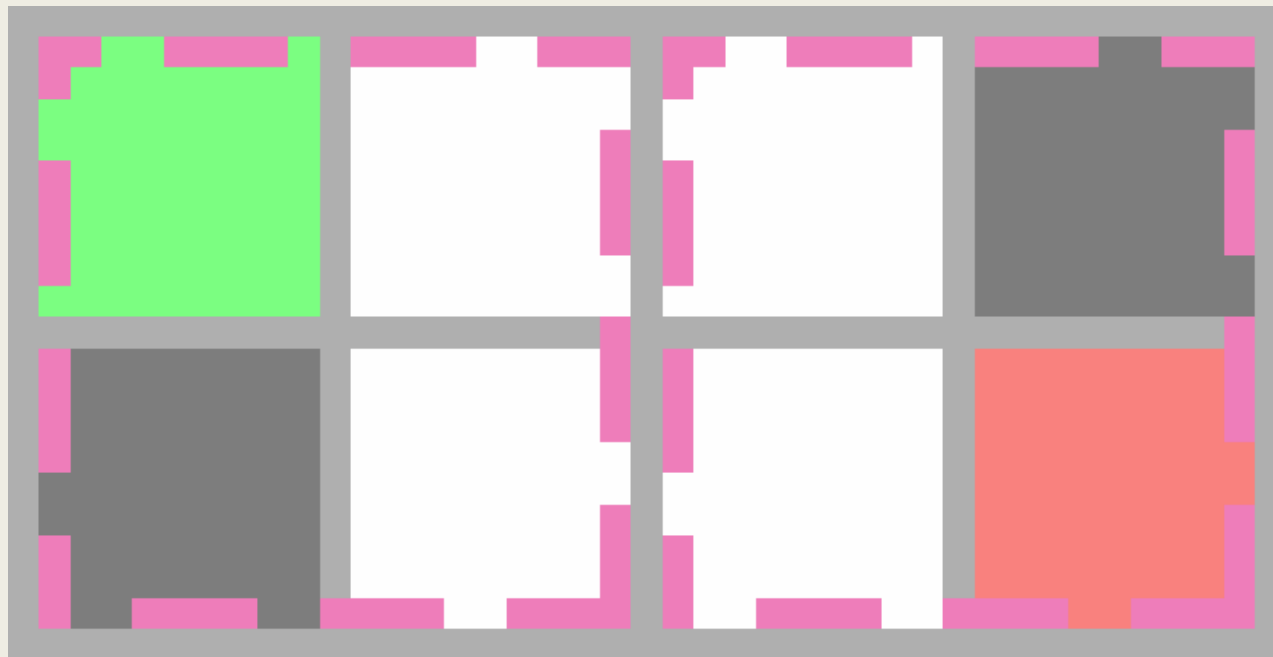


Gemischt

Kann Quax vom Grünen zum Roten Punkt?



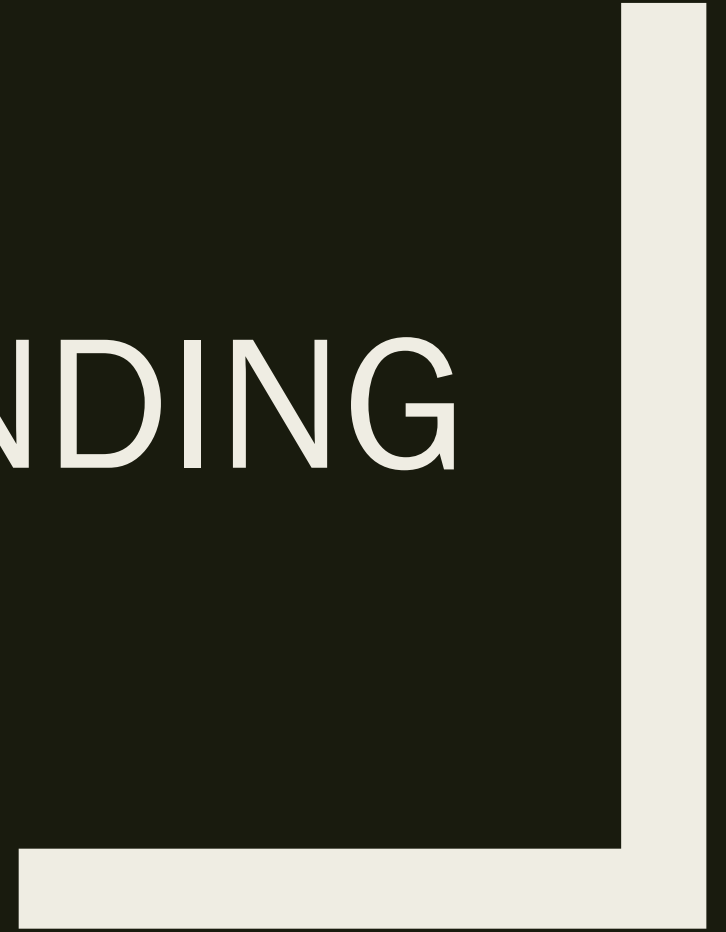




Ziel

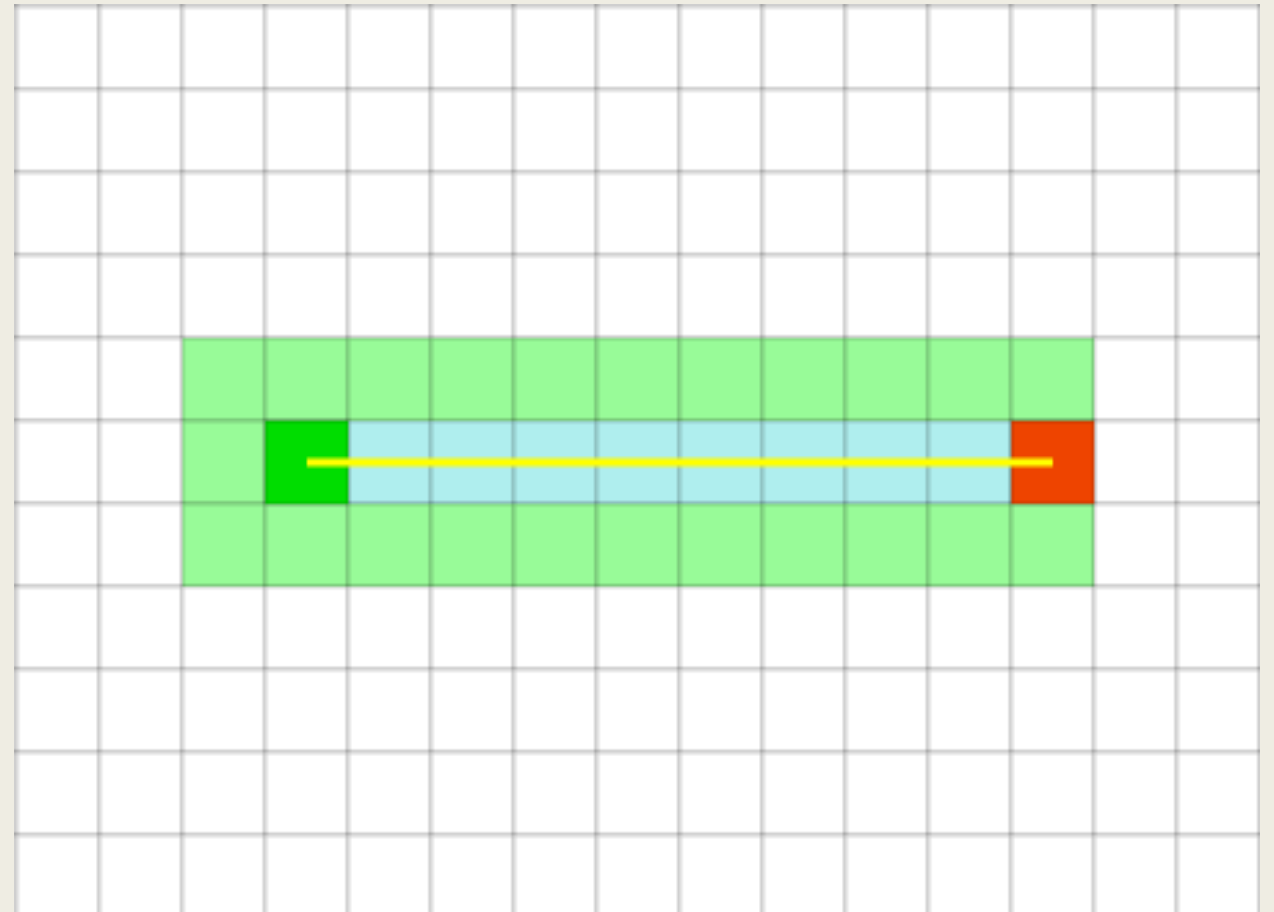
- Finden eines Weges
- Möglichst wenig Quadcopter-Flüge
- Sonderfall lösen
- Voraussetzungen:
 - *Wegfindungsalgorithmus*
 - *Optimierung dieses Algorithmus*
- (Teilaufgabe a gilt eher als Hilfestellung zu b)

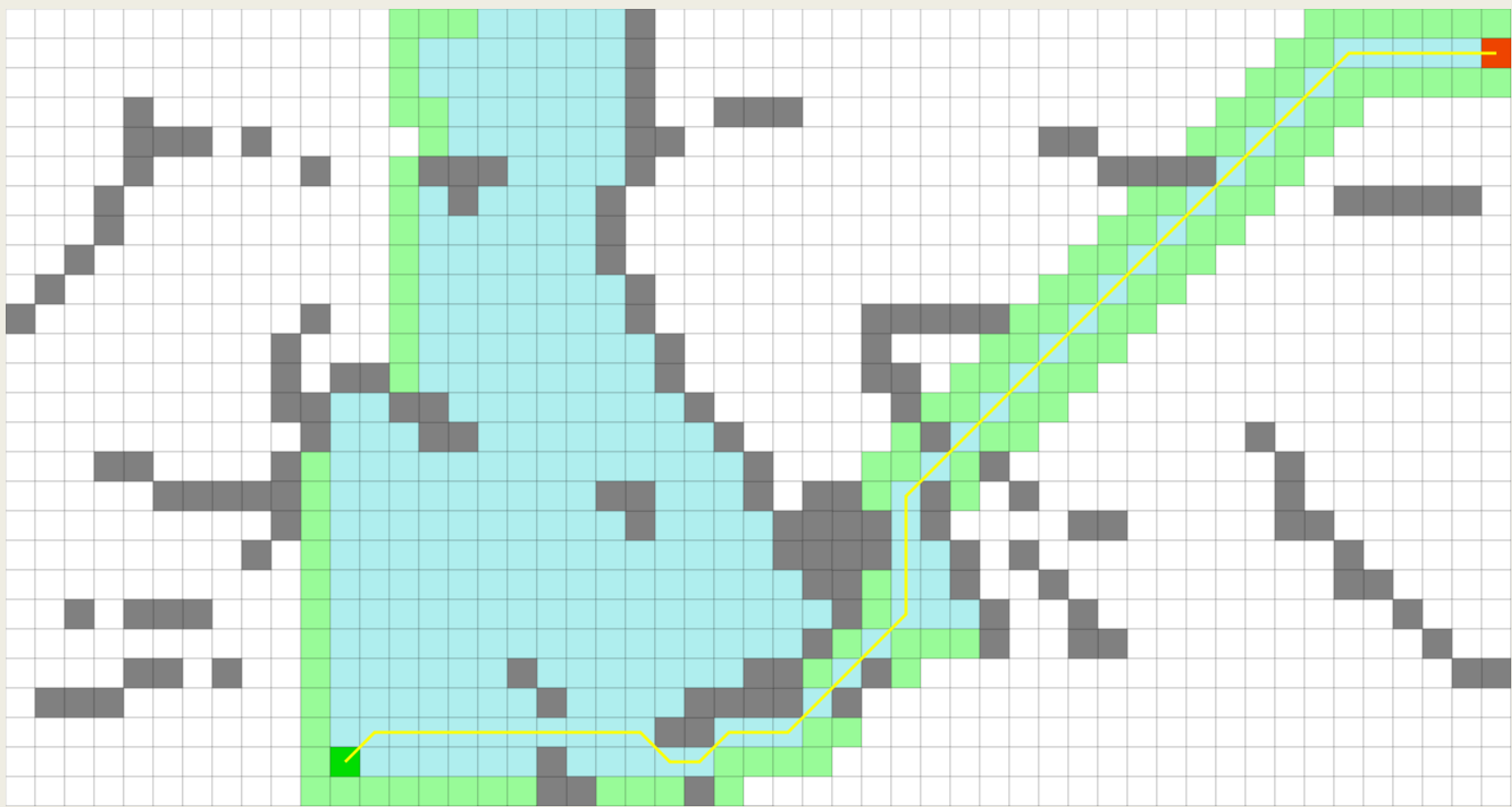
A*-PATHFINDING



Definition

- Wegfindungsalgorithmus zwischen mehreren Feldern (= Nodes) auf einer Karte
 - Es handelt sich um einen vollständigen und optimalen Algorithmus
- > Sollte ein Weg existieren, wird dieser auch gefunden





Funktionsweise - Entfernungen

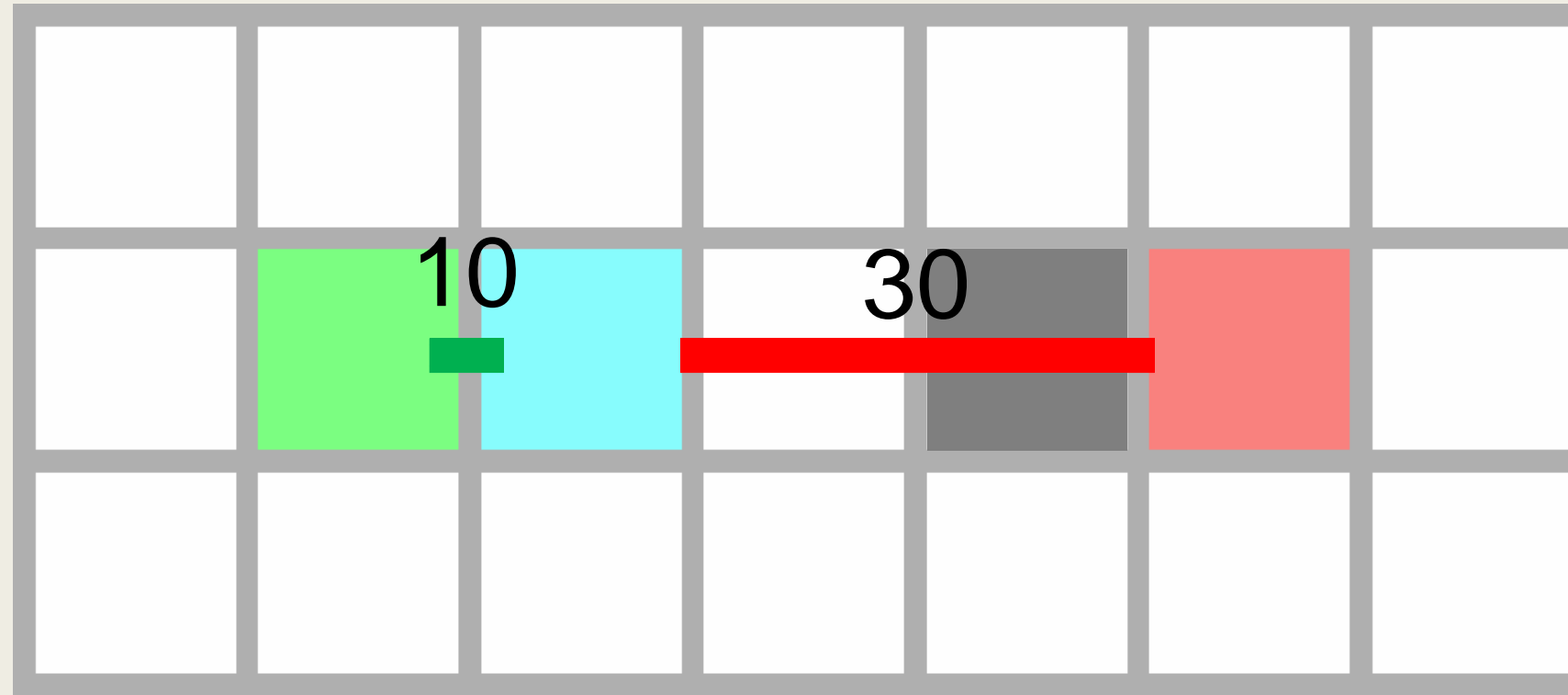
$\sqrt{2}$	1	$\sqrt{2}$
1		1
$\sqrt{2}$	1	$\sqrt{2}$

Funktionsweise - Entfernungen


14	10	14
10		10
14	10	14

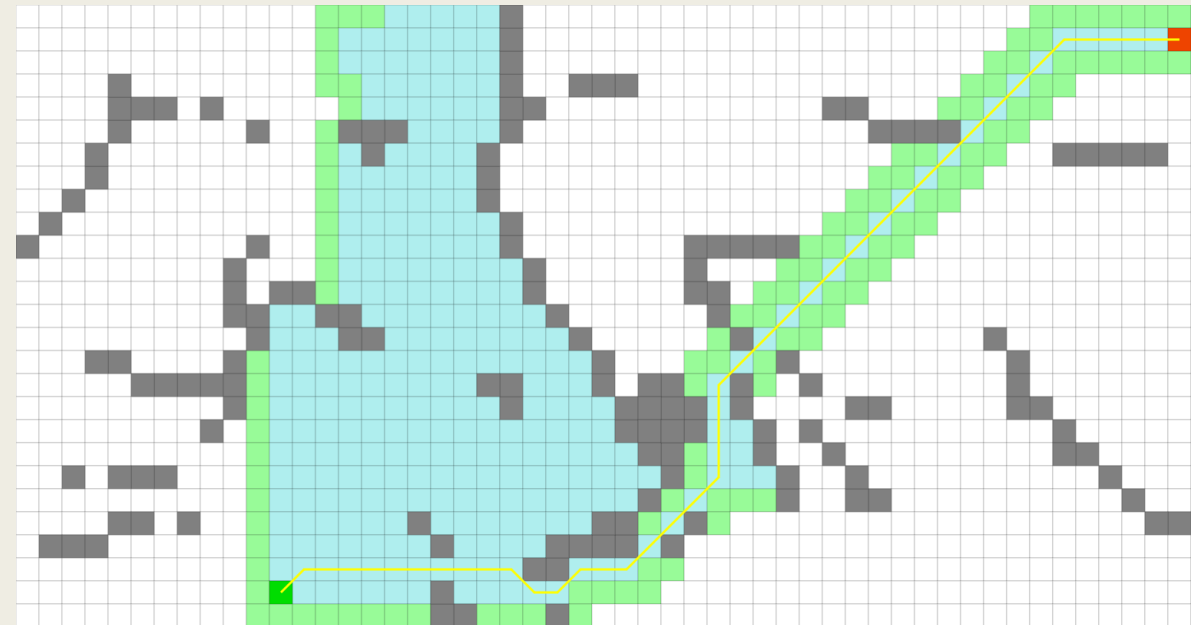
Funktionsweise - Nodes

- G-Kosten: Entfernung vom Startpunkt
- H-Kosten: Entfernung vom Zielpunkt (geschätzt)
- F-Kosten: $G + H$
- Zeiger auf Parent-Node

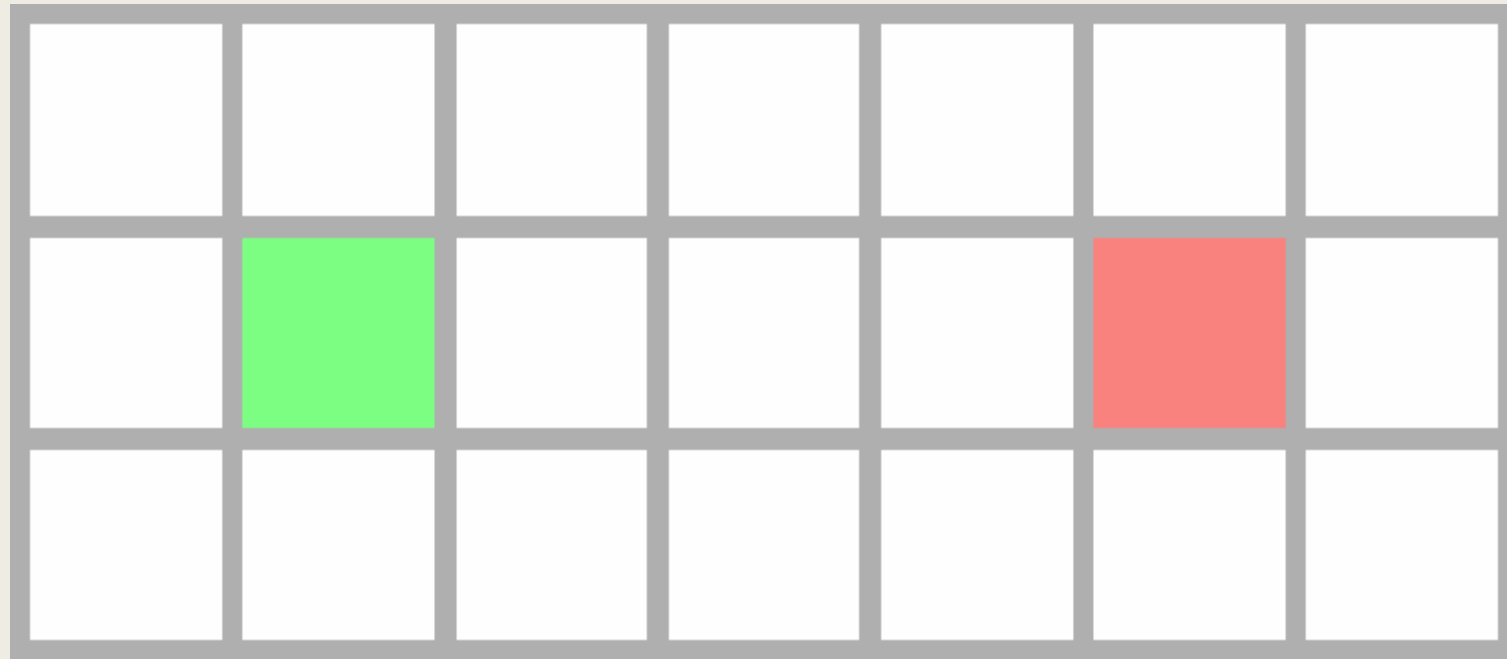


Funktionsweise - Listen

- Unbekannte Knoten
 - OpenList: Bekannte Nodes zu denen ein (möglicherweise suboptimaler) Pfad bekannt ist.
 - *Zu Beginn ist der Start-Node in dieser Liste*
 - ClosedList: Nodes zu denen der beste Weg bekannt ist
 - *Keine Mehrfachberechnung*
- 



Funktionsweise - Beispiel



Funktionsweise

	54	48				
		40				
	54	48				

Funktionsweise

	54	48	48			
			40			
	54	48	48			

Funktionsweise

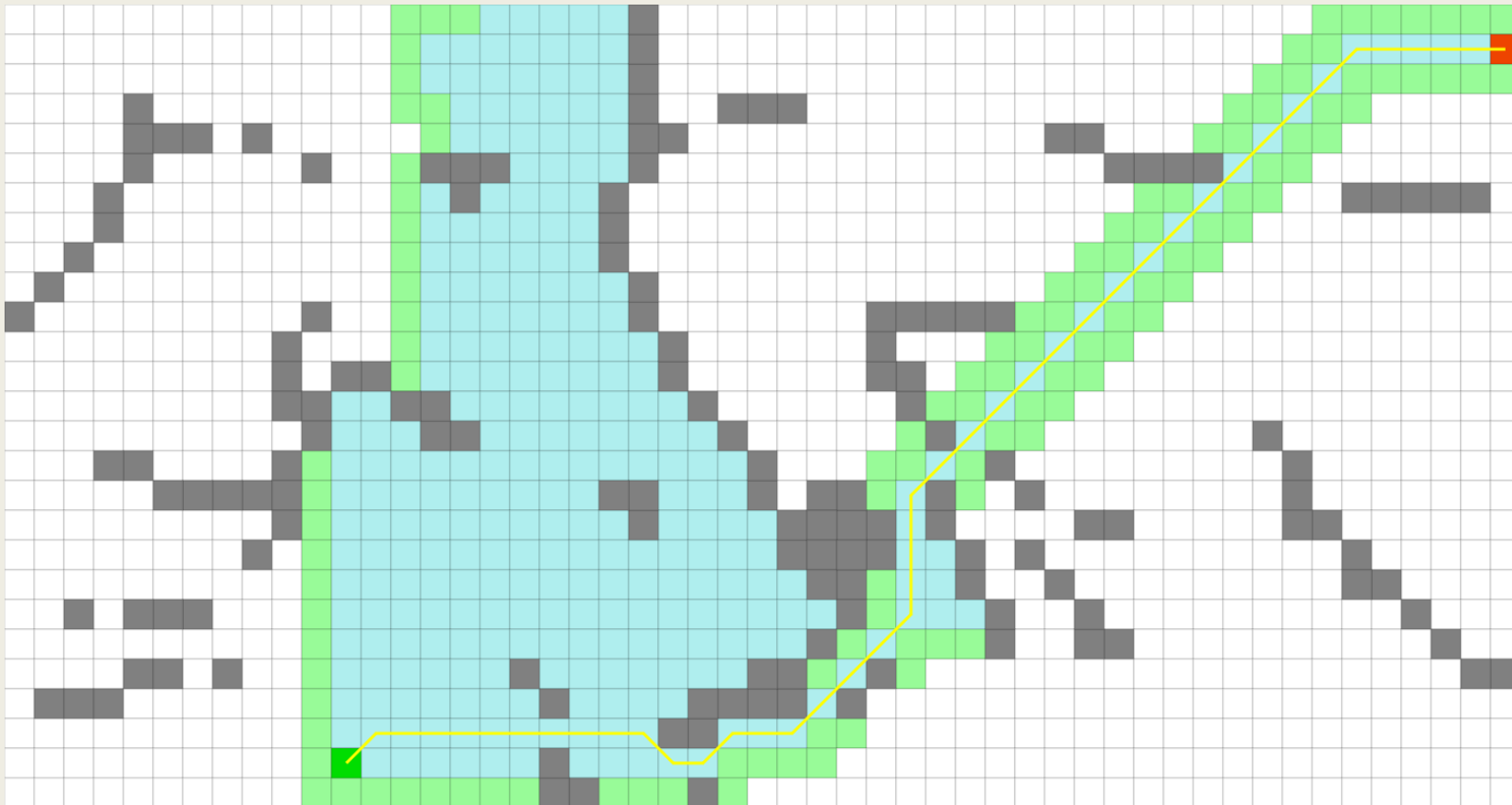
	54	48	48	48		
				40		
	54	48	48	48		

Funktionsweise

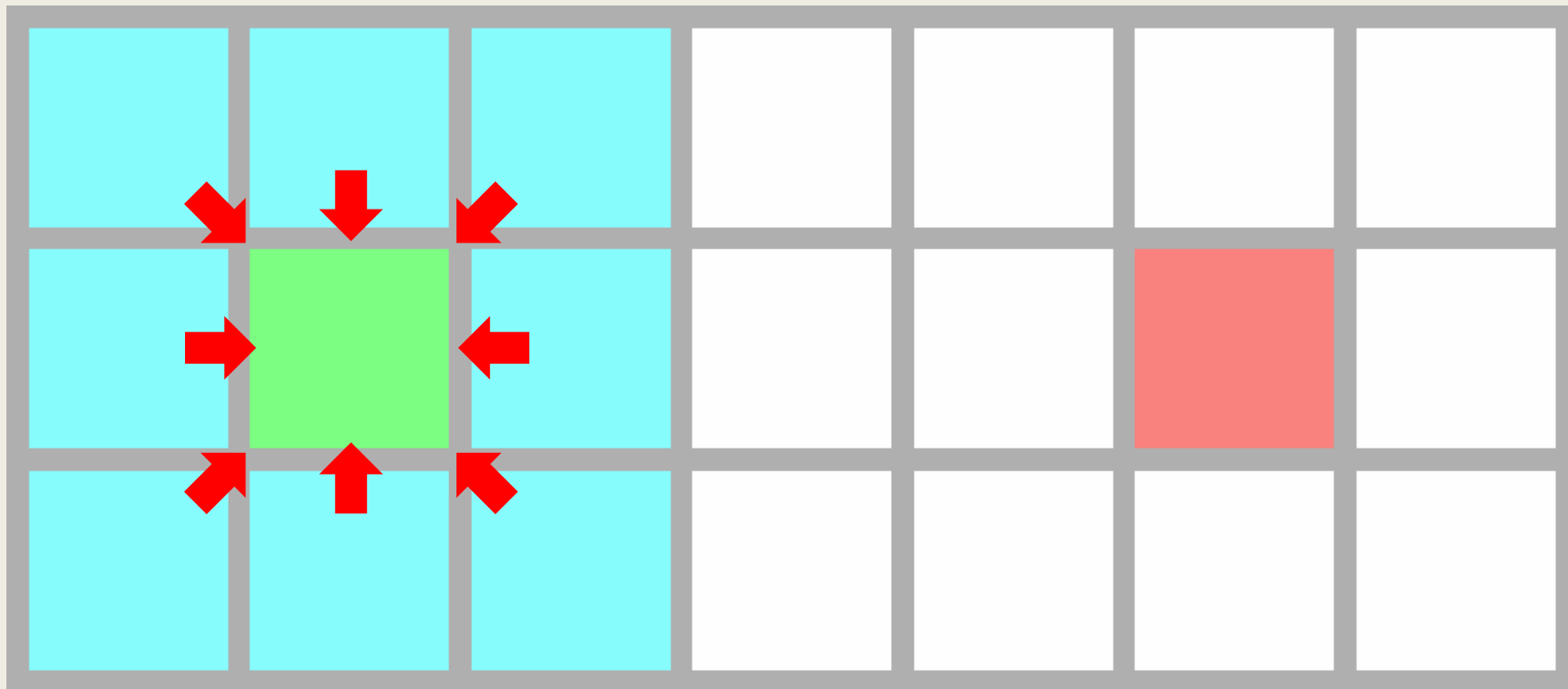
	54	48	48	48		
	54	48	48	48		

Funktionsweise

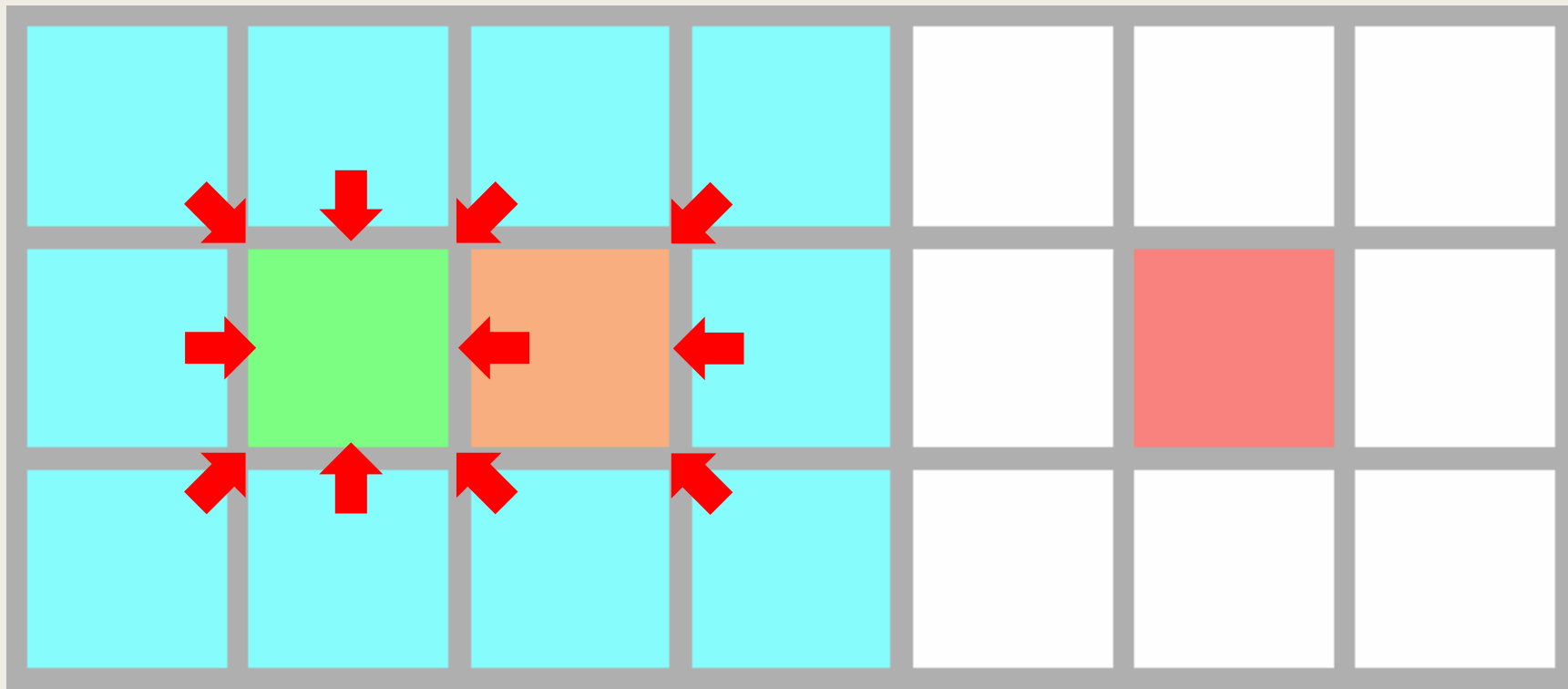
Woher weiß ich wie bei mehreren Nodes in der ClosedList der Pfad aussieht?



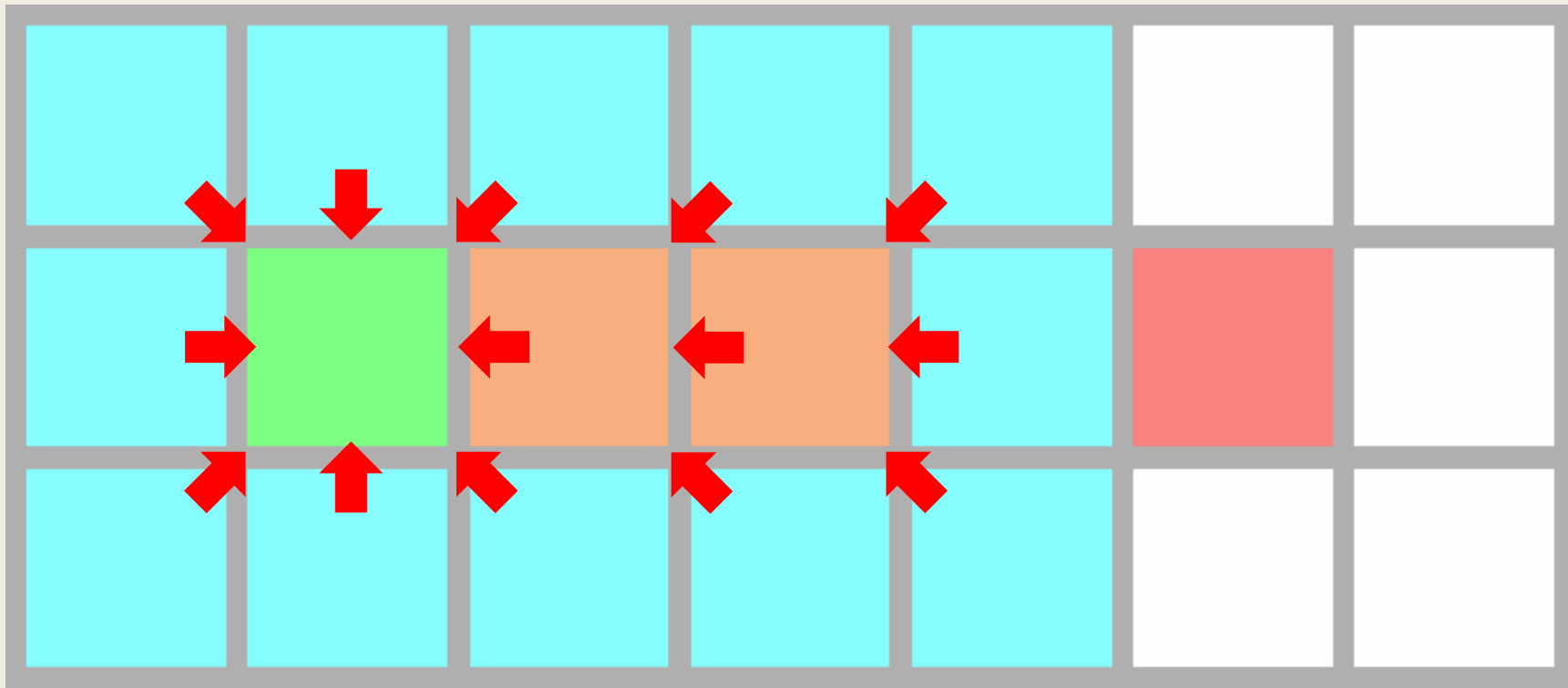
Funktionsweise



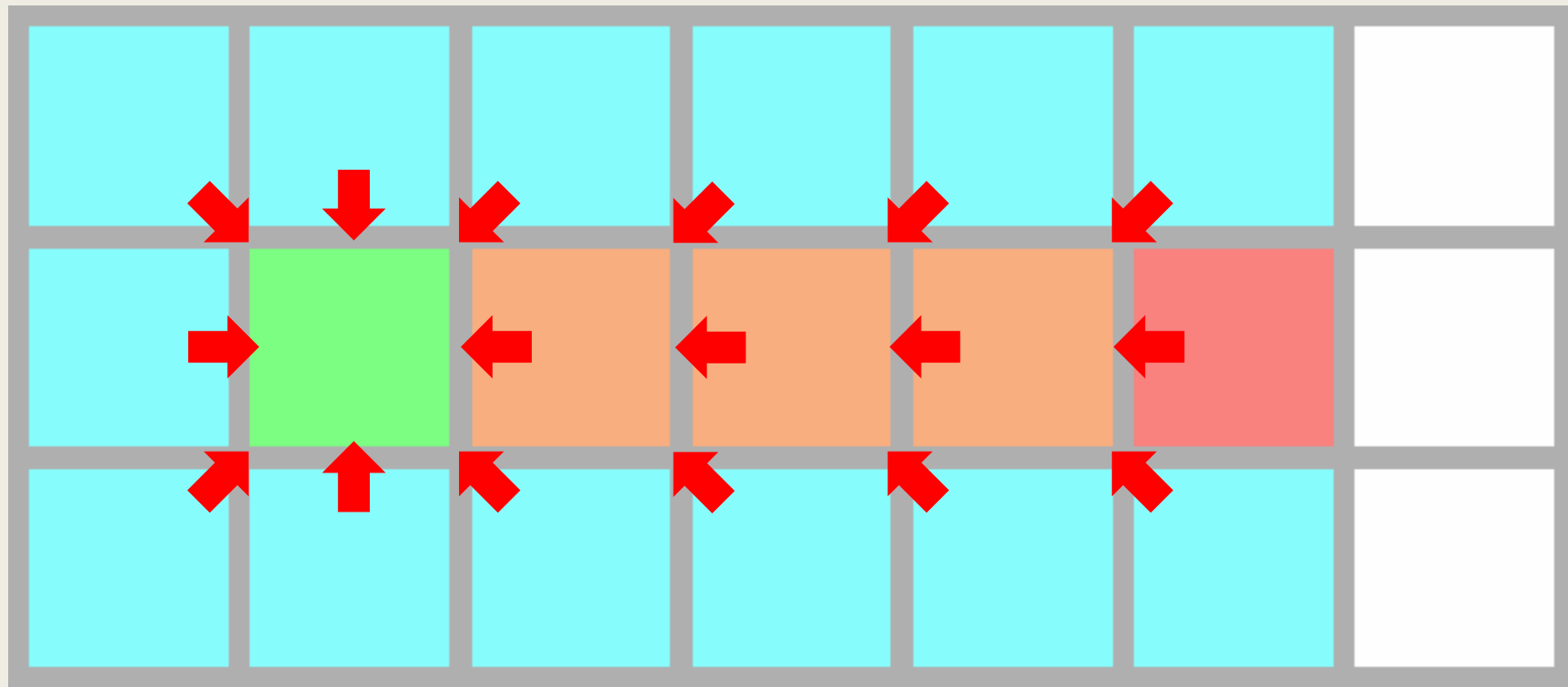
Funktionsweise



Funktionsweise



Funktionsweise



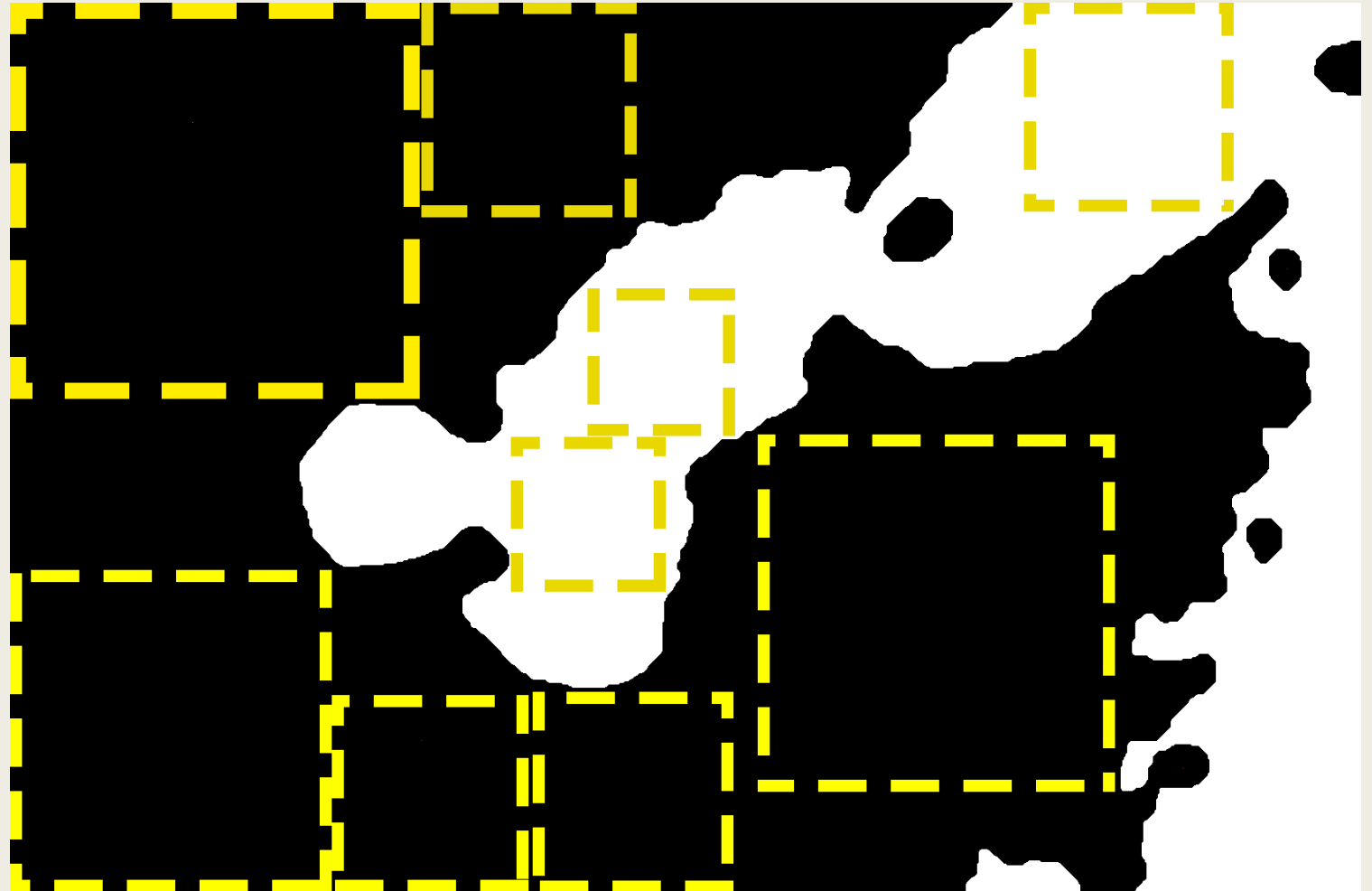
WIESO KEIN EINFACHES
A* VERWENDEN?

1674 x 1102 Pixel

$$\left\lceil \frac{w}{2} \right\rceil \cdot \left\lceil \frac{h}{2} \right\rceil$$

$$\left\lceil \frac{1672}{2} \right\rceil \cdot \left\lceil \frac{1102}{2} \right\rceil = 461.187$$

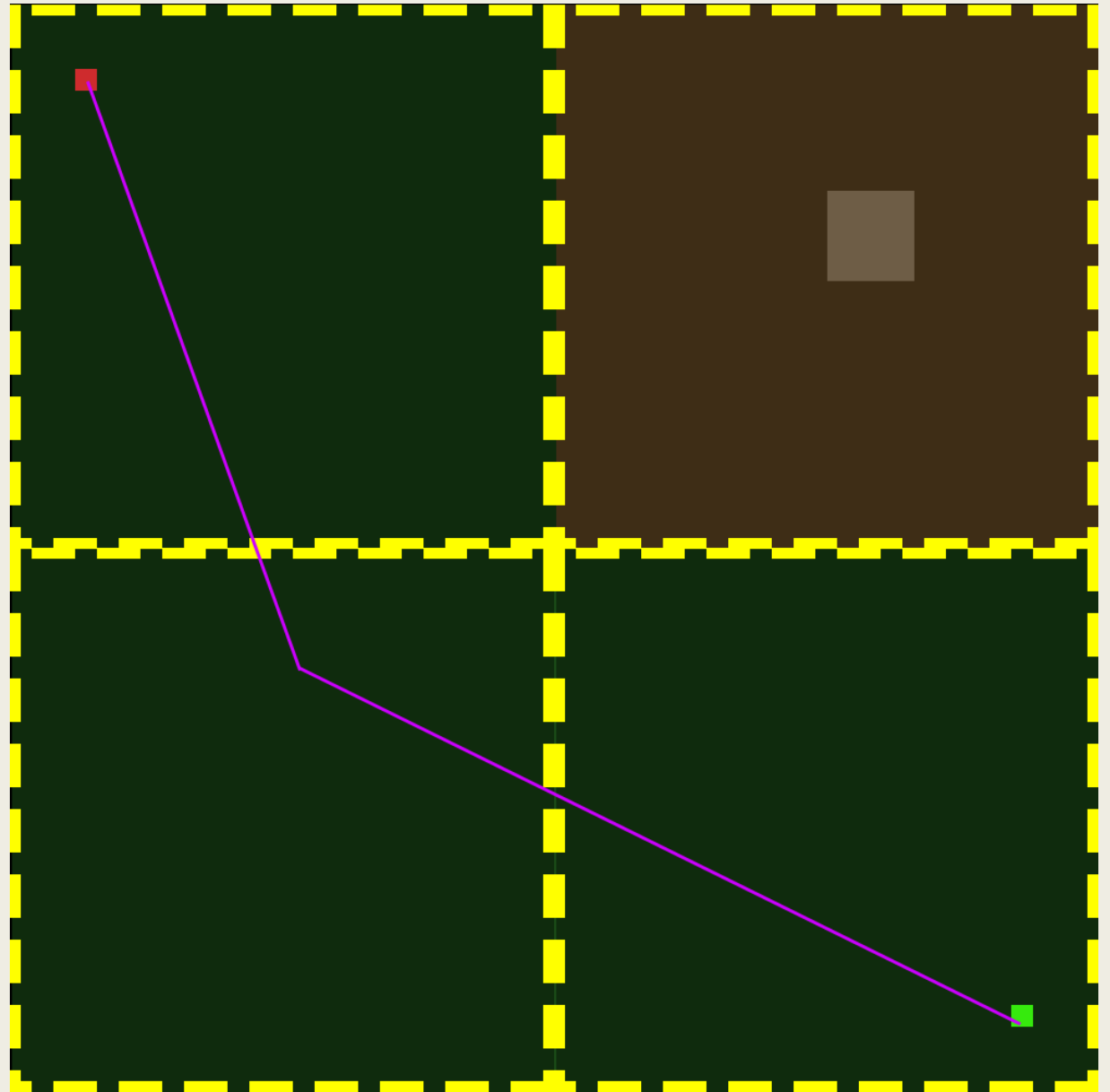
Einsparen von 60 tausend Flügen!



WIE SCHAFFEN WIR
MÖGLICHST WENIG
FLÜGE?



Wenn Quadrat $> 20 \times 20$ und **Gemischt**
→ Aufteilung in kleinere Teilquadrate

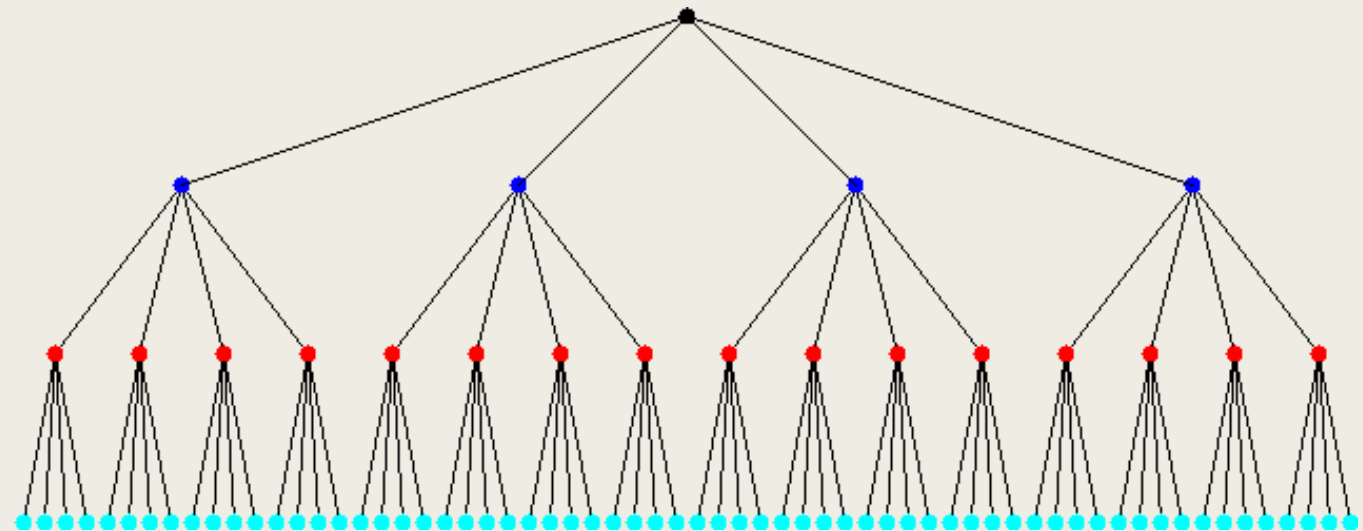
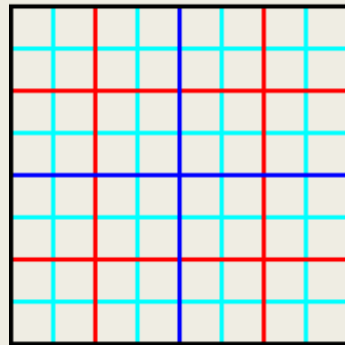


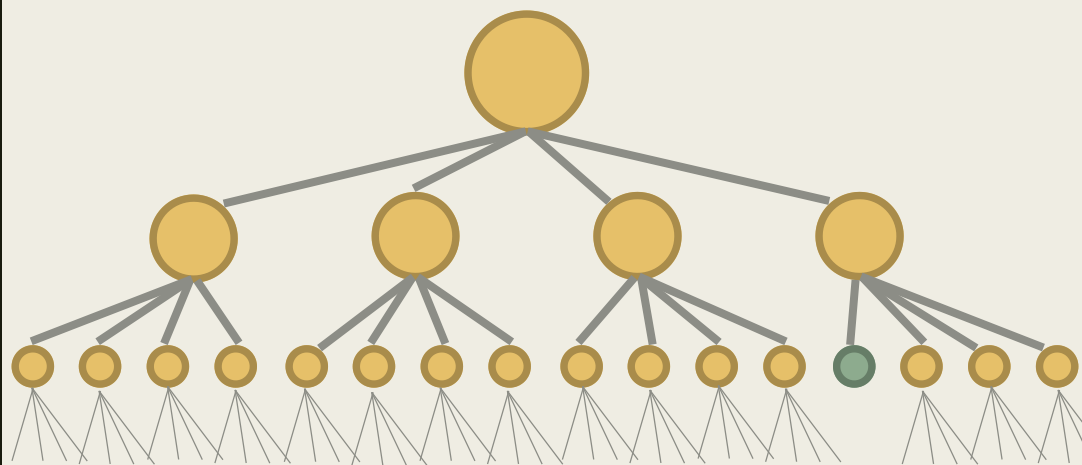
DATENSTRUKTUR QUADTREE



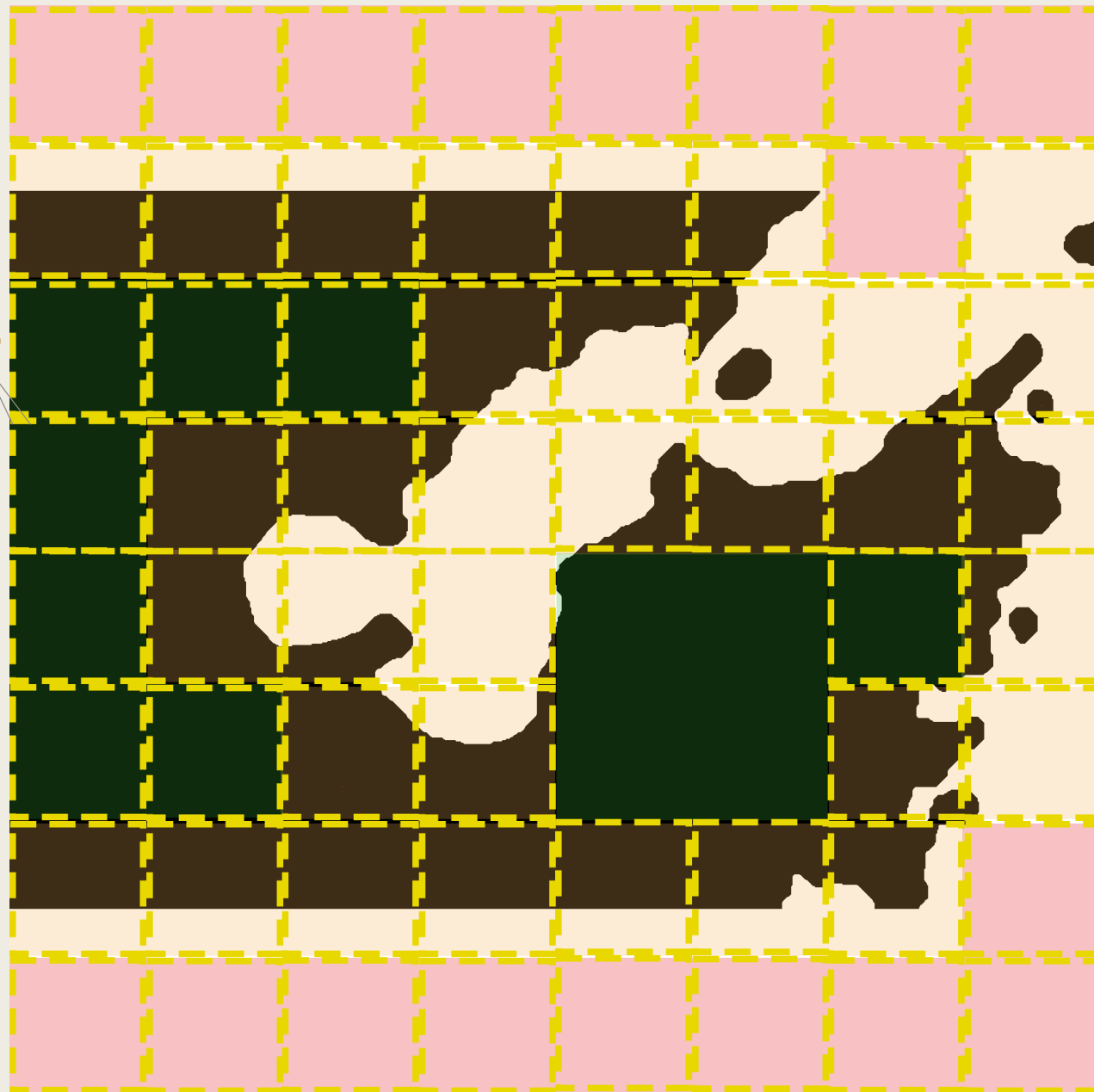
Datenstruktur: Quadtree

- Baum-Datenstruktur
- Jeder innere Knoten hat genau 4 Tochterknoten
- Sehr effizient
- Sehr platzsparend



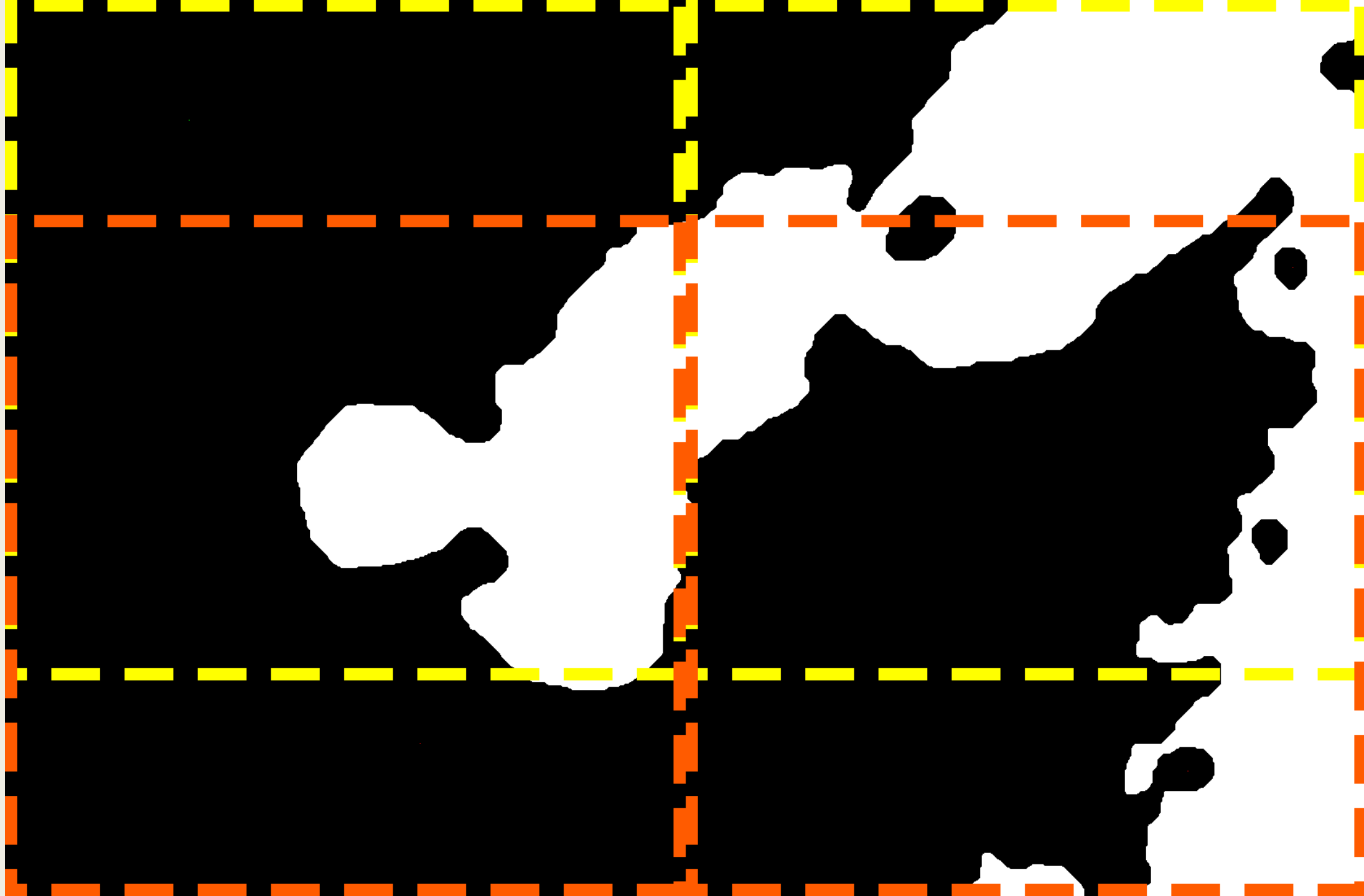


...

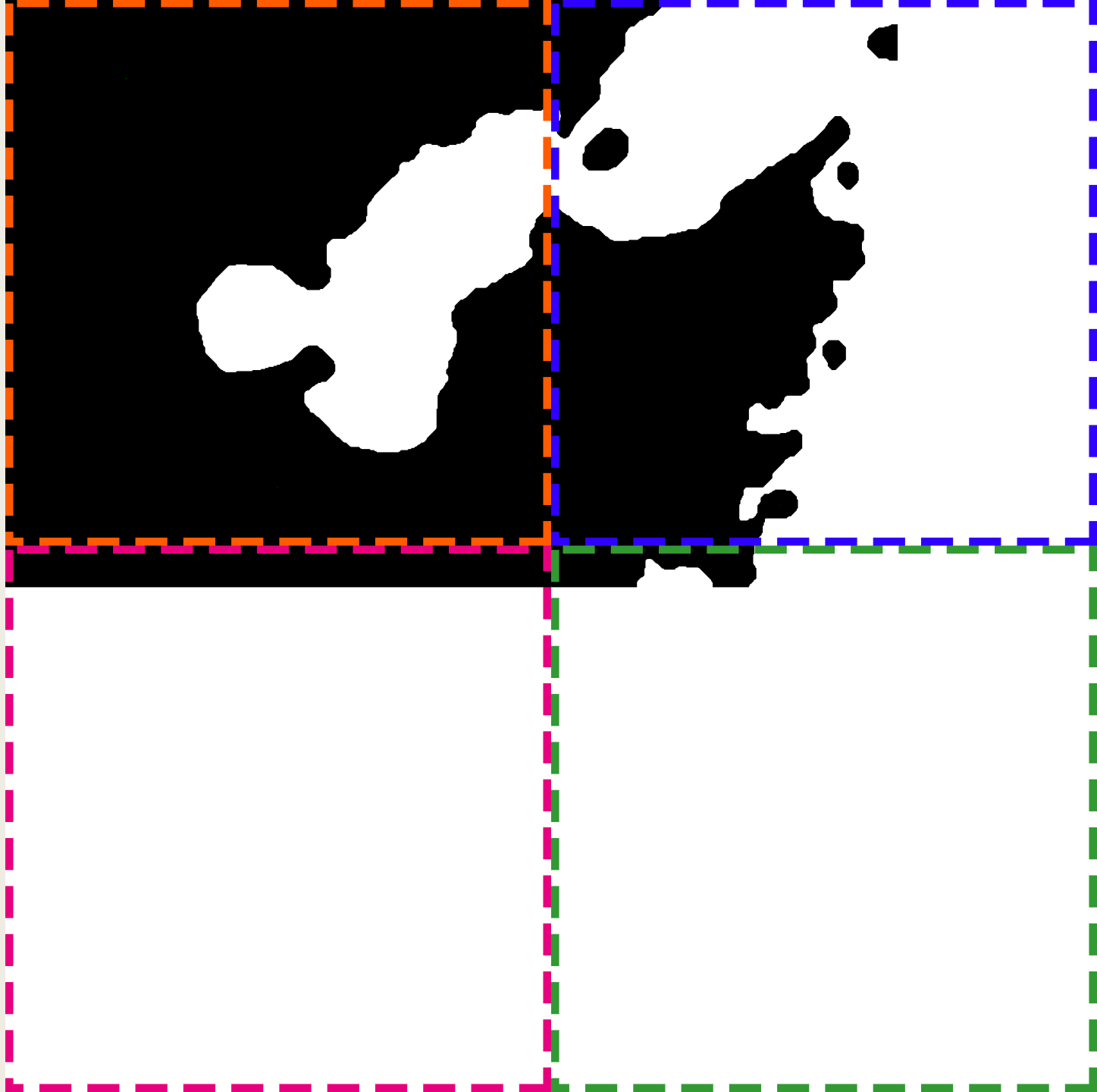


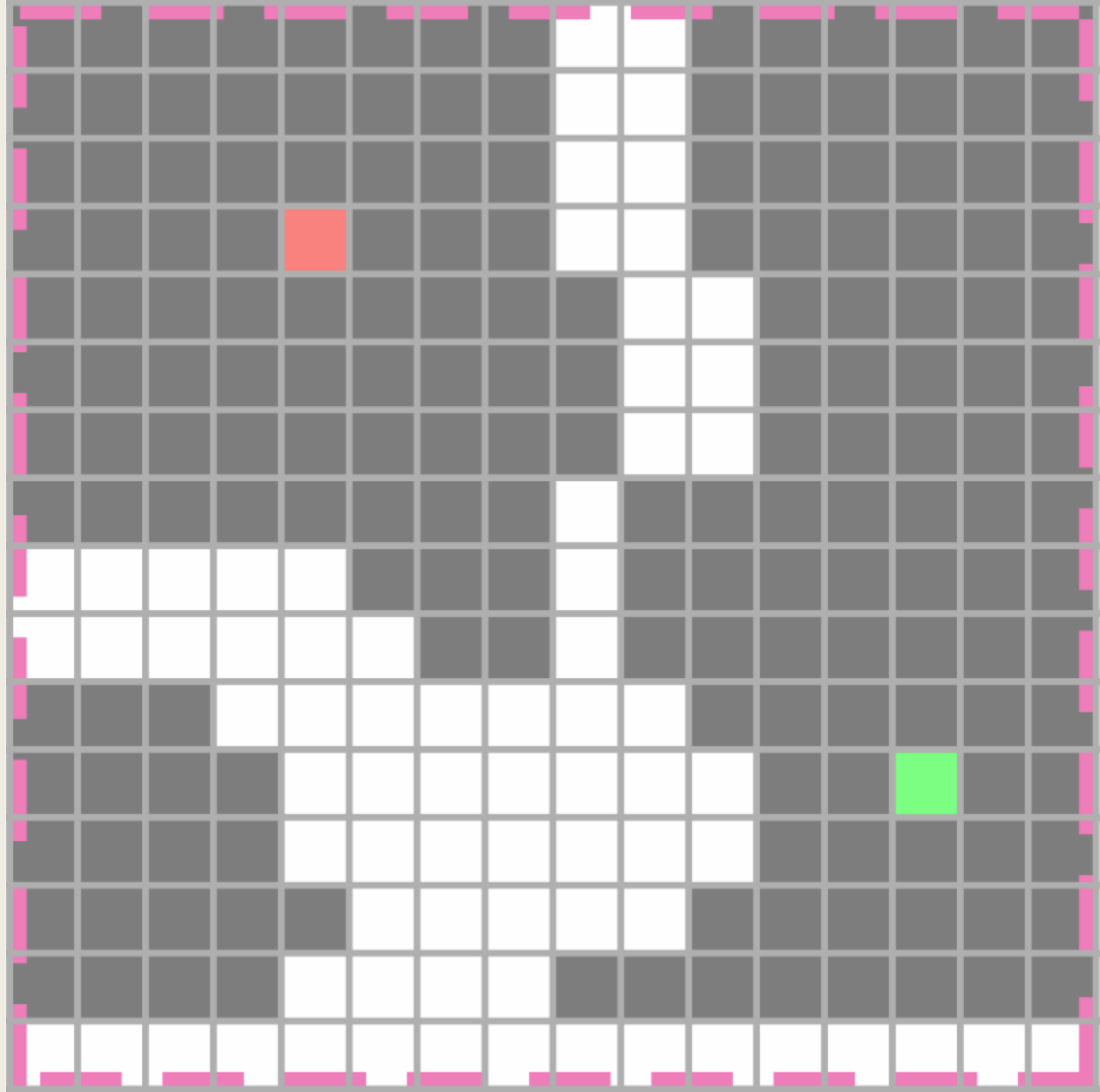
LÖSUNG: BENNET

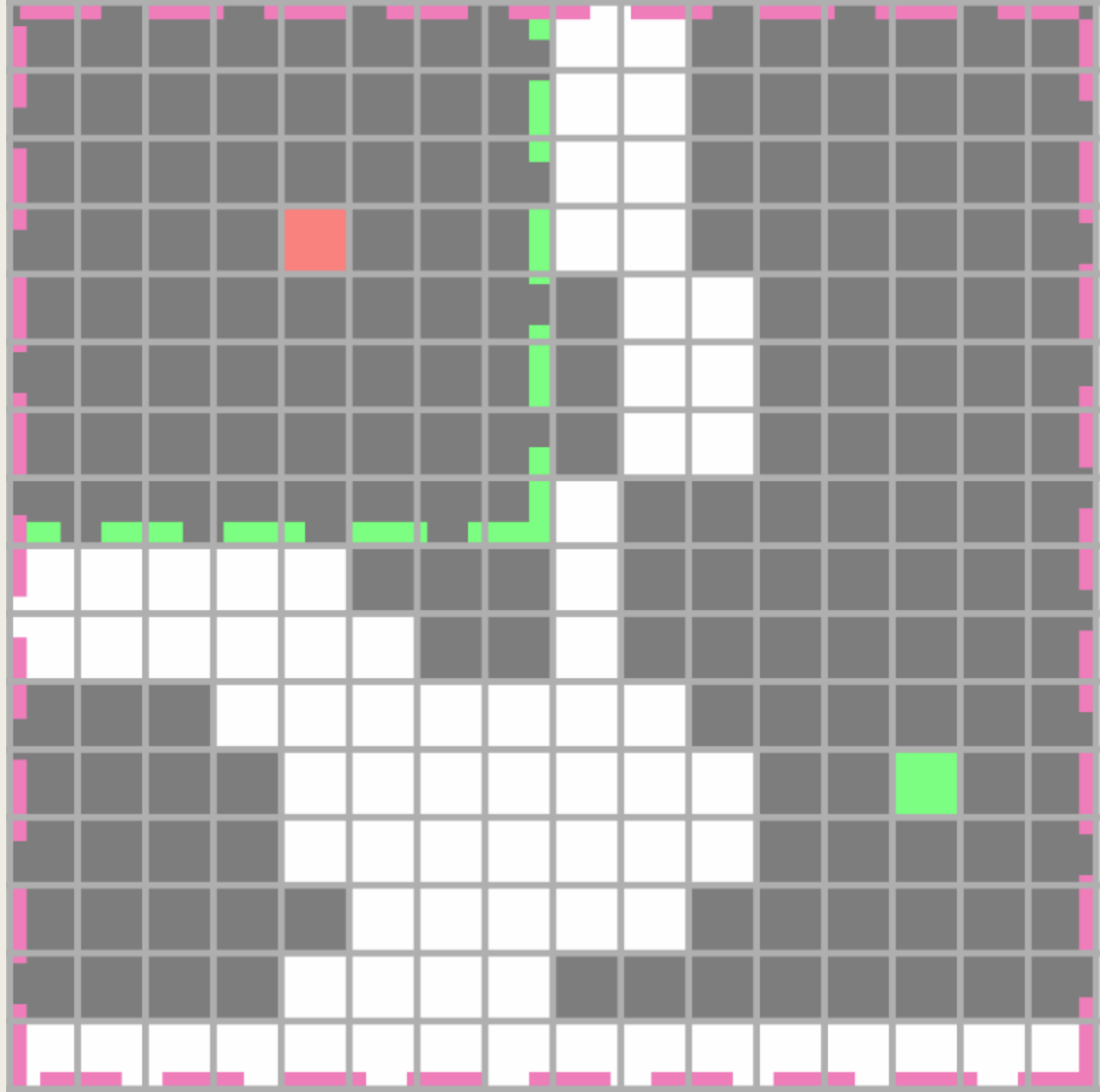


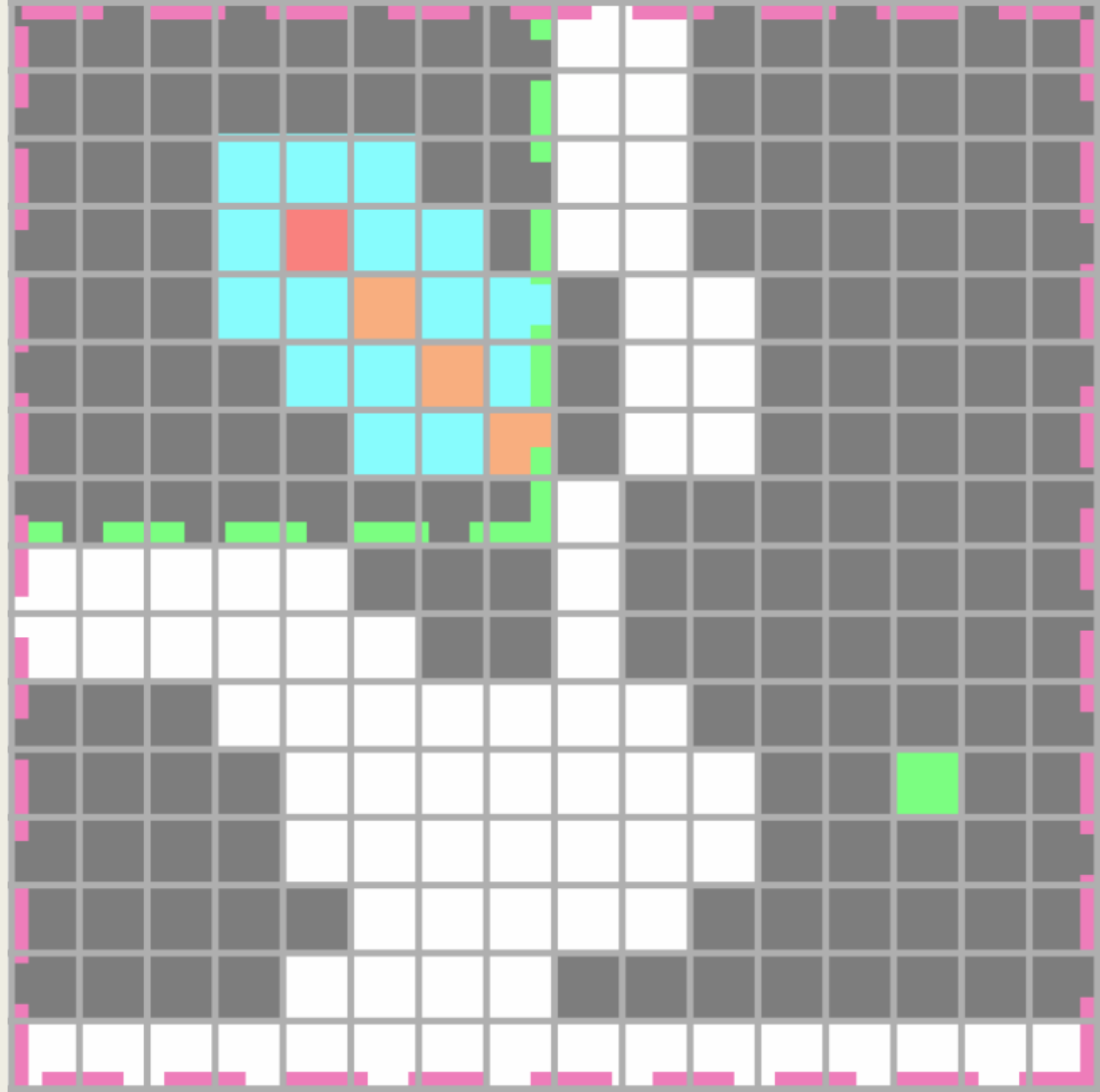


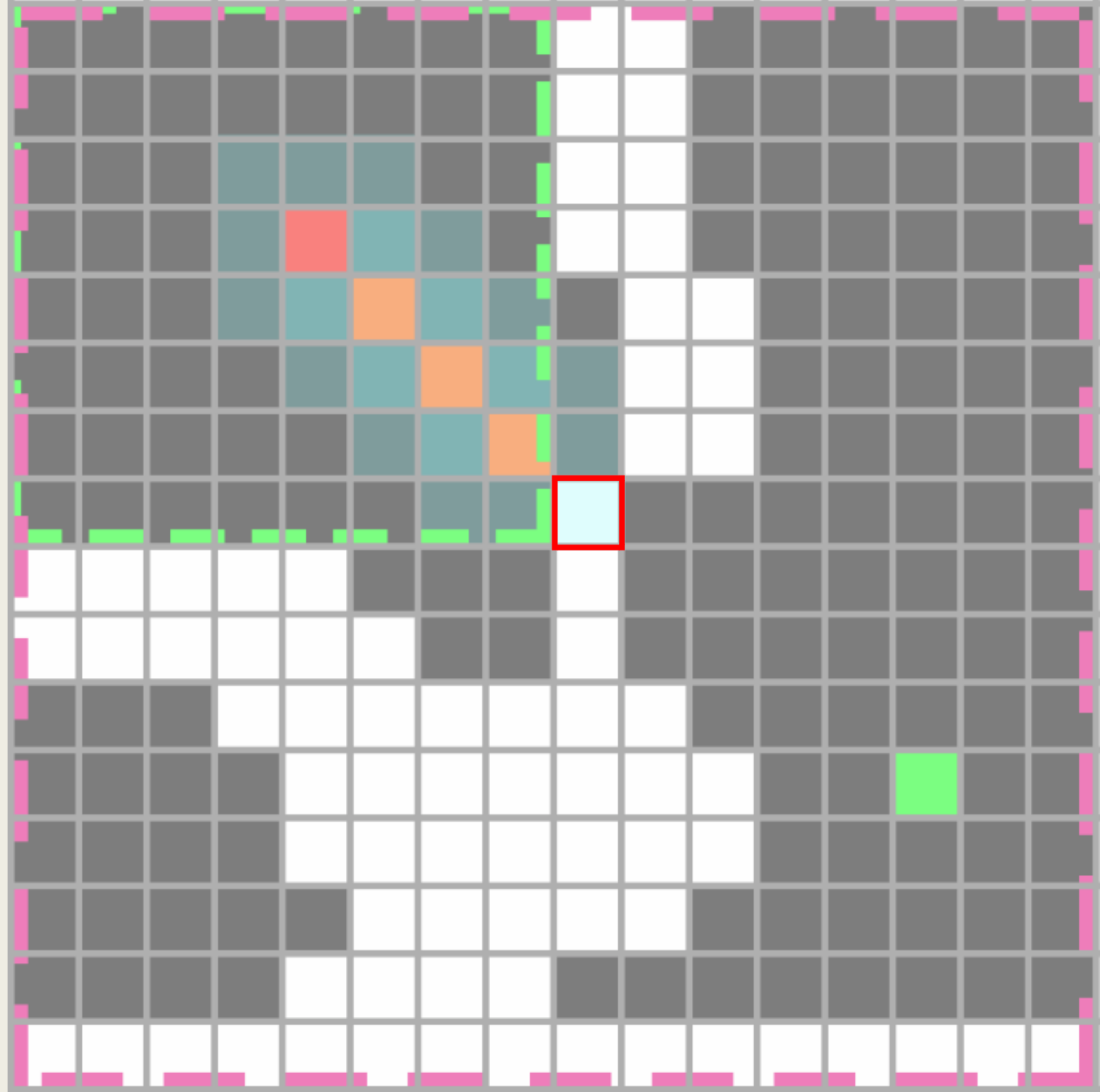
Erweitern der Karte auf ein
Vielfaches von 2

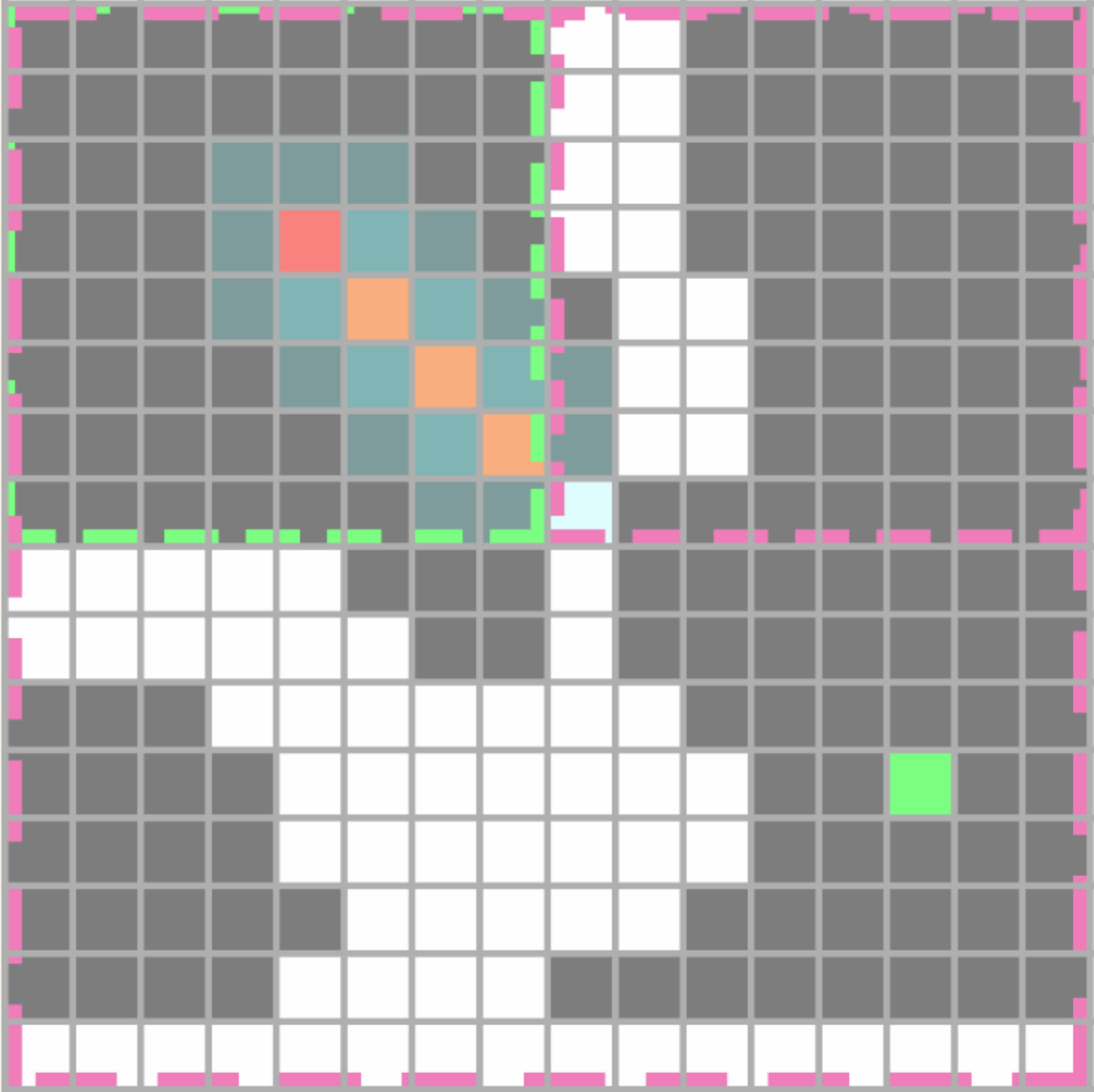


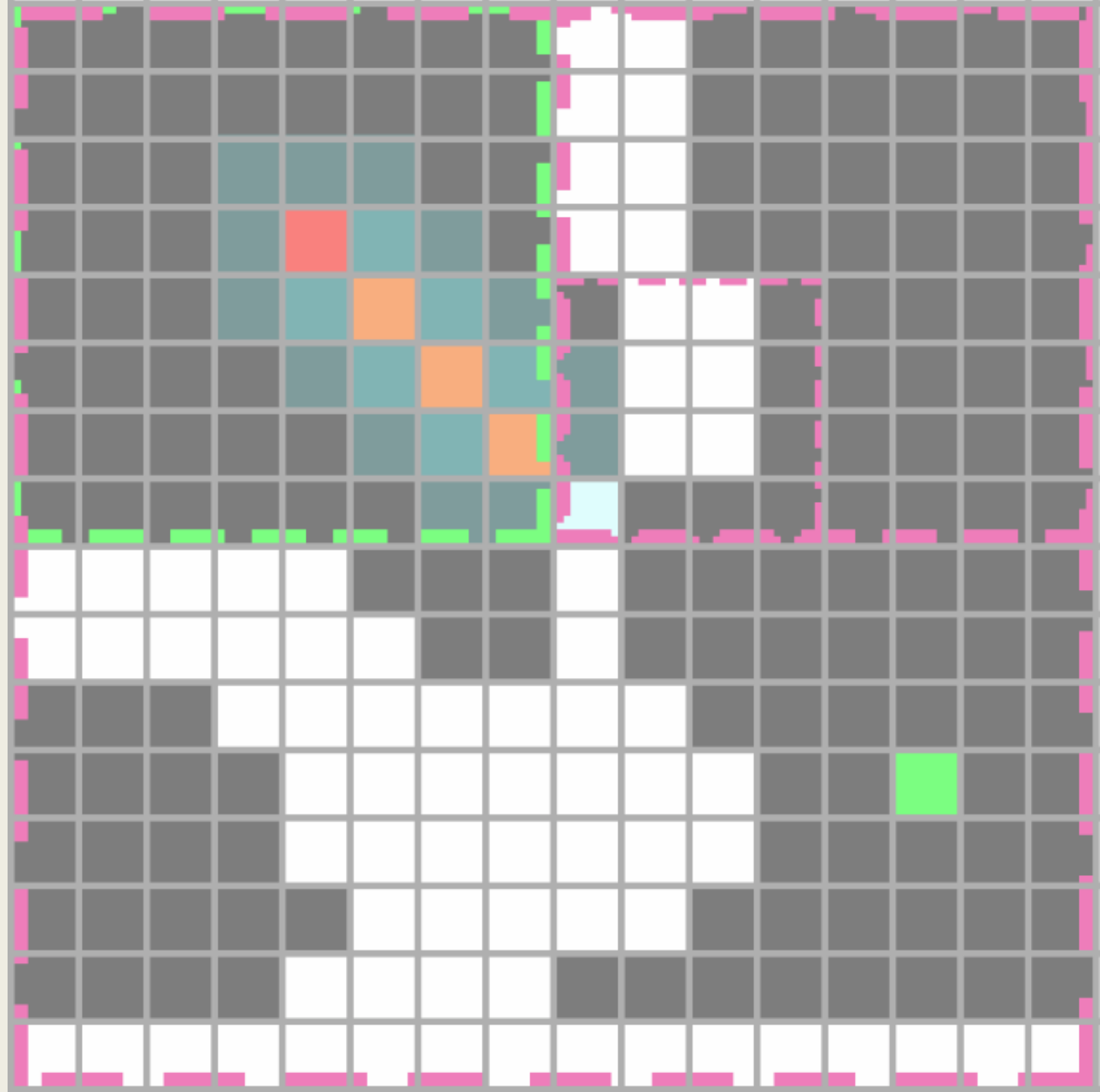


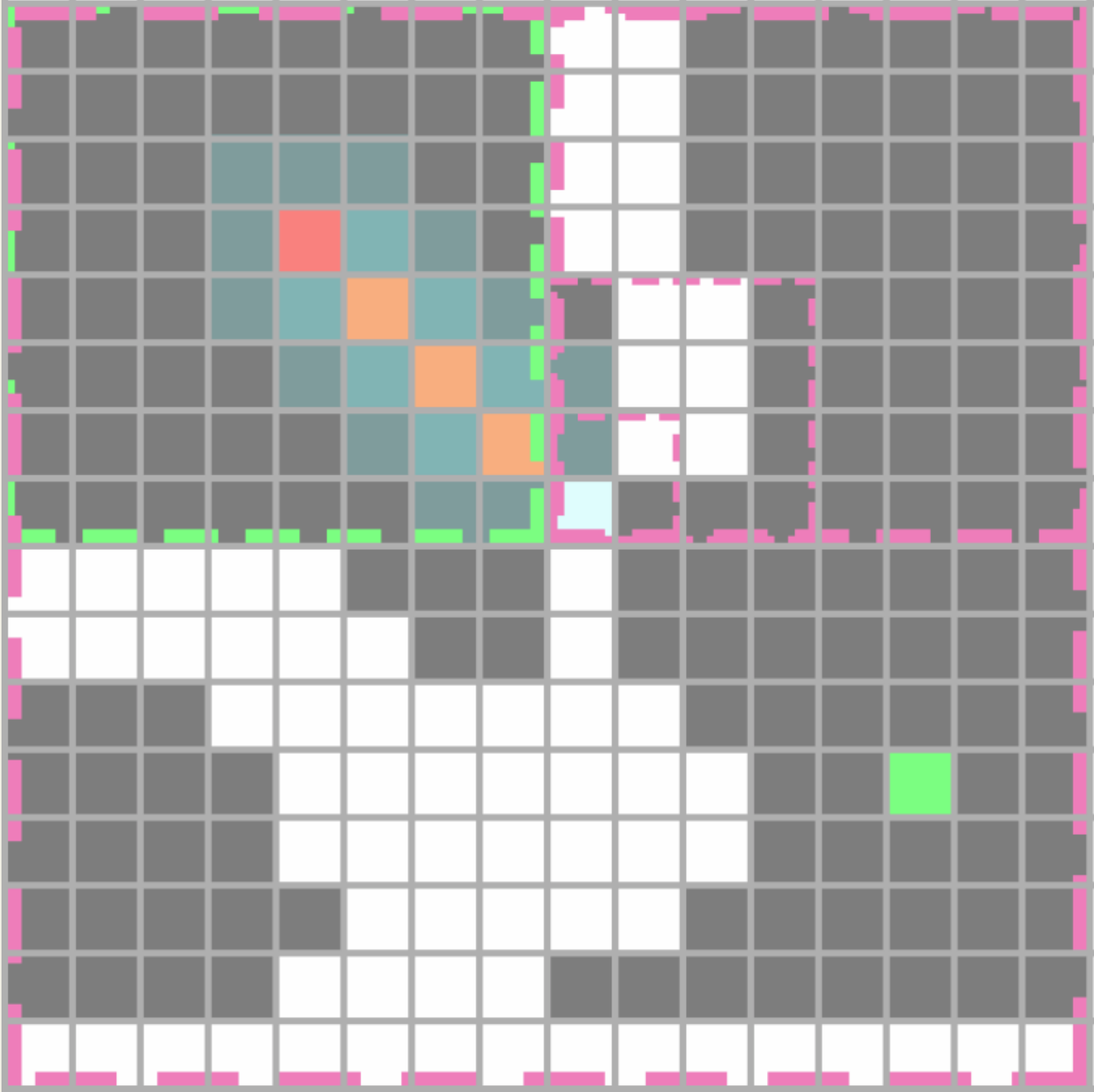


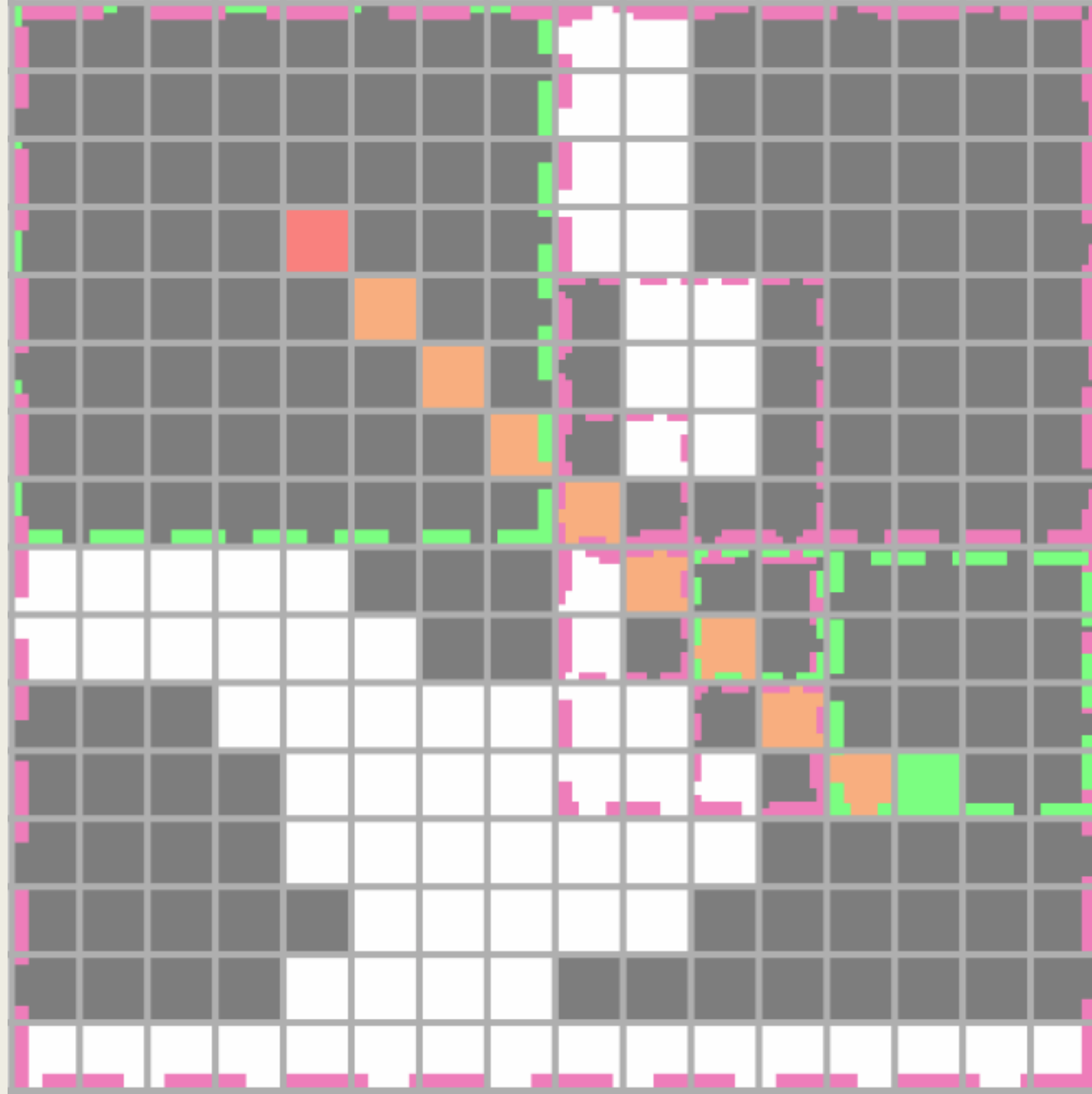






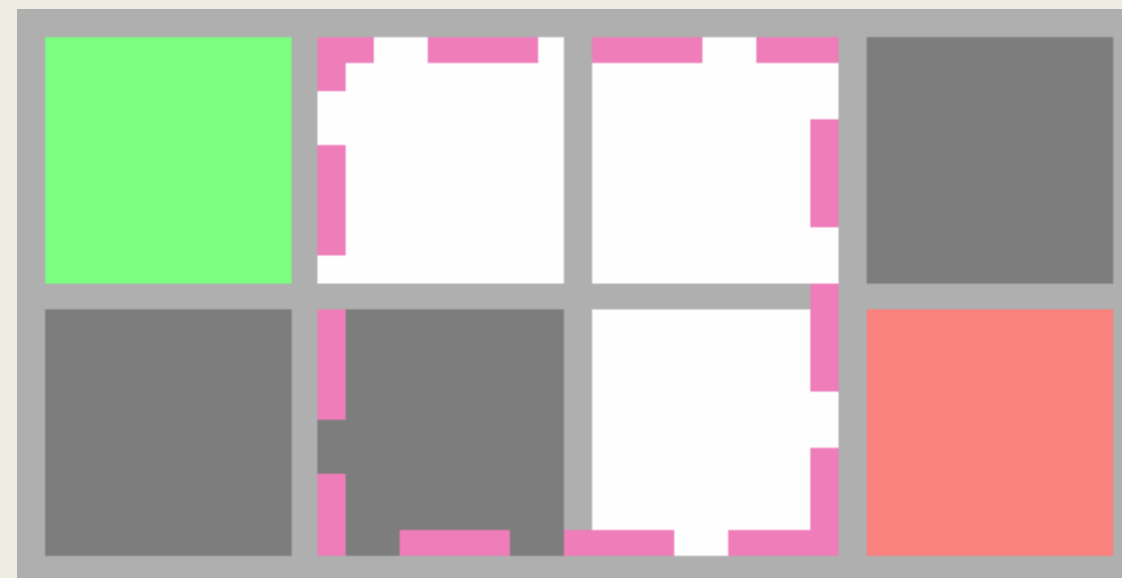
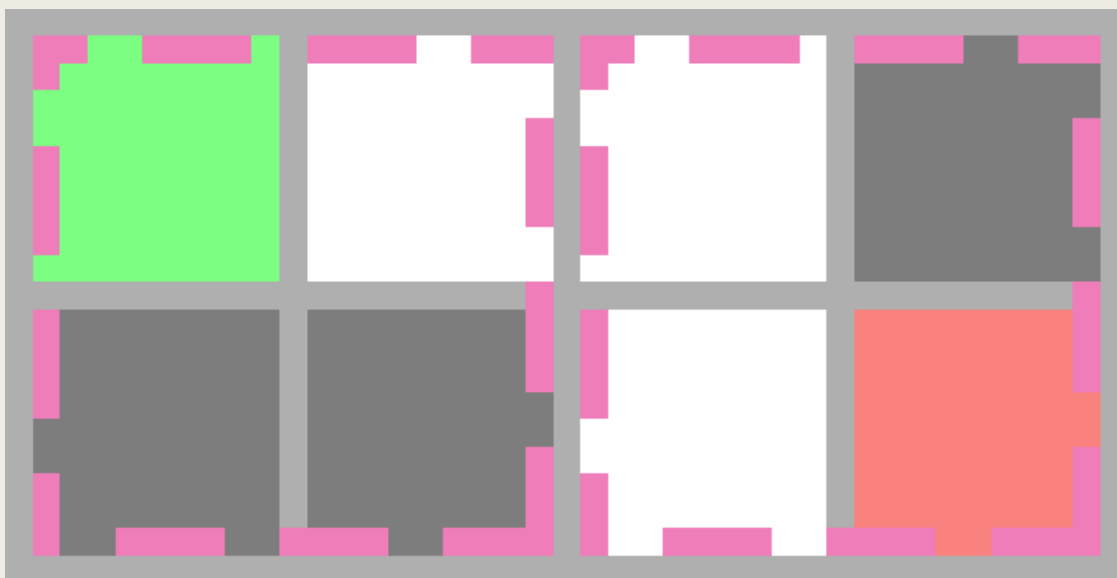
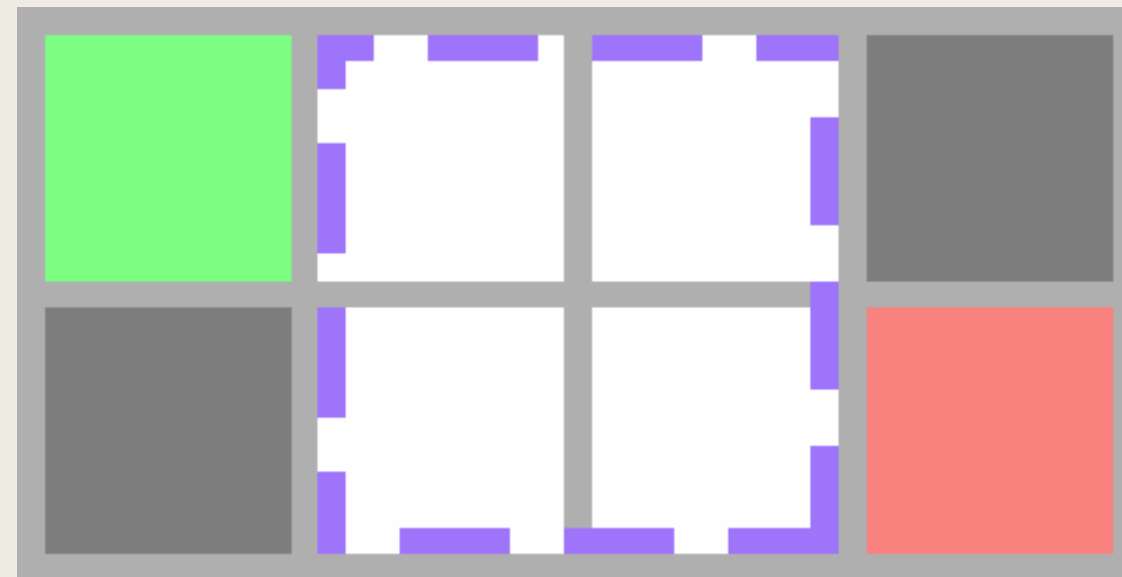
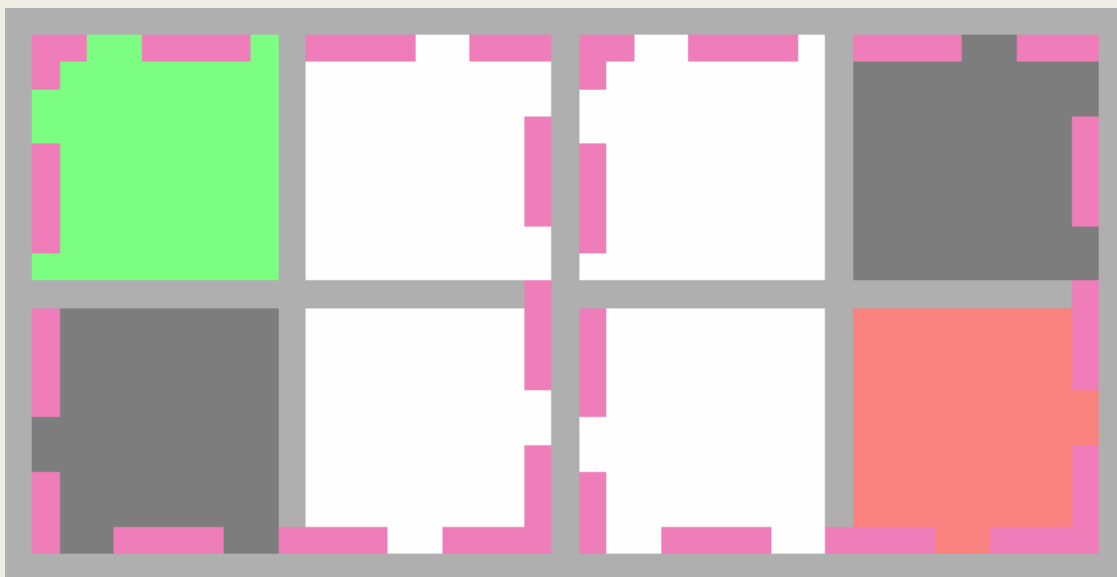






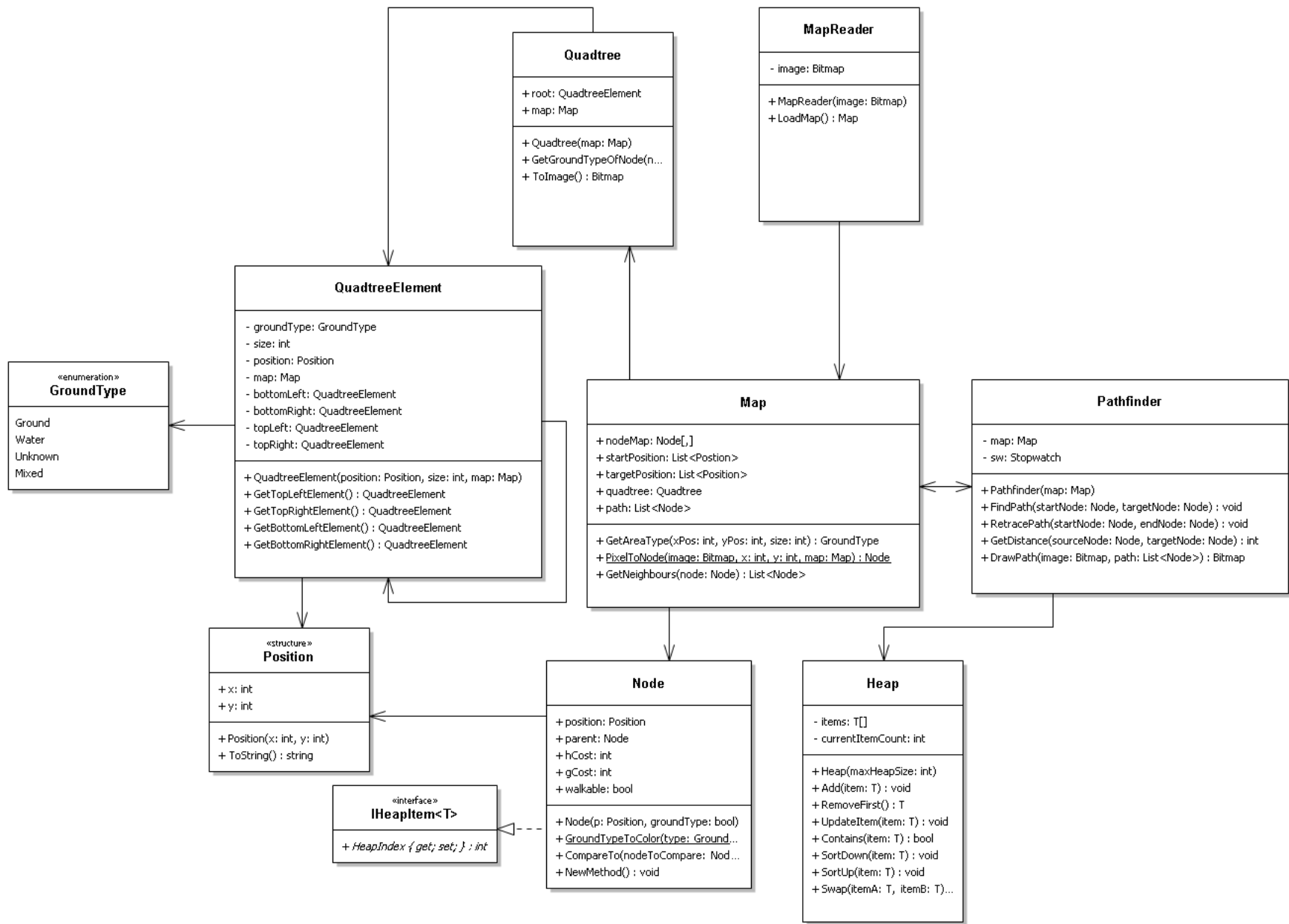
BEACHTUNG DES SONDERFALLS

Lösung: Bennet



IMPLEMENTIERUNG

Lösung: Bennet



LÖSUNG: JONAS

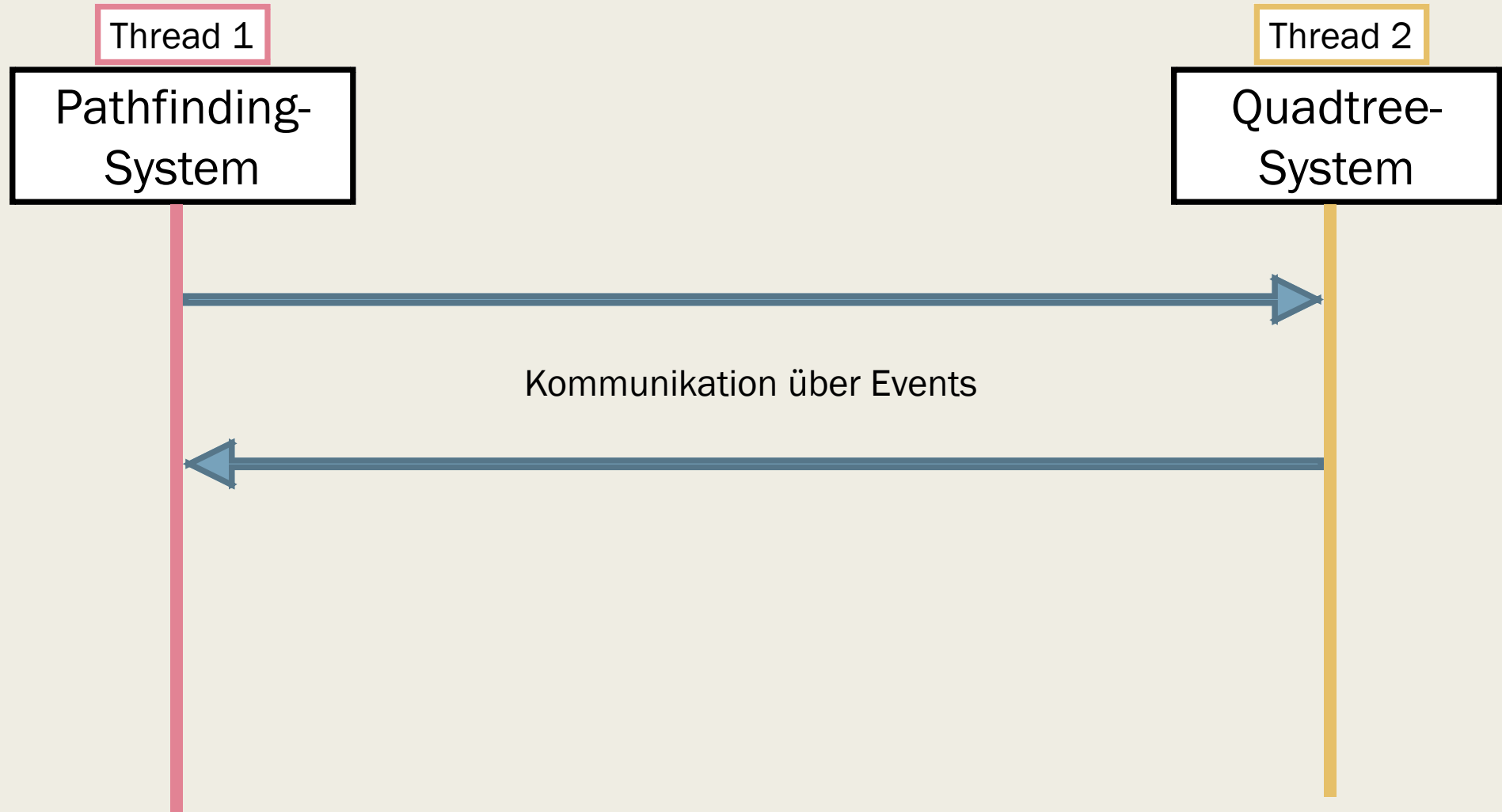


Grundlegendes

- Algorithmus-Bestandteile:
 - *A* - Pathfinding*
 - *Quadtree*
- Verwendung einer Game-Engine für effizienteres Arbeiten mit Texturen/Bildern
- Verwendung von Multithreading zur Performance-Steigerung

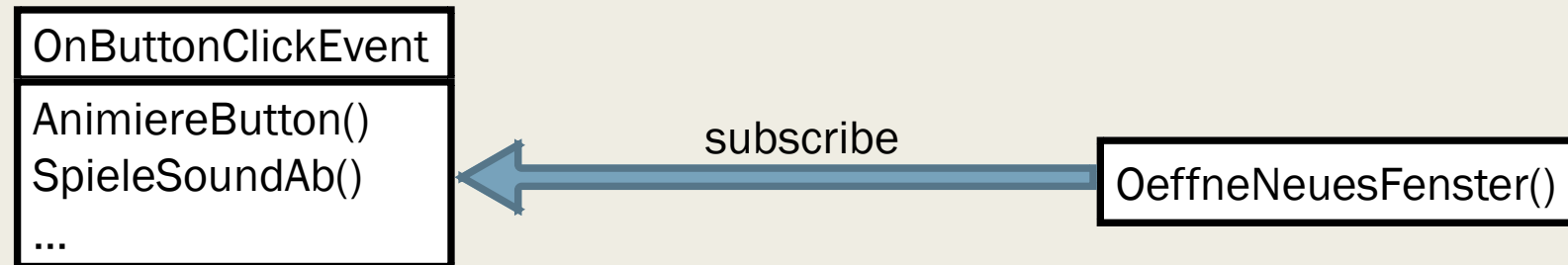


Grundlegende Programmstruktur



Events (Vereinfacht)

- Ansammlung von Methoden
- Man kann eine Methode hinzufügen („subscribe“) oder entfernen („unsubscribe“)
- Beim Aufrufen („invoke“) des Events werden alle aktuell hinzugefügten Methoden ausgeführt

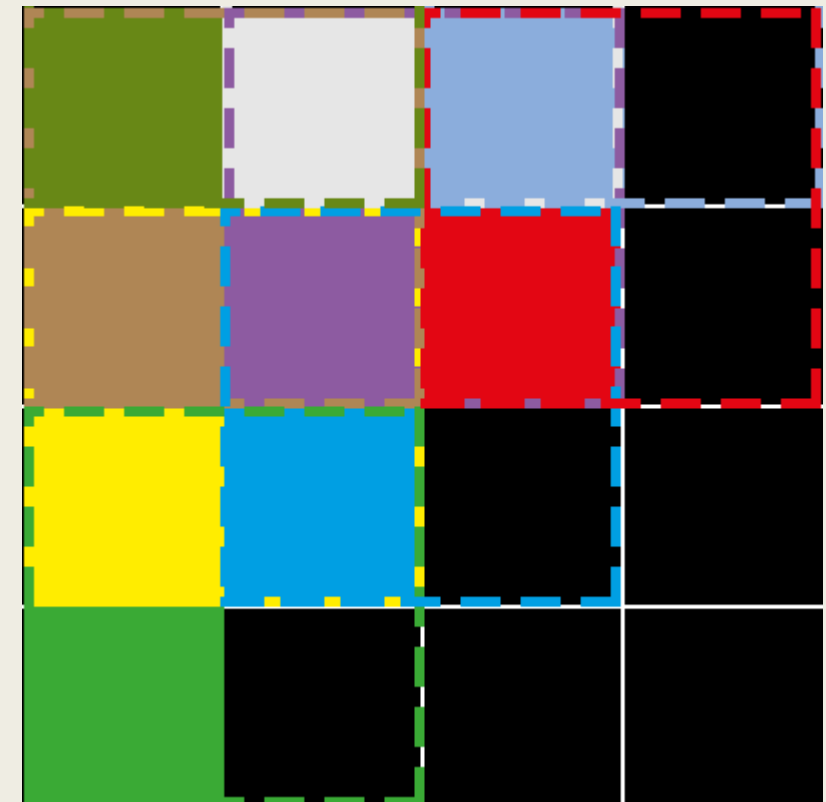


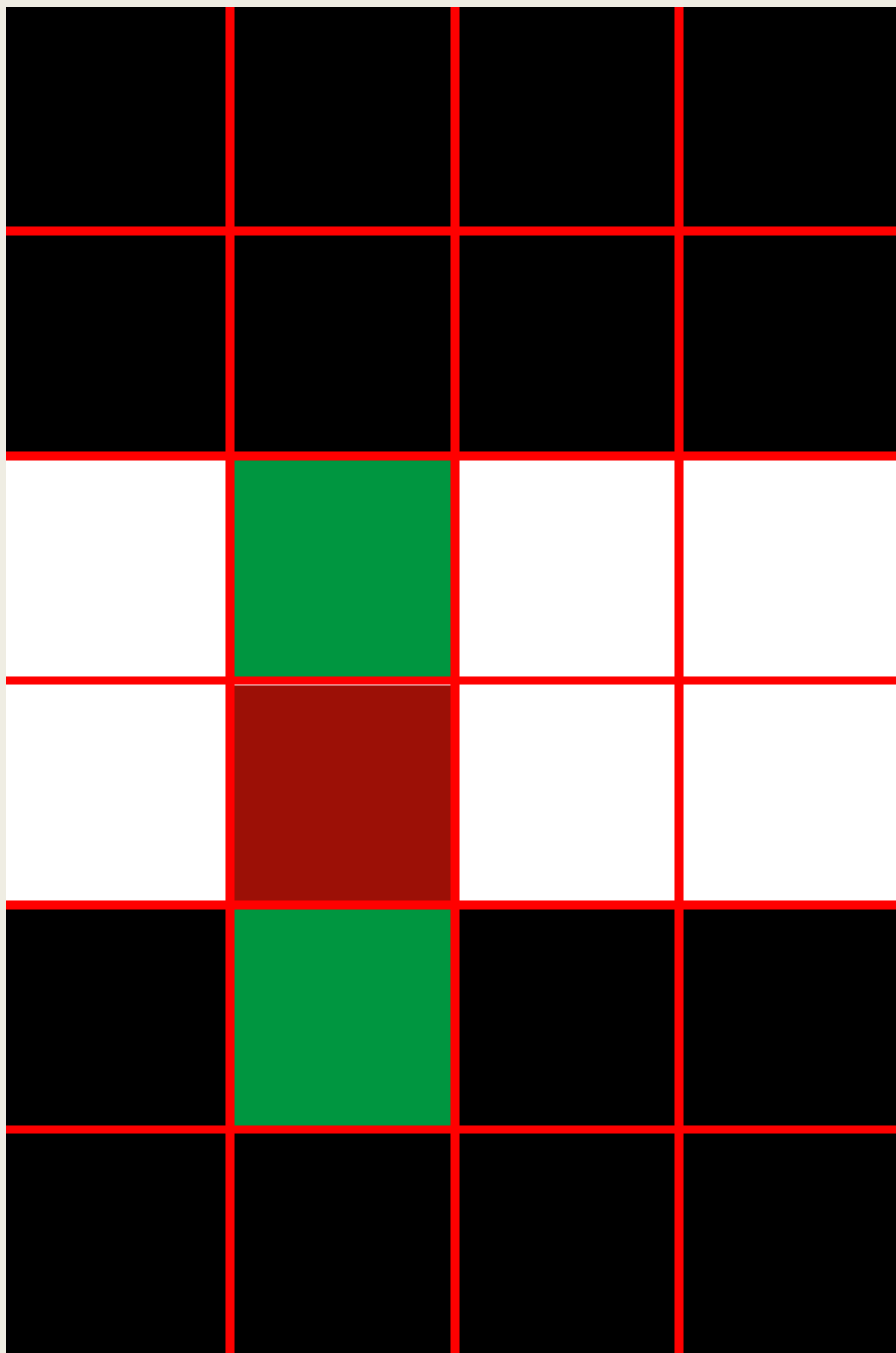
Beachtung des Sonderfalls

- A*-Grid besteht aus 10x10 Nodes → Problem bei Sonderfall
- Niemals einem einzelnen Pixel einem MapTyp zuordnen!
- LÖSUNG:

Definition des Status eines Pixels **über anliegende Pixel!**

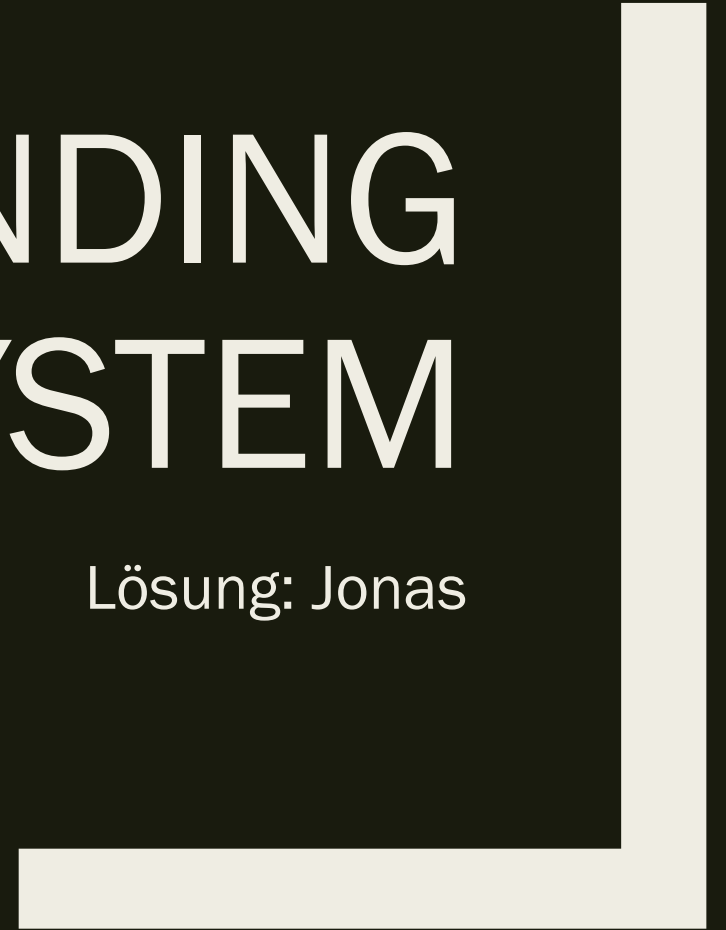
→ Definition jeder A*-Node über das zu ihr gehörige 20x20 Map-Quadrat

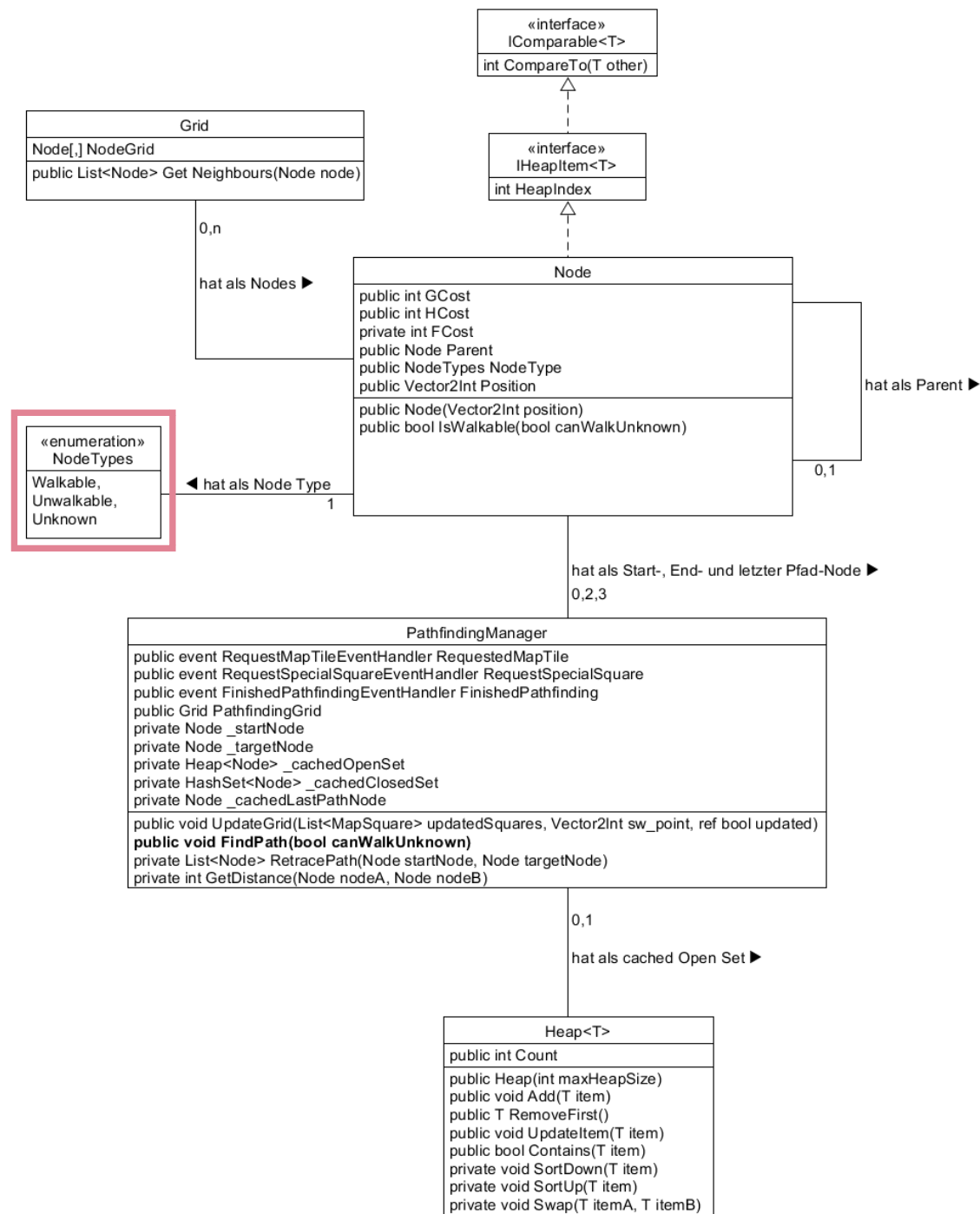




DAS PATHFINDING SYSTEM

Lösung: Jonas







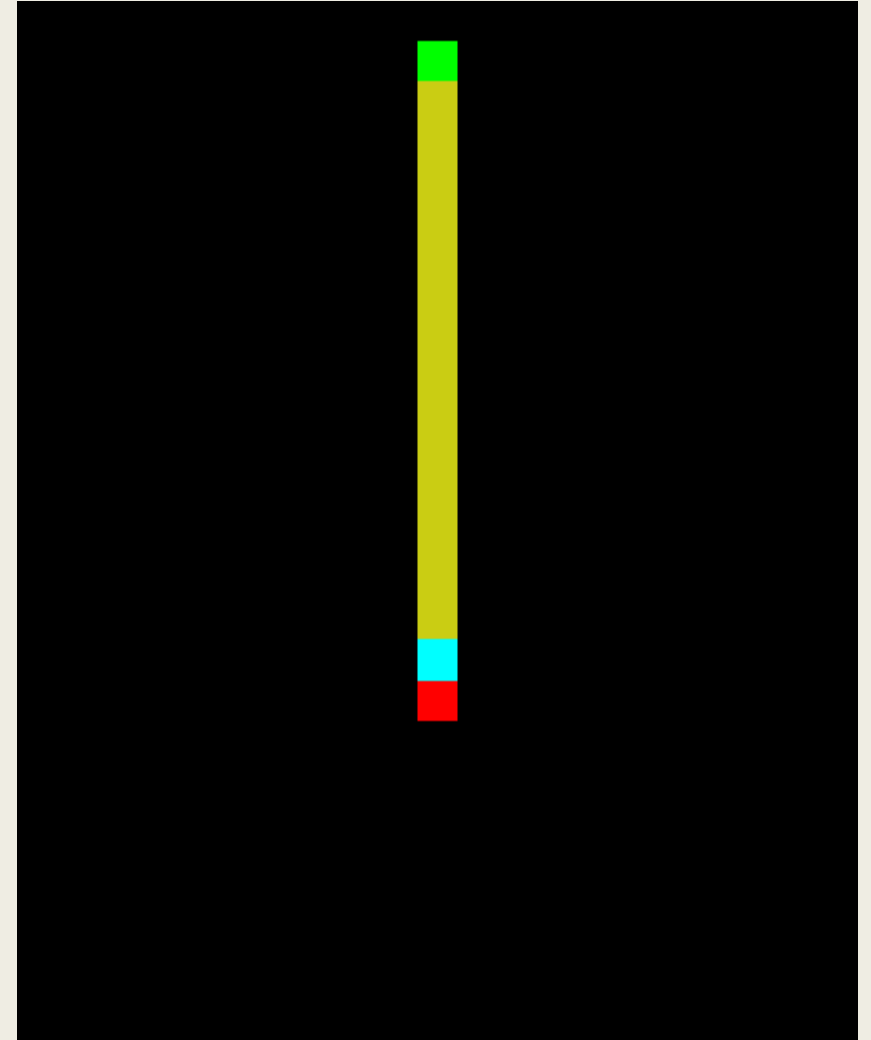
Pathfinding 1:

(Nur „walkable“-Nodes sind auch wirklich begehbar)



Pathfinding 2:

(„walkable“- UND „unknown“-Nodes sind begehbar)



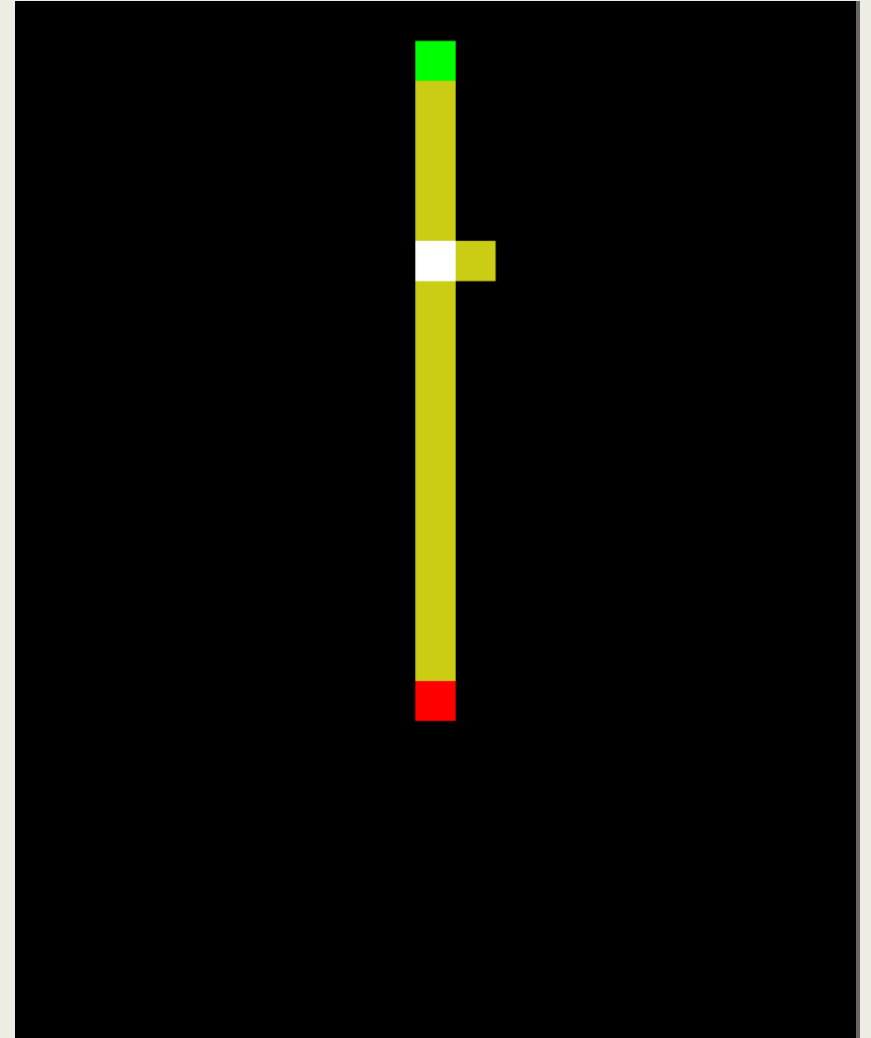
Pathfinding 1:

(Nur „walkable“-Nodes sind auch wirklich begehbar)



Pathfinding 2:

(„walkable“- UND „unknown“-Nodes sind begehbar)

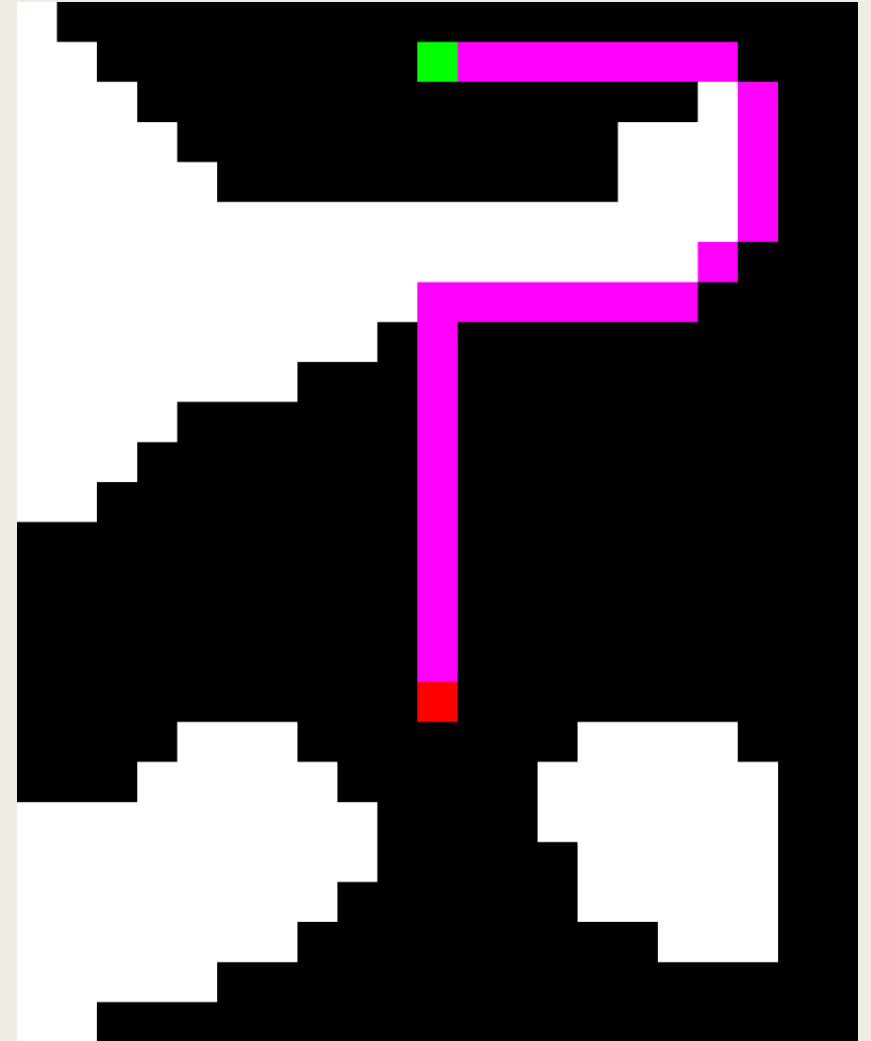


Pathfinding 1:
(Nur „walkable“-Nodes sind auch wirklich begehbar)

Pathfinding 1:
(Nur „walkable“-Nodes sind auch wirklich begehbar)



WEG GEFUNDEN!

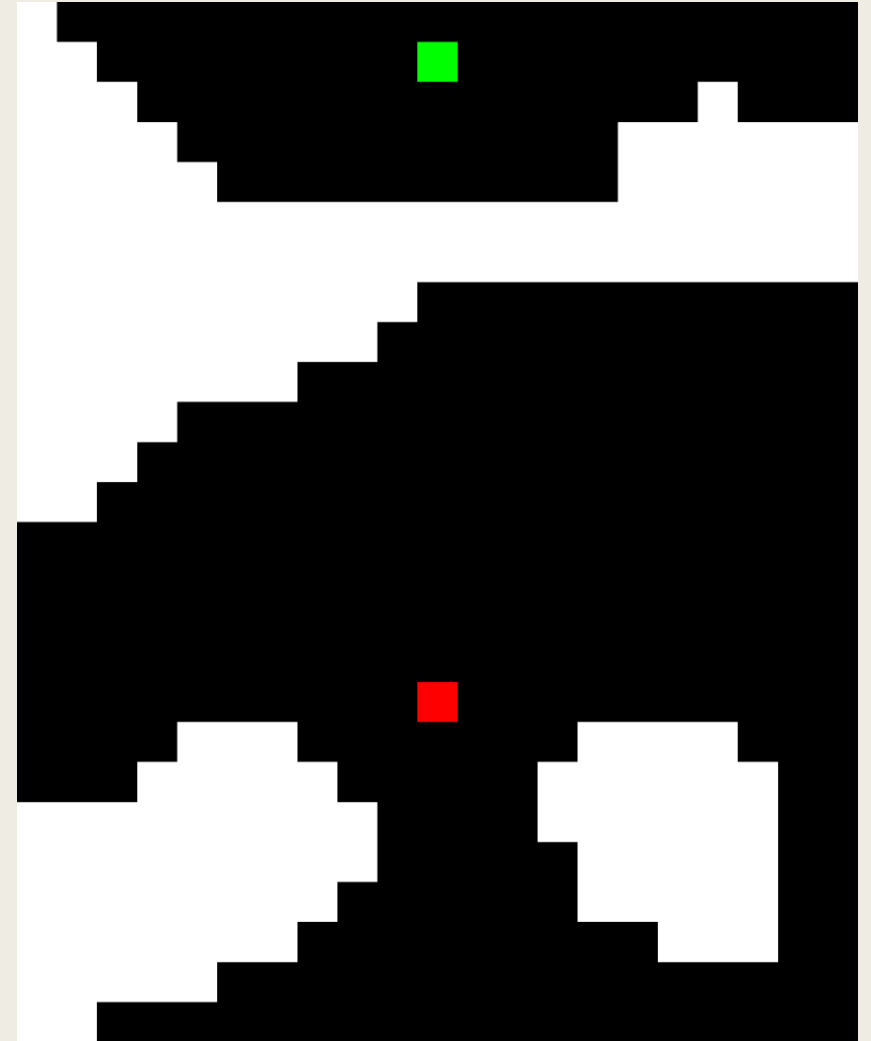


Pathfinding 2:

(Nur „walkable“-Nodes sind auch wirklich begehbar)

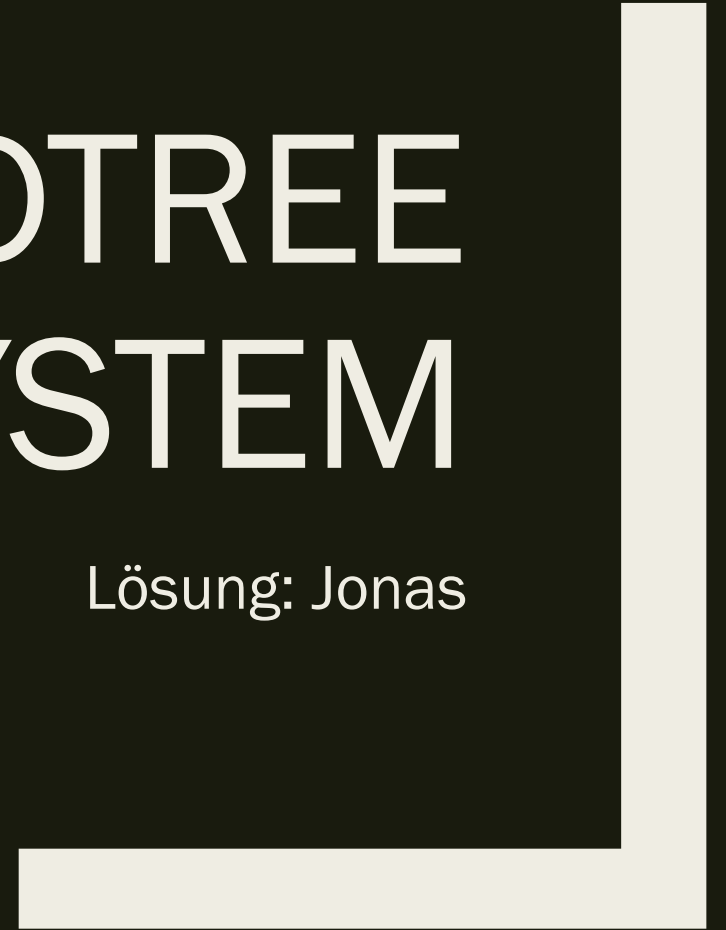


**ES GIBT KEINEN
WEG GEFUNDEN!**

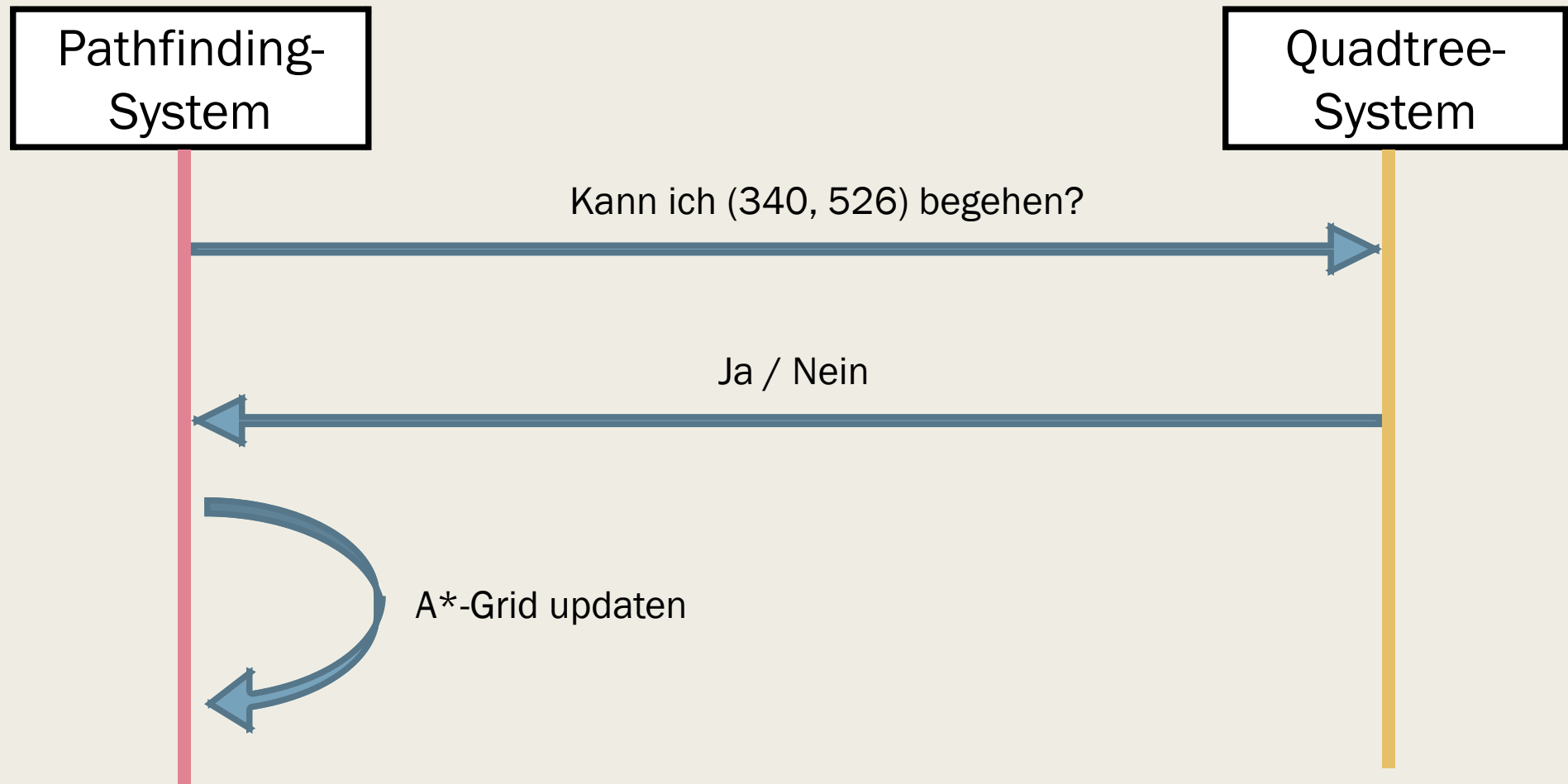


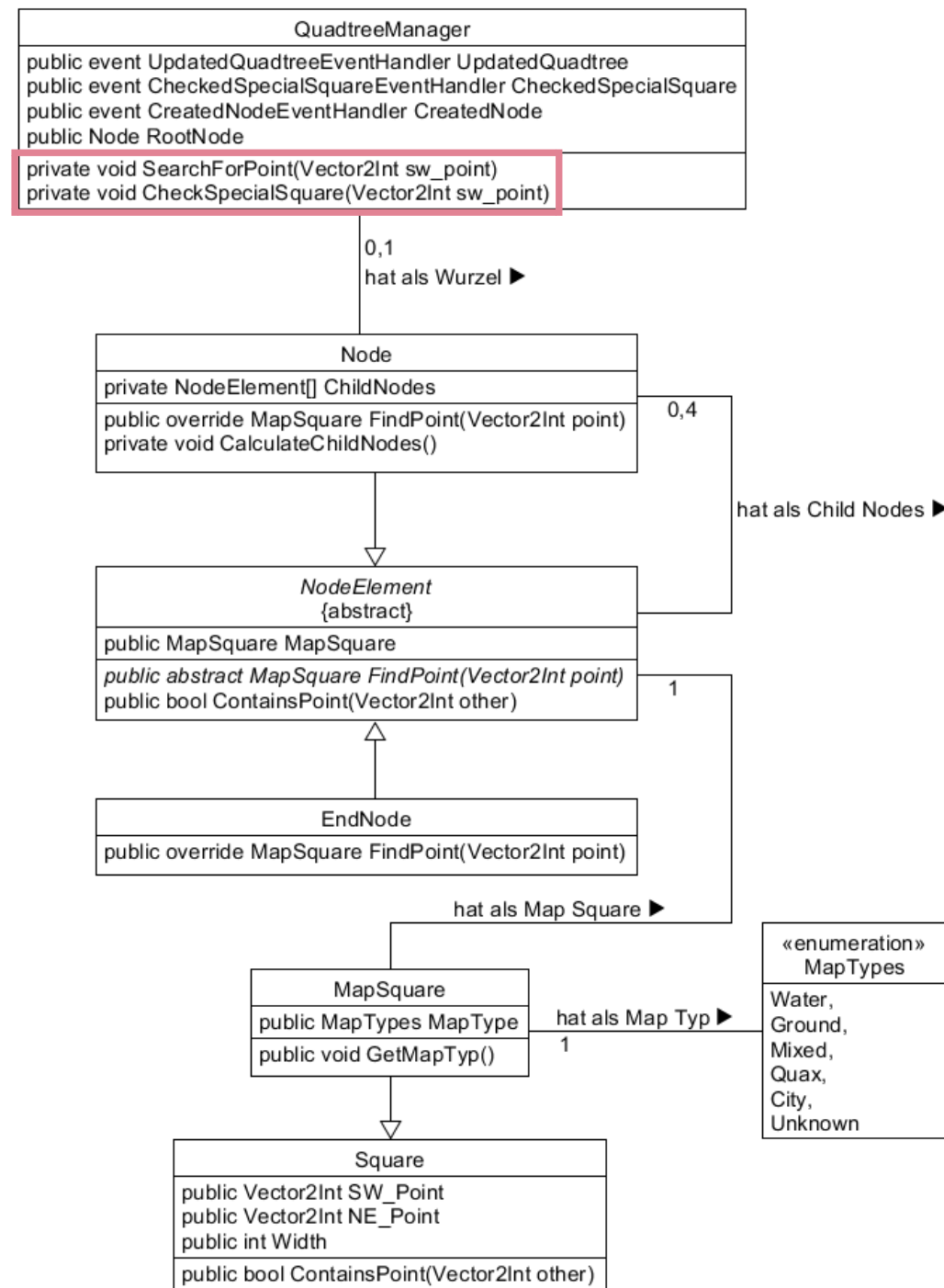
DAS QUADTREE SYSTEM

Lösung: Jonas



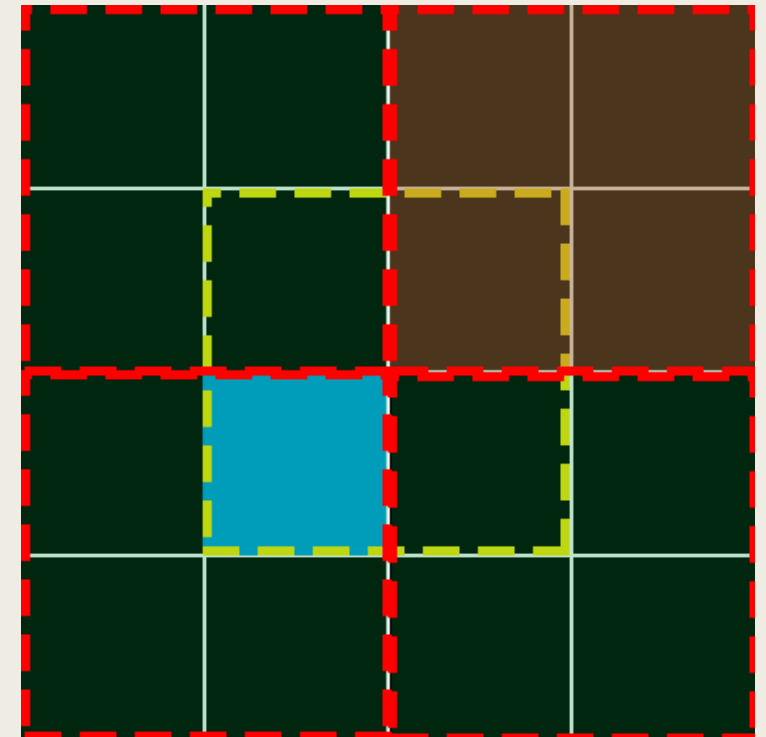
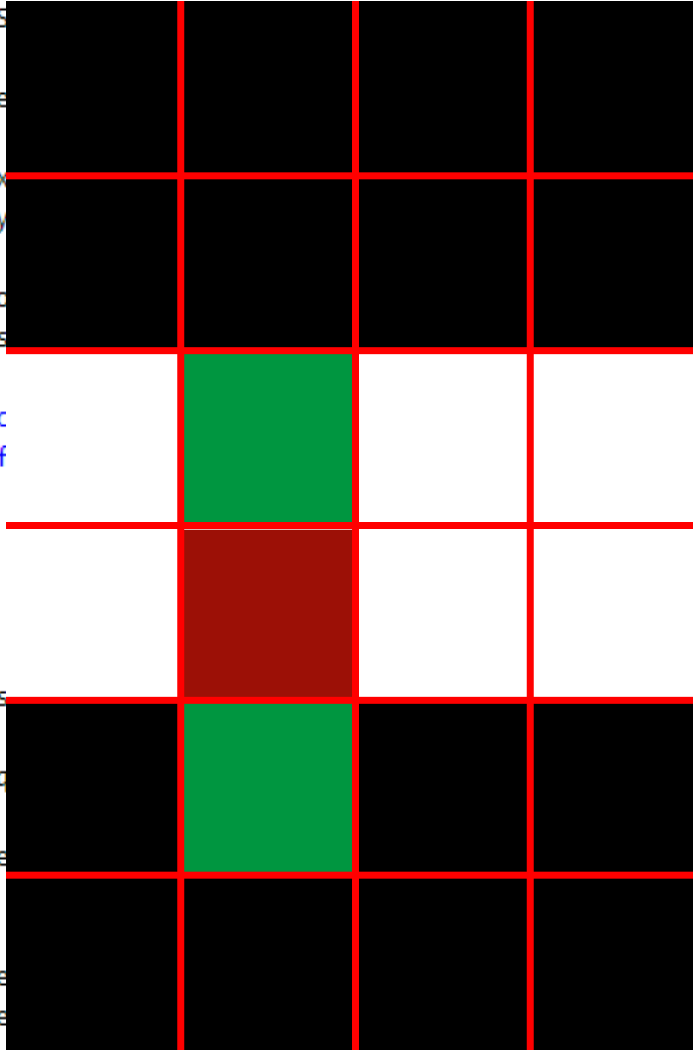
Aufgabe des Quadtree-Systems





Ist Node (x, y) begehbar?

```
1 private void S
2 {
3     var update
4
5     for (var x
6     for (var y
7     {
8         var po
9         var is
10
11     foreac
12         if
13         {
14
15         }
16     }
17
18     if (is
19
20     var so
21
22     update
23 }
24
25 if (Update
26     Update
27 }
```

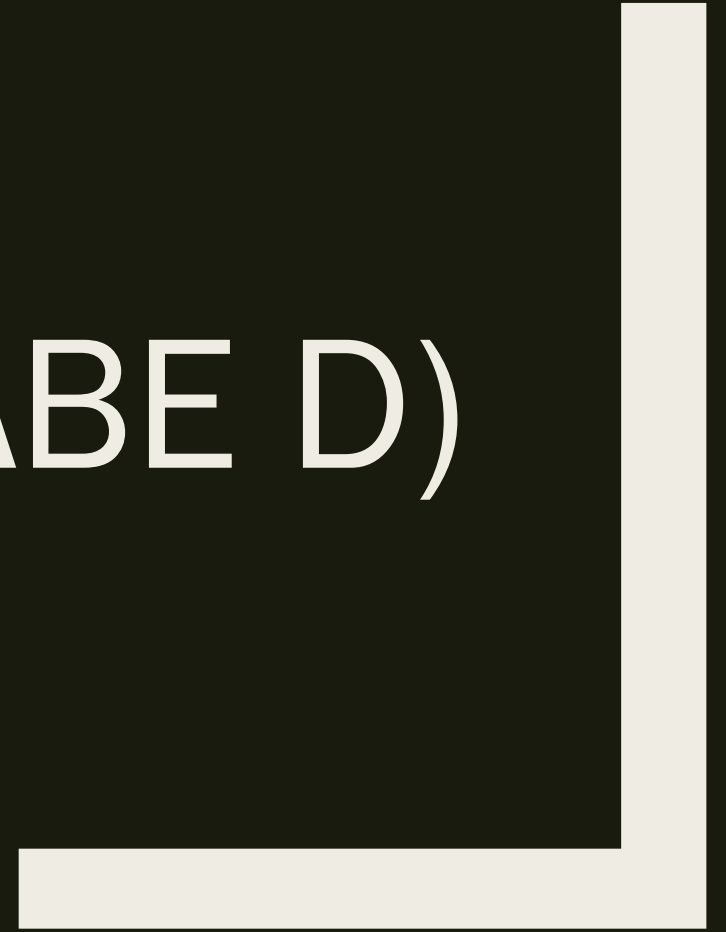


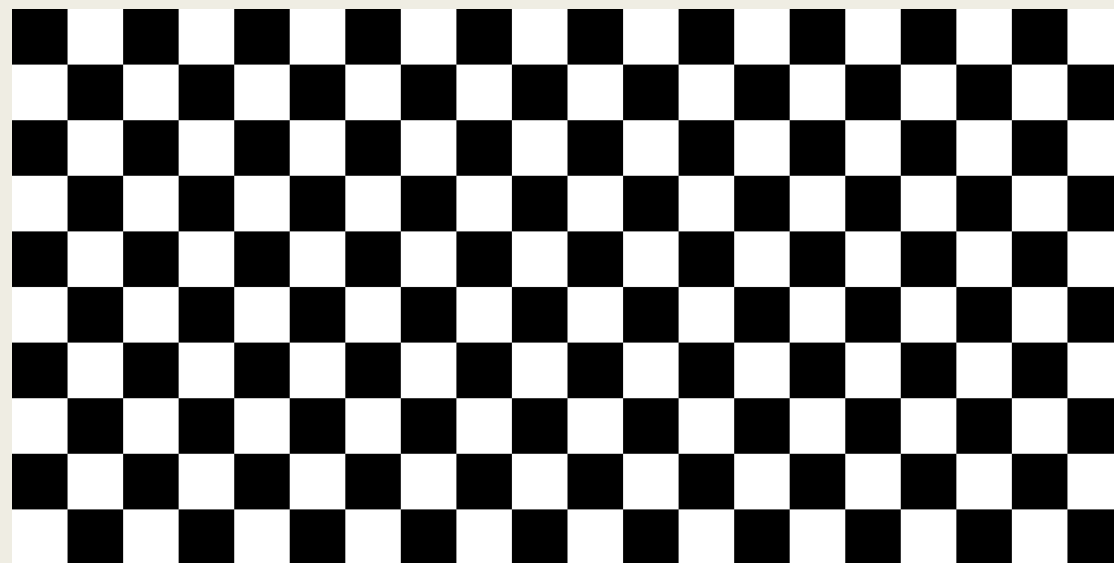
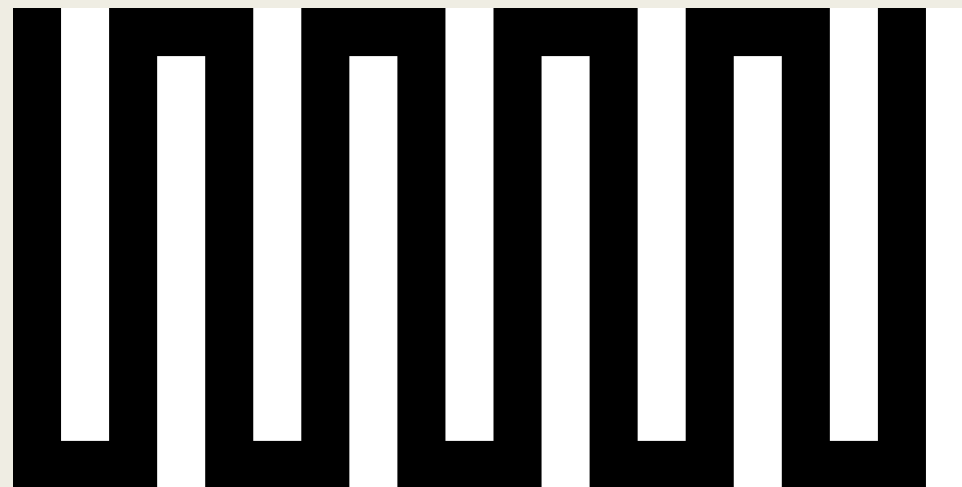
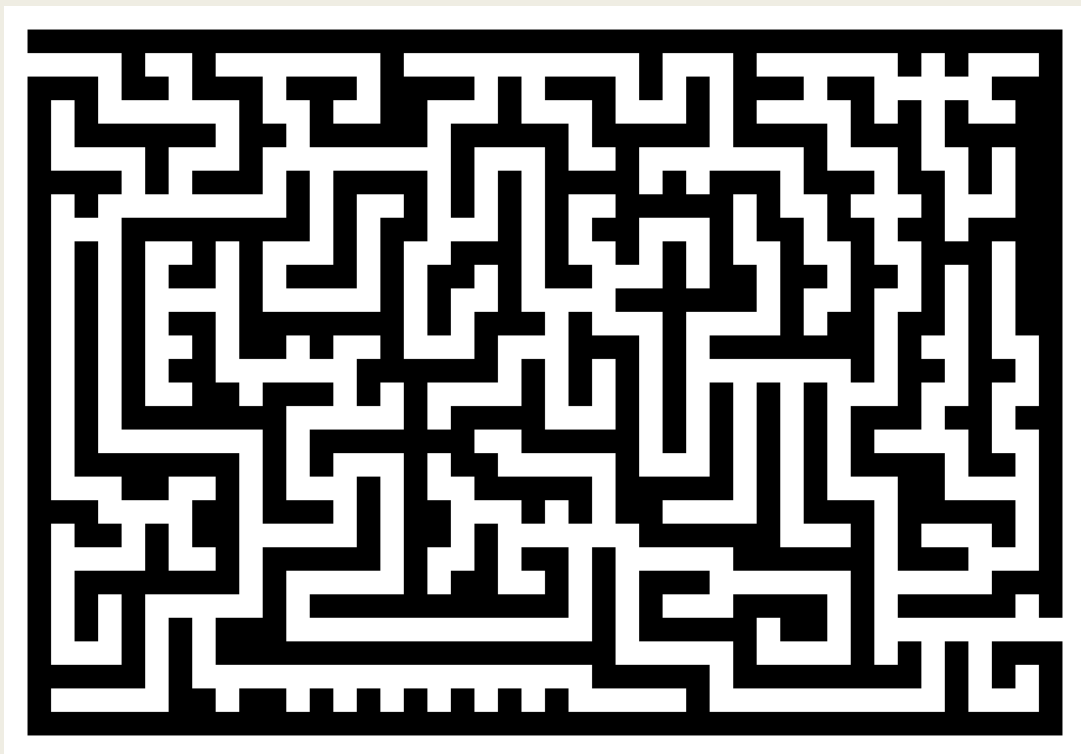
CheckSpecialSquare(...)

IMPLEMENTIERUNG

Lösung: Jonas

TEILAUFGABE D)







FAZIT

Jonas



FAZIT

Bennet

Quellen

- <https://people.eecs.berkeley.edu/~demmel/cs267/lecture26/Quadtree1.gif>
- <https://i.ytimg.com/vi/FgSEszMQt3g/maxresdefault.jpg>
- https://upload.wikimedia.org/wikipedia/commons/thumb/1/19/Unity_Technologies_logo.svg/1200px-Unity_Technologies_logo.svg.png
- https://upload.wikimedia.org/wikipedia/commons/thumb/8/8f/Flat_cross_icon.svg/500px-Flat_cross_icon.svg.png
- <https://www.kasa-solutions.com/wp-content/uploads/sites/2/2016/01/icon-check.png>
- <https://cdn4.iconfinder.com/data/icons/defaulticon/icons/png/256x256/help.png>
- <https://qiao.github.io/PathFinding.js/visual/>