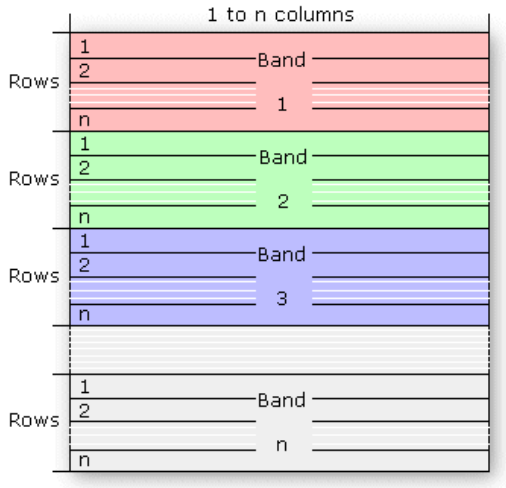
Rastermodell

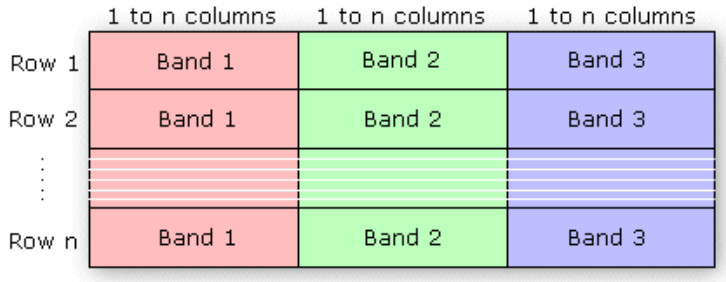
Angenommen wir wollen mehrere Bänder/Layer durchlaufen (z.B. bei hyperspektralen Daten). Welche Durchlaufvorschriften gibt es dann?

Nach dieser Kombination der Bänder/Layer kann wieder eines der vorherigen Durchlaufmuster verwendet werden.

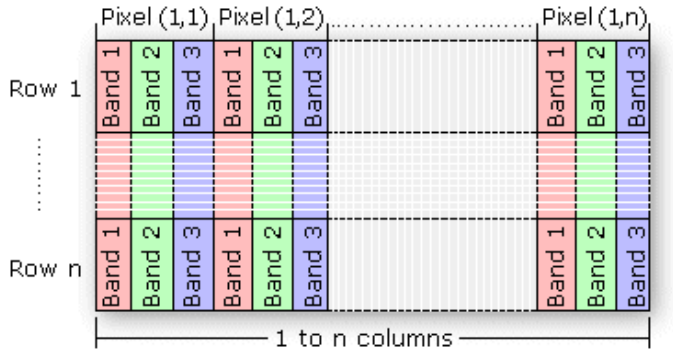
Band sequential (BSP)



Band Interleaved by Line (BIL)



Band Interleaved by Pixel (BIP)



https://desktop.arcgis.com/de/arcmap/10.3/manage-data/raster-and-images/bil-bip-and-bsq-raster-files.htm

Rastermodell

Große Datenmengen (z.B. hyperspektrale Daten) benötigen effektive speichereffiziente Methoden. Fortgeschrittene Beispiele sind z.B. JPEG (basierend auf wavelets) oder NASA fast lossless algortihm für hyperspektral Daten (www.techbriefs.com/component/content/article/tb/pub/techbriefs/information-sciences/49).

Hier wollen wir zwei einfache Methoden besprechen:

- **Lauflängenkodierung**

Statt Attributwerte für jedes Pixel zu speichern, werden die aufeinander Pixel mit gleichem Attributwert kombiniert und wie folgt gespeichert (abhängig von der Durchlaufvorschrift):

		1	2	3	4	5	6	7	8	9	10	Run-length encoding
Rows	1	A	A	A	A	B	B	B	A	A	A	(4,A),(3,B),(3,A)
	2	A	A	A	B	B	B	A	A	A	C	(3,A),(3,B),(3,A),(1,C)
	3	A	A	B	B	B	A	A	A	C	C	(2,A),(3,B),(3,A),(2,C)
	4	A	B	B	B	A	A	C	C	C	C	(1,A),(3,B),(2,A),(4,C)
	5	A	A	A	A	A	A	C	C	C	C	(6,A),(4,C)

Wie verhält sich der Speicheraufwand von Lauflänge zu Attributwert? Geht damit Informationsverlust einher? Kann man die Homogenitätsannahme aufweichen? Mit welchen Konsequenzen?

Rastermodell

- **Blockkodierung**

Statt einer zeilenweisen, könnte man auch auf flächige homogene Objekte abzielen. Zum Beispiel in dem man wie folgt codiert:

		1	2	3	4	5	6	7	8	9	10	Run-length encoding
Rows	1	A	A	A	A	B	B	B	A	A	A	(4,A),(3,B),(3,A)
	2	A	A	A	B	B	B	A	A	A	C	(3,A) ... 1,C)
	3	A	A	B	B	B	A	A	A	C	C	(2,A),(3 ...),(2,C)
	4	A	B	B	B	A	A	C	C	C	C	(1,A) ... 4,C)
	5	A	A	A	A	A	A	C	C	C	C	(6,A),(4,C)

(1,1,2,A), (1,3,1,A), (1,4,1,A), (1,5,2,B), (1,7,1,B), (1,8,2,A), (1,10,1,A), (2,3,1,A), (2,4,2,B), …

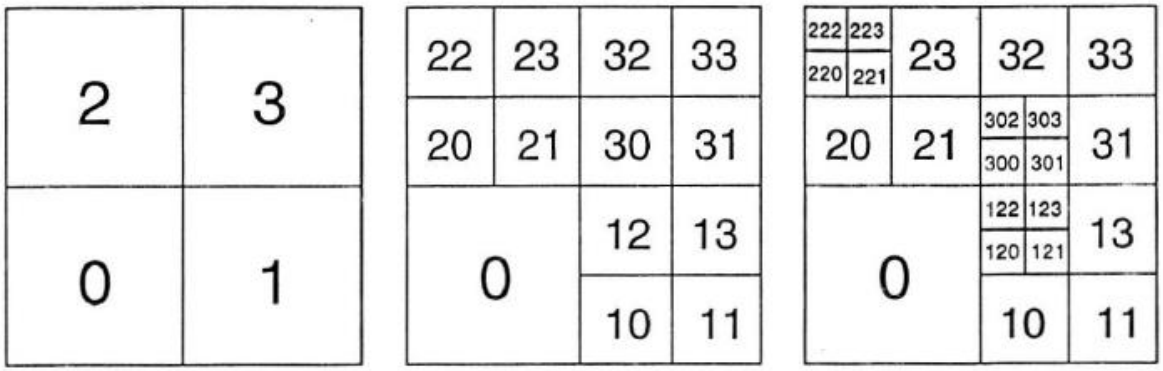
Die ersten zwei Einträge beschreiben die Zeile und Spalte der linken oberen Ecke eines Quadrates, der dritte Eintrag die Seitenlänge des Quadrates, der vierte Eintrag das Attribut.

Effizient, wenn viele großflächige homogene Gebiete vorliegen. Bei sehr komplexen Bildstruktur ist die Kompressionsrate gering.

Rastermodell

- **Quadtrees (Octrees in 3-D)**

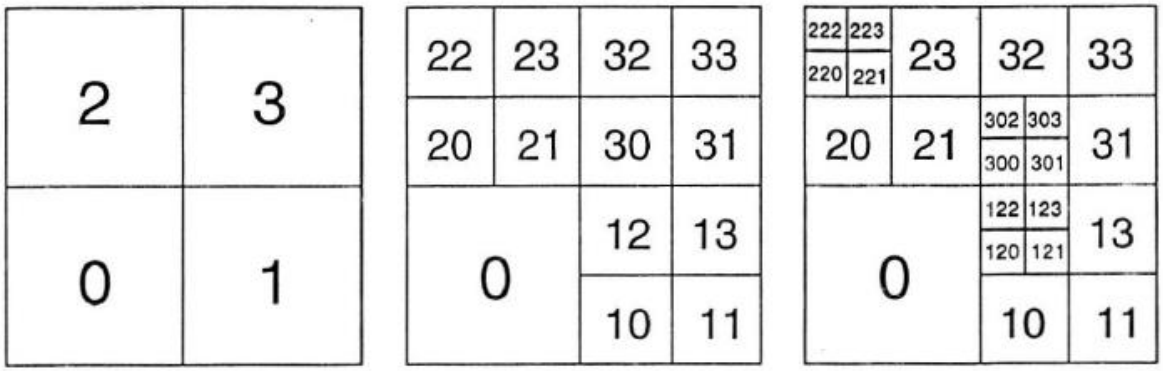
Ein weiteres auf flächige homogene Objekte abzielendes System. Sukzessive, hierarchische Unterteilung des Rasters in kleiner werdende Quadranten.



Rastermodell

- **Quadtrees (Octrees in 3-D)**

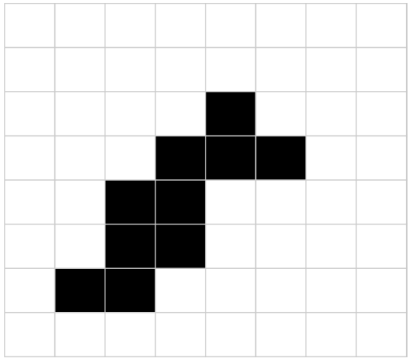
Ein weiteres auf flächige homogene Objekte abzielendes System. Sukzessive, hierarchische Unterteilung des Rasters in kleiner werdende Quadranten.

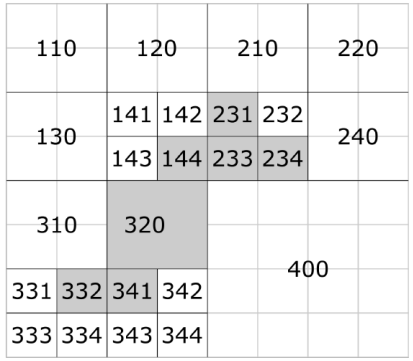


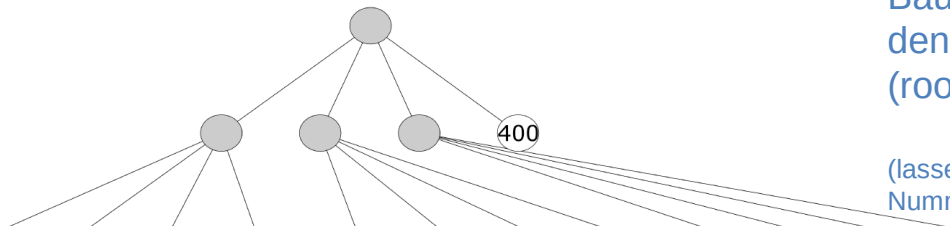
Es bietet sich an die Quadranten mit einem Zahlsystem zur Basis vier zu versehen, für eine intuitive Nummerierung der Quadranten.

Rastermodell

- **Quadtrees (Octrees in 3-D)**







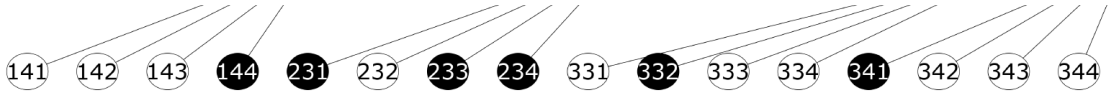
Die Zerlegung des linken Rasters mittels Quadtrees sähe wie folgt aus:

(400,W), (110,W), (120,W), (130,W), (320,B),…, (344,W)

Die weißen Flächen müssten bei einem binären Bild theoretisch nicht mit gespeichert werden. Die Endknoten des hierarchischen Baums nennt man auch Blattknoten, den obersten Knoten Wurzelknoten (root node).

(lassen Sie sich nicht durch die geänderte Nummerierung der Quadranten verwirren)

https://www.maptiler.com/google-maps-coordinates-tile-bounds-projection/

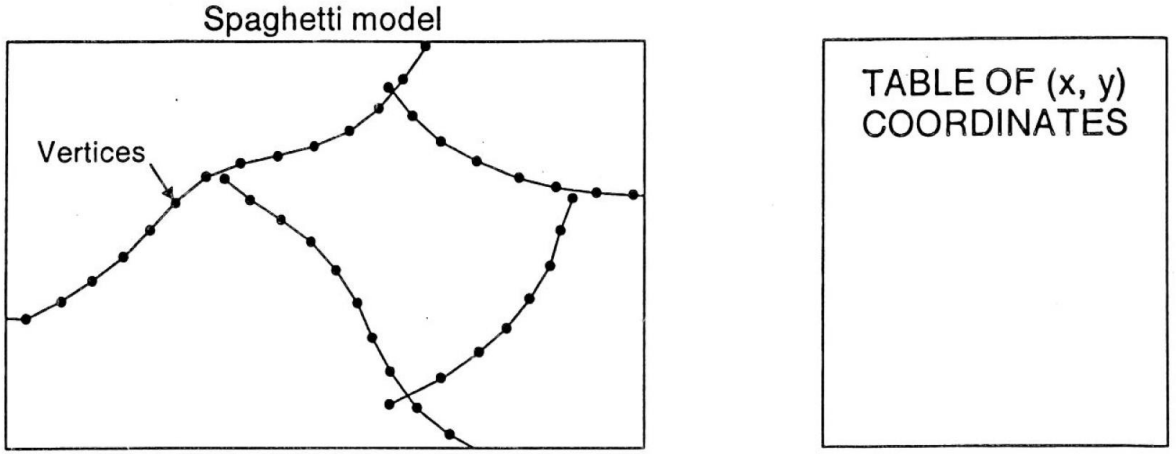


Wie verhält sich Speicheraufwand von Quadtrees zur vorher beschriebenen Blockkodierung?

Vektormodell

Grundlegende Strukturen sind Vertexe (0-D). Darauf aufbauend gibt es Linienzüge/Polygonzüge (1-D), Polygone (2-D), etc.

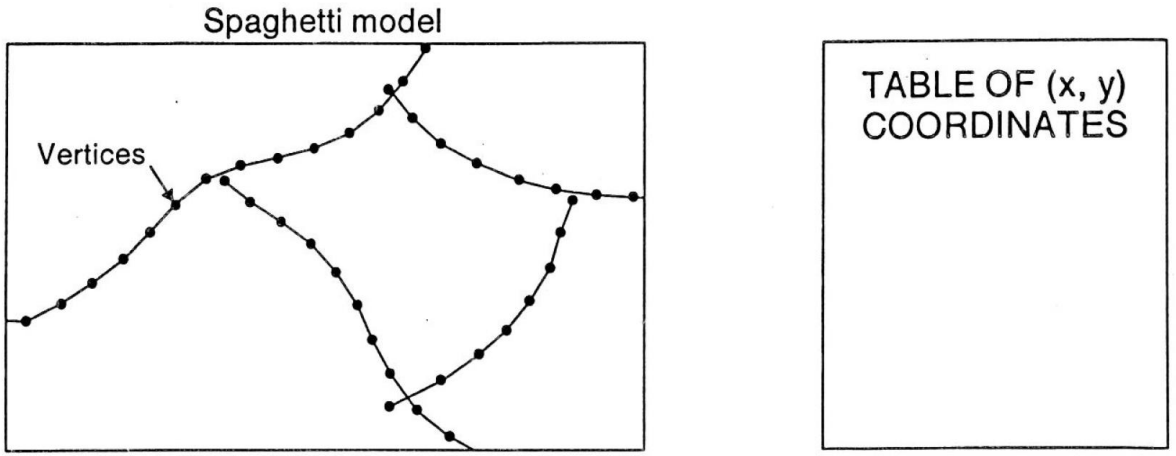
Die einfachste Datenstruktur ist die **Sphagettistruktur**:



Vektormodell

Grundlegende Strukturen sind Vertexe (0-D). Darauf aufbauend gibt es Linienzüge/Polygonzüge (1-D), Polygone (2-D), etc.

Die einfachste Datenstruktur ist die **Sphagettistruktur**:



Einzelne Objekte werden als einfache Listen von Koodinaten der beinhalteten Vertices hinterlegt (Reihenfolge spielt eine Rolle). Bei Polygonen muss der letzte Vertex mit dem ersten übereinstimmen.

Vektormodell

A. Point table. X and Y are locational coordinates, A_1, A_2, ...A_n are thematic attributes. Each record or row is a single point object, such as a mineral deposit location, or geochemical sample site.

ID #	X	Y	A_1	A_2	..	A_n
1	x_1	y_1	a_{11}	a_{12}	.	a_{1n}
2	x_2	y_2	a_{21}	a_{22}	.	a_{2n}
3	x_3	y_3	a_{31}	a_{32}	.	a_{3n}
.	.	.	.	.	.	.
...	.	.	.	.	.	.
m	x_m	y_m	a_{m1}	a_{m2}	.	a_{mn}

B. Line table[1]. Many lines are held in the same table or file. Each new line begins with a header (one or more records), followed by the locational coordinates of the vertices or points defining the line. In this case the first field of the header record is the line ID#, the second field is the number of vertices, and the third and fourth (or more) fields are attributes, such as feature codes. There are *m* lines.

1	5	2	7	**Header for line 1**
	x_1	y_1		
	x_2	y_2		Coordinates of vertices for line 1
	x_3	y_3		
	x_4	y_4		
	x_5	y_5		
2	2	4	7	**Header for line 2**
	x_1	y_1		Coordinates for line 2
	x_2	y_2		
3	15	2	8	**Header for line 3**
.	.	.		.
m	etc	etc		etc

[1] The table is nonstandard, because it contains more than one kind of record.

C. Polygon table[1]. This is essentially the same as for lines, except that the last vertex has the same coordinates as the first vertex in each polygon. Therefore there must be a minimum of four vertices per polygon. Each polygon may have many attributes, in which case the attribute data are held in a separate table, linked by polygon number. One attribute must define priority for plotting, to take care of the presence of islands. There are *m* polygons.

1	5	429	18	**Header for poly 1**
	x_1	y_1		
	x_2	y_2		Coordinates of vertices for polygon 1
	x_3	y_3		
	x_4	y_4		
	x_5	y_5		
2	4	39	12	**Header for poly 2**
	x_1	y_1		
	x_2	y_2		Coordinates for polygon 2
	x_3	y_3		
	x_4	y_4		
3	81	9	3	**Header for polygon 3**
.	.	.		.
m	etc	etc		etc

[1] This table is also nonstandard, because it contains more than one kind of record.

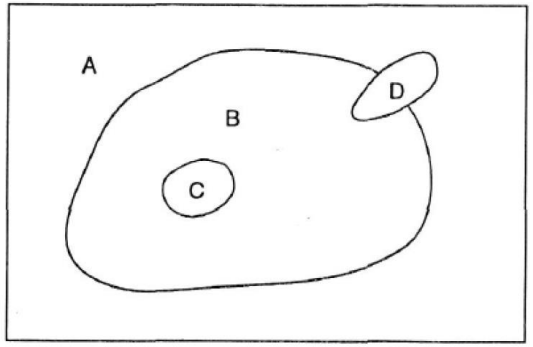
Einzelne Objekte werden als bloss ... (Reihenfolge spielt eine Rolle). ... übereinstimmen.

Es können zusätzlich Attribute hinterlegt werden. Die Attribute können in separeter Liste hinterlegt werden.

Vektormodell

Objekte in der Sphagettistruktur sind unabhängig voneinander.

- Dadurch entstehen Redundanzen (Vertexe, Linien, etc. werden eventuell mehrfach gespeichert).
- Wichtige Vorteile des Vektormodells z.B. bei Abfragen kommen nicht zum Tragen (es müssen schlussendlich wieder geometrische Information verarbeitet werden)
- Polygonen muss Priorität zugewiesen werden, um z.B. bei Plotten zu verhindern, dass grosse Polygone kleinere Polygone (etwa Inseln/Löcher) überdecken



Rock	Polygon	Priority
shale	A	0
sandstone	B	1
granite	C	2
granite	D	2

Vorteil: sehr effiziente Darstellung auf grafischen Systemen, da alle Daten direkt dargestellt werden können. Es ist kein zusätzlicher Suchaufwand in mehreren Tabellen notwedig.

Vektormodell

Topologische Datenstrukturen: Darstellung von “Nachbarschafts”beziehungen zwischen Objekten (geometrische Eigenschaften und Distanzen spielen keine grosse Rolle, sondern nur die relative Lage und Beziehung zueinander)

- *Grund-Elementtypen*: Punkte/Vertex, Kante (verbindet zwei Punkte), Fläche (begrenzt durch Kanten), Volumen (begrenzt durch Flächen). Alle Objekte von Dimension d>0 sind begrenzt durch Elemente von Dimension d-1

Vektormodell

Topologische Datenstrukturen: Darstellung von “Nachbarschafts”beziehungen zwischen Objekten (geometrische Eigenschaften und Distanzen spielen keine grosse Rolle, sondern nur die relative Lage und Beziehung zueinander)

- *Grund-Elementtypen*: Punkte/Vertex, Kante (verbindet zwei Punkte), Fläche (begrenzt durch Kanten), Volumen (begrenzt durch Flächen). Alle Objekte von Dimension d>0 sind begrenzt durch Elemnte von Dimension d-1
- Grund-Elementtypen erlauben Defintion weiterer *topologischer Elemente*:
  - **Linie/Linienzug/Bogen (line/arc)**: geordnete, zusammenhängende Menge von Kanten mit einem End- und einem Anfangsknoten
  - **Knoten (Node)**: Punkt an dem eine Linie beginnt/endet oder an dem sich zwei oder mehrere Linien treffen
  - **Kette/Bogen (chain/arc)**: Linien, die Teil der Genze eines Flächenobjektes sind
  - **Ring**: Grenze eines Flächenenlements, welche aus mehreren Ketten/Bögen besteht
  - **Polygon**: Flächenelement begrenzt durch ein oder mehrere Ringe (mindestens ein äußerer Ring)
    - Einfaches Polygon: nur ein äußerer Ring
    - Komplexes Polygon: ein oder mehrere innere Ringe, zusätzlich zu einem äußeren Ring

Vektormodell

Topologische Datenstrukturen: Darstellung von “Nachbarschafts”beziehungen zwischen Objekten (geometrische Eigenschaften und Distanzen spielen keine grosse Rolle, sondern nur die relative Lage und Beziehung zueinander)

- *Grund-Elementtype*
  Kanten), Volumen
  durch Elemnte von
- Grund-Elementtype
  - **Linie/Linienz**
    Kanten mit ei
  - **Knoten (Noc**
    mehrere Linie
  - **Kette/Bogen**
  - **Ring**: Grenze
  - **Polygon**: Flä
    äußerer Ring
    - Einfach
    - Komple
      äußere