# Koordinatenumwandlung

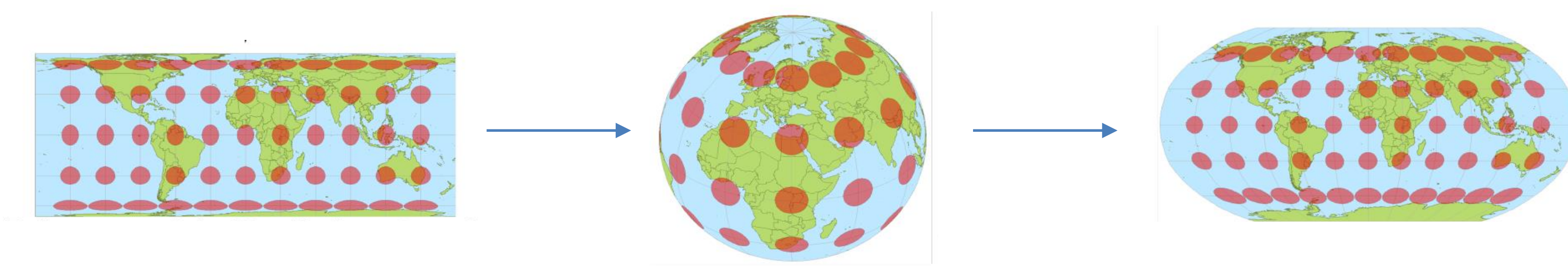
**Georeferenzierung** umfasst (grob gesagt) den Workflow, welcher den externen Raumbezug der eigentlichen Daten in den Raumbezug des GIS-Projektes umwandelt. Es muss der externe Raumbezug (z.B. geographische Koordinaten (λ,φ) oder Pixel in einem Luftbild) in ein üblicherweise durch (x,y)-Koordinaten gegebenen Raumbezug des GIS Projektes (lokales Koordinatensystem).

Wir unterscheiden oft zwei Koordinatenoperation:

- **Koordinatenumwandlung**
- Koordinatentransformartion

$$(x', y') \xrightarrow{T_1^{-1}} (\lambda, \varphi) \xrightarrow{T_2} (x'', y'')$$

Mit den bekannten Kartenprojektionen $T_1$ und $T_2$.



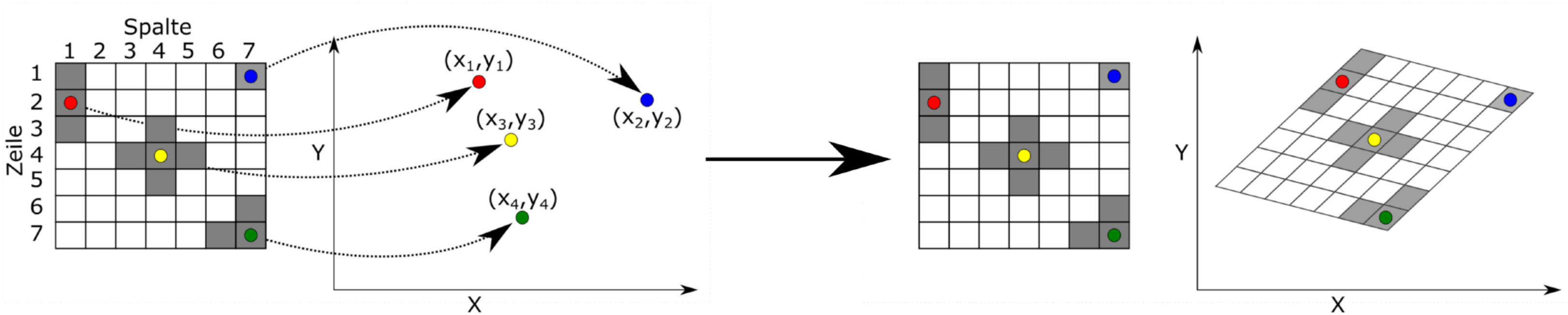
# Koordinatentransformation

**Georeferenzierung** umfasst (grob gesagt) den Workflow, welcher den externen Raumbezug der eigentlichen Daten in den Raumbezug des GIS-Projektes umwandelt. Es muss der externe Raumbezug (z.B. geographische Koordinaten (λ,φ) oder Pixel in einem Luftbild) in ein üblicherweise durch (x,y)-Koordinaten gegebenen Raumbezug des GIS Projektes (lokales Koordinatensystem).

Wir unterscheiden oft zwei Koordinatenoperation:

- Koordinatenumwandlung
- **Koordinatentransformation**

Koordinatentransformationen werden verwendet, wenn kein expliziter Formalismus bekannt ist (z.B. zwei Rasterbilder liegen jeweils nur in relativen Pixelkoordinaten vor).




Transformation basiert auf **Passpunkten** in den jeweiligen lokalen Koordinaten

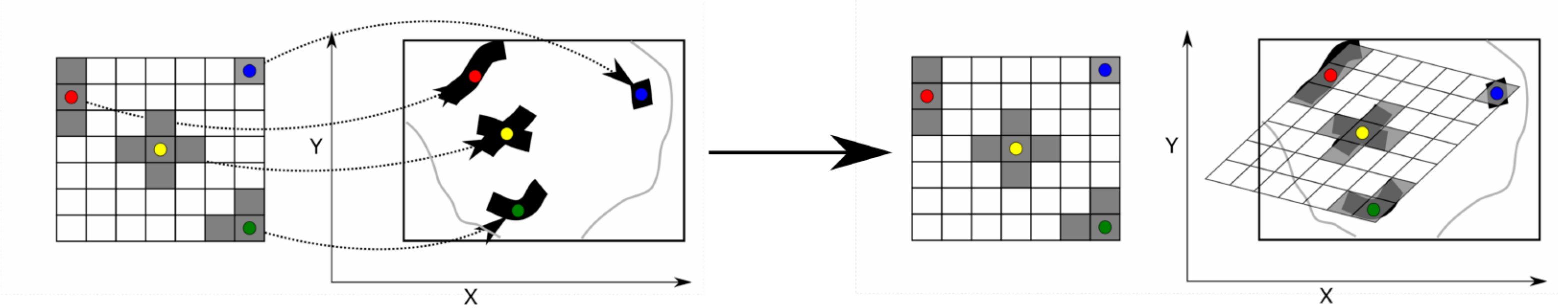
# Koordinatentransformation

**Georeferenzierung** umfasst (grob gesagt) den Workflow, welcher den externen Raumbezug der eigentlichen Daten in den Raumbezug des GIS-Projektes umwandelt. Es muss der externe Raumbezug (z.B. geographische Koordinaten (λ,φ) oder Pixel in einem Luftbild) in ein üblicherweise durch (x,y) Koordinaten gegebenen Raumbezug des GIS Projektes (lokales Koordinatensystem).

Wir unterscheiden oft zwei Koordinatenoperation:

- Koordinatenumwandlung
- **Koordinatentransformartion**

Koordinatentransformationen werden verwendet, wenn kein expliziter Formalismus bekannt ist (z.B. zwei Rasterbilder liegen jeweils nur in relativen Pixelkoordinaten vor).




Transformation basiert auf Zuweisung von Objekten zu Passpunkten im Zielkoordinatensystem

# Koordinatentransformation

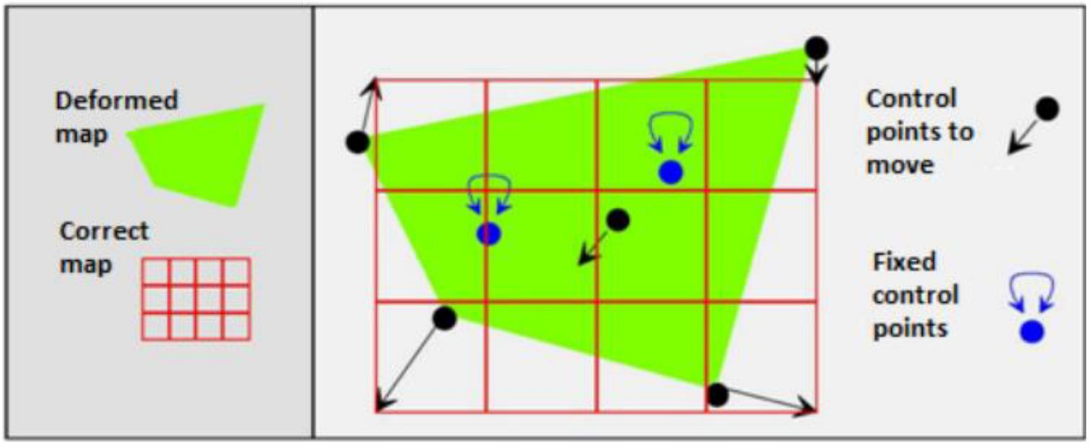
**Affin-lineare Abbildungen** können zwischen lokalen Koordinaten (x,y) und (x',y') in zwei verschiedenen Koordinatensystemen transformieren.

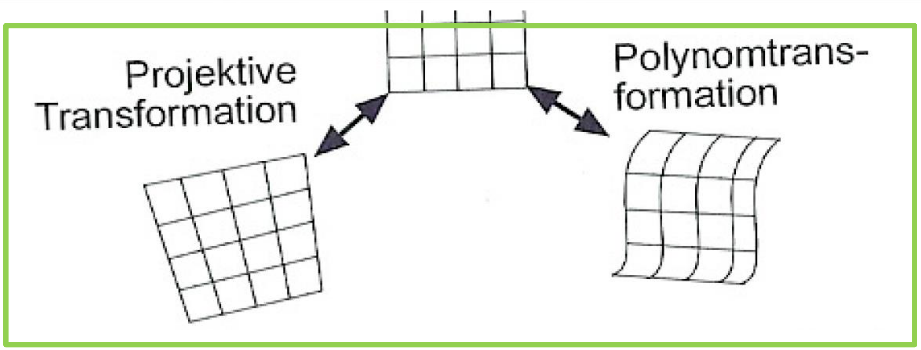
$\begin{pmatrix} x' \\ y' \end{pmatrix} = \mathrm{M}$

wobei $\mathrm{M}$

$\mathrm{M} = \begin{pmatrix} m_1 \\ m_2 \end{pmatrix}$

die z.B. au



# Koordinatentransformation

**Affin-lineare Abbildungen** können zwischen lokalen Koordinaten (x,y) und (x',y') in zwei verschiedenen Koordinatensystemen transformieren.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \mathbf{M} \begin{pmatrix} x \\ y \end{pmatrix} + \mathbf{v},$$

wobei $\mathbf{M}$ eine Matrix ist

$$\mathbf{M} = \begin{pmatrix} m_{1,1} & m_{1,2} \\ m_{2,1} & m_{2,2} \end{pmatrix} = \mathbf{RS}$$

die z.B. aus Rotationen $\mathbf{R}$ und Skalierungen und Scherungen $\mathbf{S}$ besteht.

Auch als Polynome 1. Grades ausdrückbar:

$$x' = m_{1,1}x + m_{1,2}y + v_1$$
$$y' = m_{2,1}x + m_{2,2}y + v_2$$

# Koordinatentransformation

**Verzerrungen** und **nichtlineare Effekte** benötigen Transformationen höherer Ordnung (z.B. Polynome vom Grad zwei, drei, etc.): Für Grad zwei sähe es wie folgt aus

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \mathbf{M} \begin{pmatrix} x \\ y \end{pmatrix} + (x, y)\, \mathbf{N} \begin{pmatrix} x \\ y \end{pmatrix} + \mathbf{v},$$

wobei $\mathbf{M} \in \mathbb{R}^{2\times 2}$ und $\mathbf{N} \in \mathbb{R}^{2\times 2\times 2}$ geeignete Matrizen sind.

vgl. Matlab Beispiel transformations.m / coordinateTransformations.m

Auch als Polynome 2. Grades ausdrückbar:

$$x' = m_{1,1}x + m_{1,2}y + v_1 + n_{1,1,1}x^2 + n_{1,2,1}y^2 + n_{1,1,2}xy + n_{1,2,2}yx$$
$$y' = m_{2,1}x + m_{2,2}y + v_2 + n_{2,1,1}x^2 + n_{2,2,1}y^2 + n_{2,1,2}xy + n_{2,2,2}yx$$

mit

$$n_{1,1,2}xy + n_{1,2,2}yx = n_{1,,2}xy$$
$$n_{2,1,2}xy + n_{2,2,2}yx = n_{2,,2}xy$$

# Koordinatentransformation

Sind die Matrizen **M** und **N** bekannte, so kann einfach zwischen den Koordinaten gewechselt werden. Sind Sie jedoch nicht bekannt, so müssen die Einträge erstmal durch Passpunkte bestimmt werden. Dazu ist das Lösen von (linearen und ggfs. nichtlinearen) Gleichungssystemen nötig: Etwa bei drei Passpunkten im affin-linearen Fall

$$\begin{pmatrix} x_1' \\ x_2' \\ x_3' \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \begin{pmatrix} m_{1,1} \\ m_{1,2} \\ v_1 \end{pmatrix}, \qquad \begin{pmatrix} y_1' \\ y_2' \\ y_3' \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \begin{pmatrix} m_{2,1} \\ m_{2,2} \\ v_2 \end{pmatrix}$$

wobei $m_{1,1}, \ldots, m_{2,2}$ die (unbekannten) Einträge der Matrix $\mathbf{M}$ sind und $v_1, v_2$ die (unbekannten) Einträge des Vektors $\mathbf{v}$ sind.

# Koordinatentransformation

Sind die Matrizen **M** und **N** bekannte, so kann einfach zwischen den Koordinaten gewechselt werden. Sind Sie jedoch nicht bekannt, so müssen die Einträge erstmal durch Passpunkte bestimmt werden. Dazu ist das Lösen von (linearen und ggfs. nichtlinearen) Gleichungssystemen nötig: Etwa bei drei Passpunkten im affin-linearen Fall

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \begin{pmatrix} m_{1,1} \\ m_{1,2} \\ v_1 \end{pmatrix}, \qquad \begin{pmatrix} y'_1 \\ y'_2 \\ y'_3 \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \begin{pmatrix} m_{2,1} \\ m_{2,2} \\ v_2 \end{pmatrix}$$

wobei $m_{1,1}, \ldots, m_{2,2}$ die (unbekannten) Einträge der Matrix $\mathbf{M}$ sind und $v_1, v_2$ die (unbekannten) Einträge des Vektors $\mathbf{v}$ sind.

Gibt es ungünstige Positionen der Passpunkte in obigem Fall? Falls ja, was wäre ein Beispiel und welche Probleme könnten auftauchen?

# Koordinatentransformation

Sind die Matrizen **M** und **N** bekannte, so kann einfach zwischen den Koordinaten gewechselt werden. Sind Sie jedoch nicht bekannt, so müssen die Einträge erstmal durch Passpunkte bestimmt werden. Dazu ist das Lösen von (linearen und ggfs. nichtlinearen) Gleichungssystemen nötig: Etwa bei drei Passpunkten im affin-linearen Fall

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \begin{pmatrix} m_{1,1} \\ m_{1,2} \\ v_1 \end{pmatrix}, \qquad \begin{pmatrix} y'_1 \\ y'_2 \\ y'_3 \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \begin{pmatrix} m_{2,1} \\ m_{2,2} \\ v_2 \end{pmatrix}$$

wobei $m_{1,1}, \ldots, m_{2,2}$ die (unbekannten) Einträge der Matrix $\mathbf{M}$ sind und $v_1, v_2$ die (unbekannten) Einträge des Vektors $\mathbf{v}$ sind.

Gibt es ungünstige Positionen der Passpunkte in obigem Fall? Falls ja, was wäre ein Beispiel und welche Probleme könnten auftauchen?

Wieviele Passpunkte braucht man jeweils für ein affin-lineare, eine quadratische (Grad zwei), und kubische (Grad drei) Transformation?

3, 6, bzw. 10 (da jeweils 6, 12, bzw. 20 Freiheitsgrade)

# Koordinatentransformation

Sind die Matrizen **M** und **N** bekannte, so kann einfach zwischen den Koordinaten gewechselt werden. Sind Sie jedoch nicht bekannt, so müssen die Einträge erstmal durch Passpunkte bestimmt werden. Dazu ist das Lösen von (linearen und ggfs. nichtlinearen) Gleichungssystemen nötig: Etwa bei drei Passpunkten im affin-linearen Fall

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \begin{pmatrix} m_{1,1} \\ m_{1,2} \\ v_1 \end{pmatrix}, \qquad \begin{pmatrix} y'_1 \\ y'_2 \\ y'_3 \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \begin{pmatrix} m_{2,1} \\ m_{2,2} \\ v_2 \end{pmatrix}$$

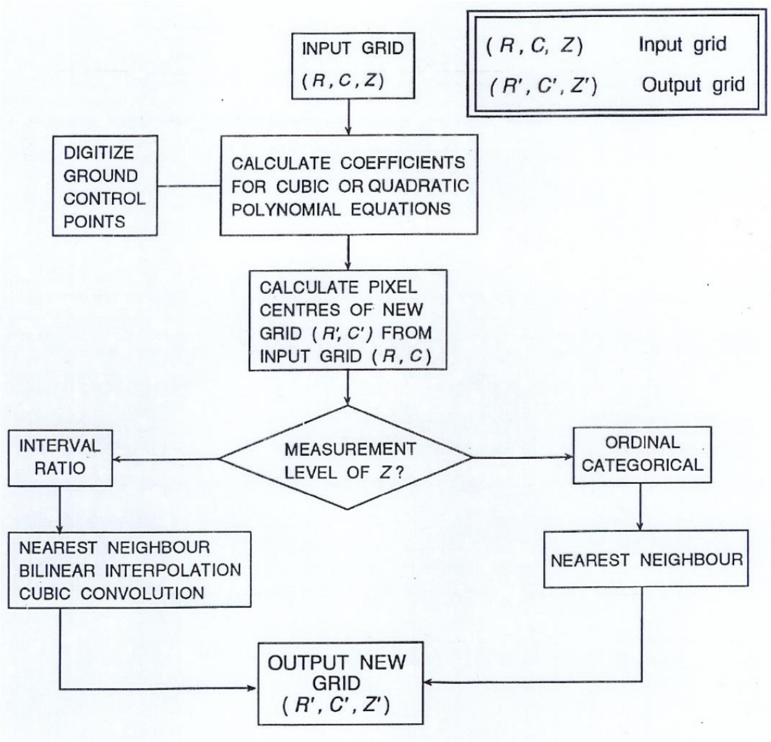
wobei $m_{1,1}, \ldots, m_{2,2}$ die (unbekannten) Einträge der Matrix $\mathbf{M}$ sind und $v_1, v_2$ die (unbekannten) Einträge des Vektors $\mathbf{v}$ sind.

Wie ändert sich die Anzahl der Freiheitsgrade z.B. im affin-linearen Fall, wenn wir die Dimension erhöhen, etwa Dimension drei, vier, fünf? Wie ändert sich die Anzahl der nötigen Passpunkte?

12, 20, bzw. 30 (jeweils 4, 5, bzw. 6 Passpunkte nötig)

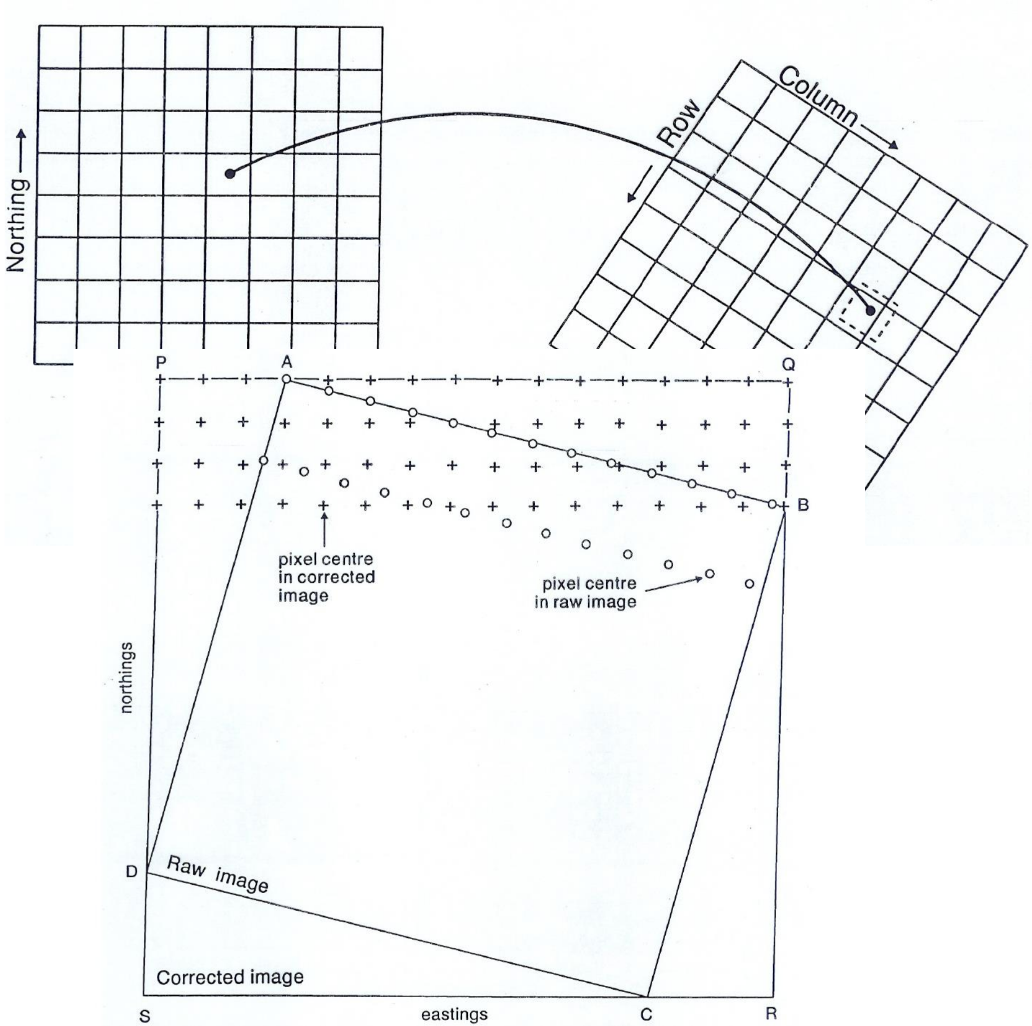
# Georeferenzierung von Rasterdaten

Rasterdaten sind üblicherweise durch Attributwerte in Pixeln eines kartesischen Gitters. Nach Anwendung einer Koordinatentransformation befinden sich die transformierten Pixel in der Regel nicht mehr auf einem kartesischen Gitter und die Attributwerte müssen interpoliert werden.



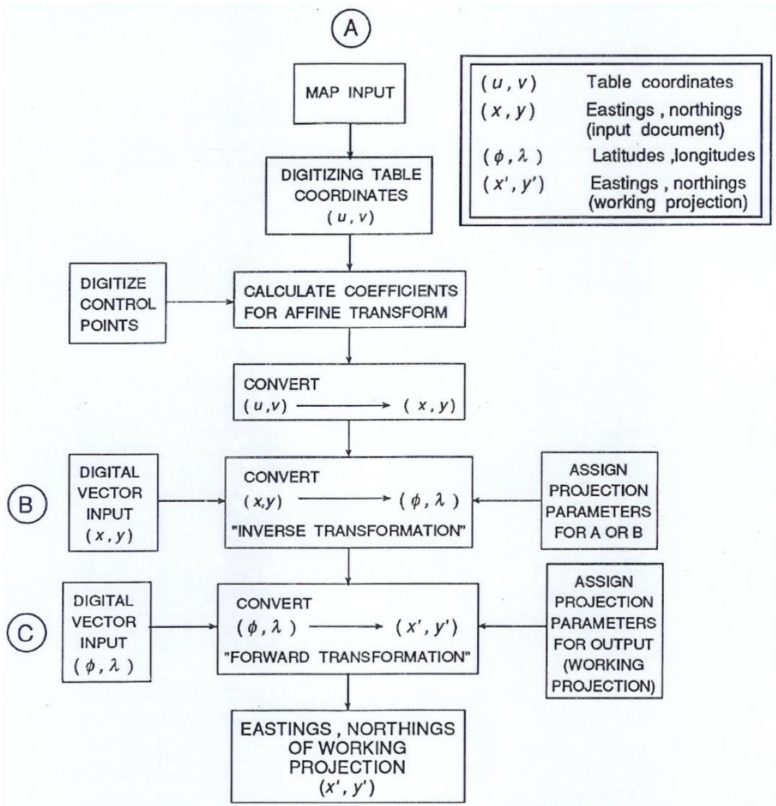

Bonham-Carter: Geographic Information Systems for Geoscientists: Modeling with GIS (1994)

# Georeferenzierung von Rasterdaten



# Georeferenzierung von Vektordaten

Vektordaten benötigen kein zugrundeliegendes kartesisches Gitter. Datensätze (hier z.B. A,B,C) können direct in ein gemeinsames Koordinatensystem überführt werden.




Bonham-Carter: Geographic Information Systems for Geoscientists: Modeling with GIS (1994)

# Übersicht

1. Definitionen, Funktionen, Anwendungen (Vorlesung 1)
2. Koordinatensysteme und -transformationen (Vorlesung 2+3)
3. **Räumliche Datenmodellierung**
4. Vermaschungen
5. Räumliche Interpolation
6. Transformationen, Filtermethoden, Sonstiges

# Räumliche Datenmodellierung

- **Datenmodell**: konzeptionelles Schema zu Organisation der Daten (z.B. Vektor- oder Rastermodell)
- **Datenstruktur**: Formen der Repräsentation eines Datenmodells (z.B. Matrixdarstellung für ein Rastermodell
- **Datenformat**: Möglichkeiten der Speicherung einer Datenstruktur (z.B. für eine Matrix im Rastermodell: array von durch Kommata getrennten Attributwerten)

# Objekttypen

Räummliche Objekte können in verschiedener Gruppen sortiert oder anhand verschiedener Eigenschaften charakterisiert werden.

- **Dimension**: 0-d (Punktobjekte), 1-d (Linienobjekte), 2-D (Flächenobjekte), 3-D (Volumenobjekte), …; auch 2.5-D Objekte begrifflich möglich (2-D Objekte mit zusätzlichem räumlichen Parameter)

- **Natürliche vs. künstliche Objekte**:
  - natürliche Objekte = real auftretende Strukturen wie Flüsse, Gebäude, Gesteinskörper;
  - künstliche Objekte = von natürlichen physischen Objekten abgeleitete Darstellungen (Pixel, Isolinien, …) oder nicht-physische motivierte Objekte (Verwaltungsbezirke, Staatsgrenzen, …)

- **Diskrete vs. kontinuierliche Objekte**:
  - diskret = es können nur endlich viele Werte oder isolierte Werte angenommen werden (Lithologien, Einwohnerzahl, …),
  - kontinuierlich = alle ‘Zwischenwerte’ können ohne ‘Sprünge’ angenommen warden (Höhen, Stoffkonzentrationen, …)

=> vergl. Skalen-Begriff aus “Datenanalyse/Statistik”

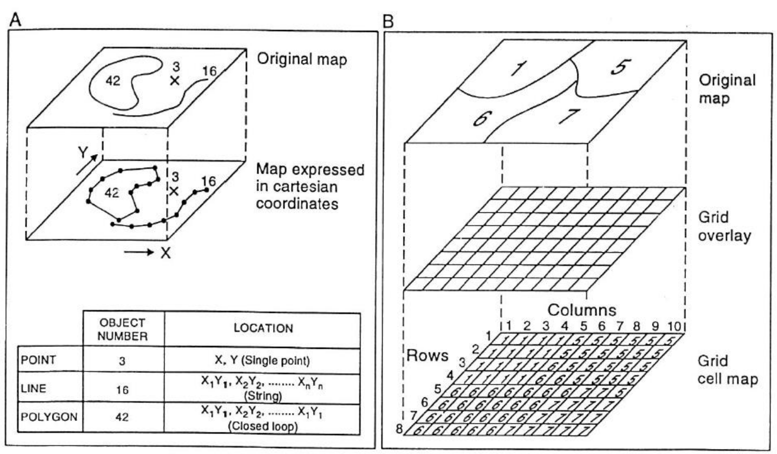
# Objekttypen

- **Auflösungsbegrenzt vs. Definitionsbegrenzt**:
  - auflösungsbegrenzt = die Anzahl und Verteilung der Messdaten schränkt Aussagen über das Objekt ein (z.B. Genauigkeit der Küstenlinie hängt von Auflösung einer Luftbildaufnahme ab);
  - definitionsbegrenzt = zusätzliche Abhängigkeit von definierten Grenzwerten oder Attributen (z.B. Grenzwerte für Stoffkonmzentrationen oder Konturlinie basierend auf vorgegebener Höhe)

- **Unregelmässig vs. regelmässig**:
  - regelmässig = Menge gleichförmiger Objekte (Pixel, Daten auf regelmässigem Gitter, …);
  - unregelmäßig = Menge von Objekten unregelmässiger Form (Triangulierung mit nichtkongruenten Dreiecken, Waldgrenzen, …)

# Raster- vs. Vektormodell

- **Rastermodell**:
  - gleichmässige Gitterstruktur
  - grundlegende Einheiten sind gleichförmig, üblicherweise Pixel (2-D) oder Voxel (3-D)
  - jeder grundlegenden Einheit wird ein Attributwert zugeordnet
  - grundlegenede Einheiten geben Auflösung vor

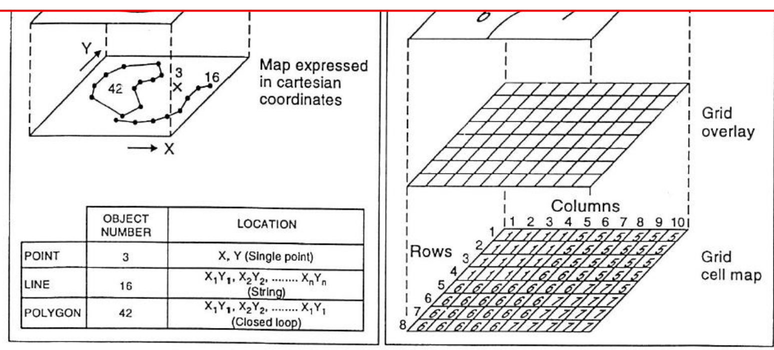


# Raster- vs. Vektormodell

- **Rastermodell**:
  - gleichmässige Gitterstruktur
  - grundlegende Einheiten sind gleichförmig, üblicherweise Pixel (2-D) oder Voxel (3-D)
  - jeder grundlegenden Einheit wird ein Attributwert zugeordnet
  - grundlegenede Einheiten geben Auflösung vor

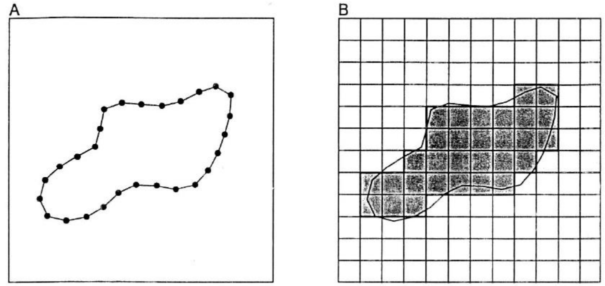
- **Vektormodell**:
  - Grundelegende Einheiten sind Vertexe (0-D Punktobjekte); und darauf aufbauend Linien/Polygonzüge, Polygone, …
  - Exakte Koordinaten für Vertexe, kein Auflösungsverlust
  - jedem Element kann ein oder mehrere Attributwerte zugeordnet werden

Sagen Sie Bitte niemals, einfach nur: „die Grundelemente eines Vektormodells sind Vektoren“!



# Raster- vs. Vektormodell

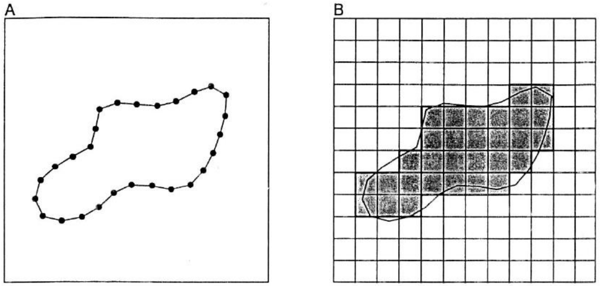
- **Rastermodell**:
  - ausschliesslich flächenhafte Betrachtung von Objekten: jedes Pixel beschreibt eine (möglicherweise sehr kleine) Fläche
  - größere Einzelobjekte lassen sich nur über Attributwerte voneinander abgrenzen
  - Datenerfassung einfach und effizient
  - Position und Nachbarschaftsbeziehungen sehr einfach über Gitteranordnung definiert
  - Unter Umständen sehr Speicheraufwendig



# Raster- vs. Vektormodell

- **Rastermodell**:
  - ausschliesslich flächenhafte Betrachtung von Objekten: jedes Pixel beschreibt eine (möglicherweise sehr kleine) Fläche
  - größere Einzelobjekte lassen sich nur über Attributwerte voneinander abgrenzen
  - Datenerfassung einfach und effizient
  - Position und Nachbarschaftsbeziehungen sehr einfach über Gitteranordnung definiert
  - Unter Umständen sehr Speicheraufwendig

- **Vektormodell**:
  - Auflösungsunabhängig
  - größere Einzelobjekte als Einheit darstellbar
  - Effiziente Speicherung und Darstellung
  - Nachbarschaftsbeziehungen müssen festgelegt werden, aufwendigeres Anlegung der Datenstruktur



# Raster- vs. Vektormodell

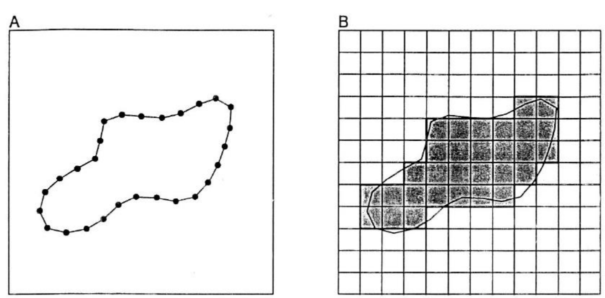
Bestimmung von Umfang und Fläche eines Objektes

- **Rastermodell**:
  - Fläche = Anzahl Pixel x Fläche pro Pixel
  - Umfang = Anzahl der Randpixel x Länge einer Pixelkante

Ist die Formel für den Umfang so komplett korrekt? Bei welchen Pixeln in unterem Bild müsste aufgepasst werden?

Obige Umfangs-Formel ist nur eine grobe Abschätzung, exakt wäre:

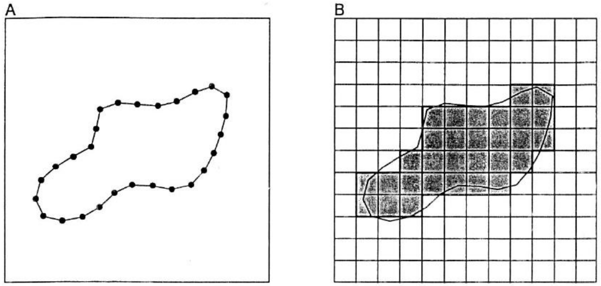
$$\text{Umfang} = \left( \sum_{\text{Anzahl Pixel}} \text{Anzahl der Nachbarpixel mit anderem Attribut} \right) \cdot \text{Kantenlänge}$$



# Raster- vs. Vektormodell

Bestimmung von Umfang und Fläche eines Objektes

- **Rastermodell**:
  - Fläche = Anzahl Pixel x Fläche pro Pixel
  - Umfang = Anzahl aller Nachbarpixel x Länge einer Pixelkante (Schätzung!)
- **Vektormodell**:
  - Fläche = Summe der Fläche der Teildreiecke
  - Umfang = Summe der Länge der Berandungslinien

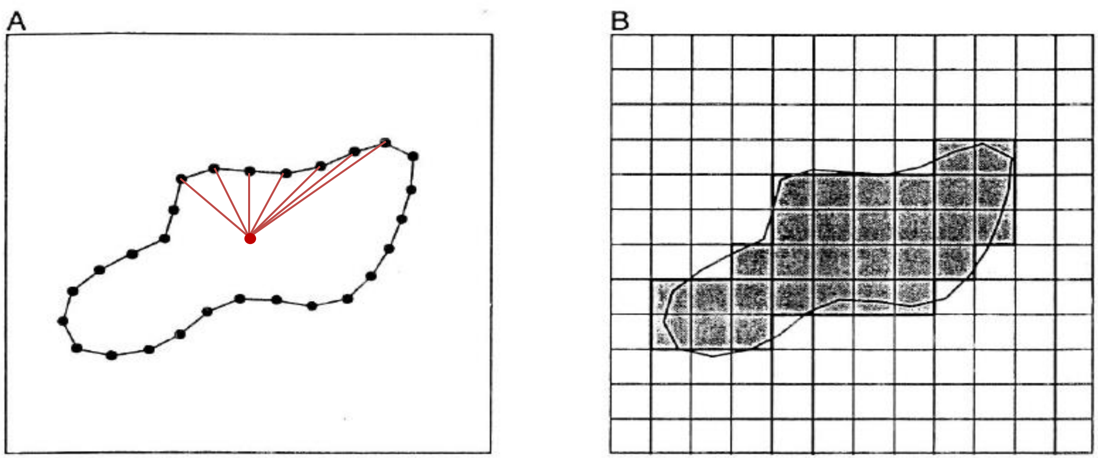


# Raster- vs. Vektormodell

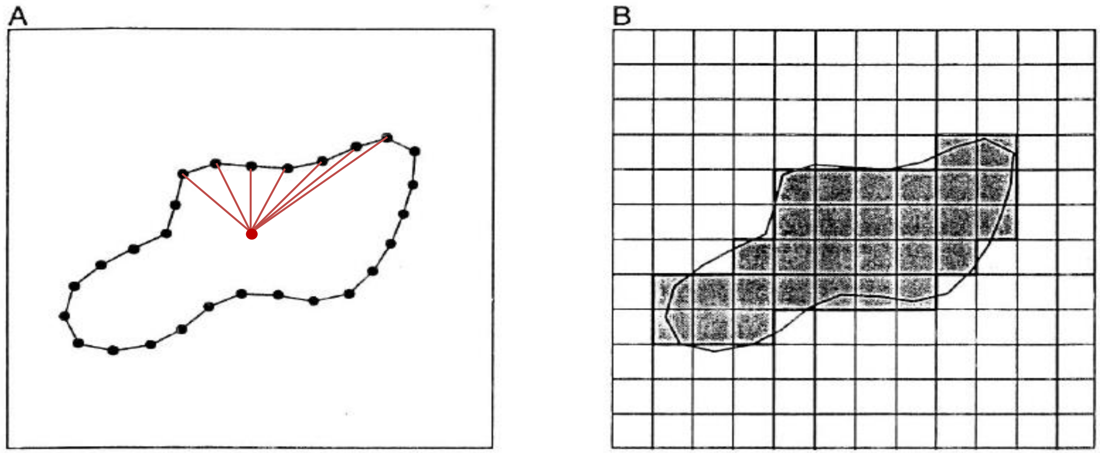
Bestimmung von Umfang und Fläche eines Objektes

- **Rastermodell**:
  - Fläche = Anzahl Pixel x Fläche pro Pixel
  - Umfang = Anzahl der Randpixel x Länge einer Pixelkante (Schätzung!)

- **Vektormodell**:
  - Fläche = Summe der Fläche der Teildreiecke
  - Umfang = Summe der Länge der Berandungslinien



# Raster- vs. Vektormodell




Die Fläche eines Dreiecks mit Eckpunkten $(x_1, y_1), (x_2, y_2), (x, y)$ ist

$$\frac{1}{2}|(x - x_1)(y_1 - y_2) + (x_1 - x_2)(y_1 - y)| = \left| \det \begin{pmatrix} x & x_1 & x_2 \\ y & y_1 & y_2 \\ 1 & 1 & 1 \end{pmatrix} \right|.$$

Für das gesamte Polygon im Vektormodell ergibt sich als Fläche

$$\frac{1}{2} \sum_{i=1}^{n} |x_i y_{i+1} - x_{i+1} y_i|.$$

Beachte, dass für letzteres Aufhebungseffekte bei der Summierung über alle Dreiecke eine Rolle spielen, so dass der Referenzpunkt $(x, y)$ herausfällt.