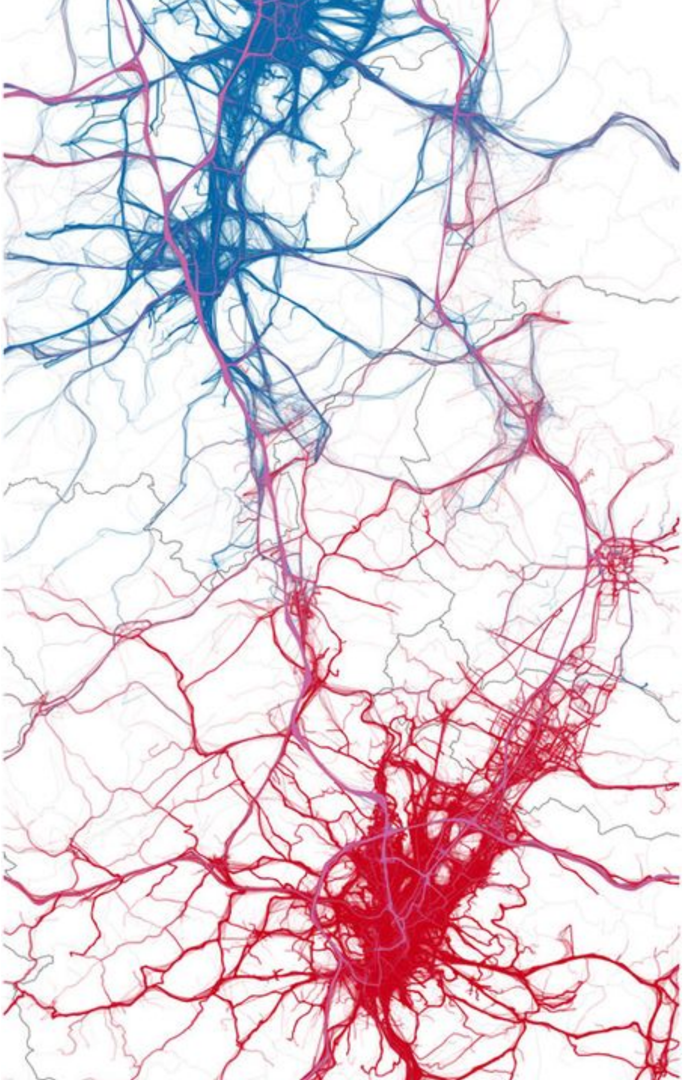


IN SUPREMÆ DIGNITATIS 1343

Consiglio Nazionale delle Ricerche

02 Fundamental Concepts

Geographic Information Systems (GIS)

- A computer system for capturing, storing, querying, analyzing, and displaying geospatial data.

- Geospatial data describe both the locations and characteristics of spatial features, e.g.:
  - to describe a road or a restaurant, we refer to its location (i.e., where it is) and its characteristics;
  - to describe a trajectory, we refer to the sequence of locations

Fundamental concepts

- **Geographic Coordinate Systems**
  reference system for locating points on the Earth's surface
- **Vector data model**
  representation of spatial features in GIS
- **Trajectory**
  sequence of points that describe an individual's movements
- **Spatial tessellation**
  division of the space into non-overlapping tiles
- **Flows**
  movements of groups of people between places

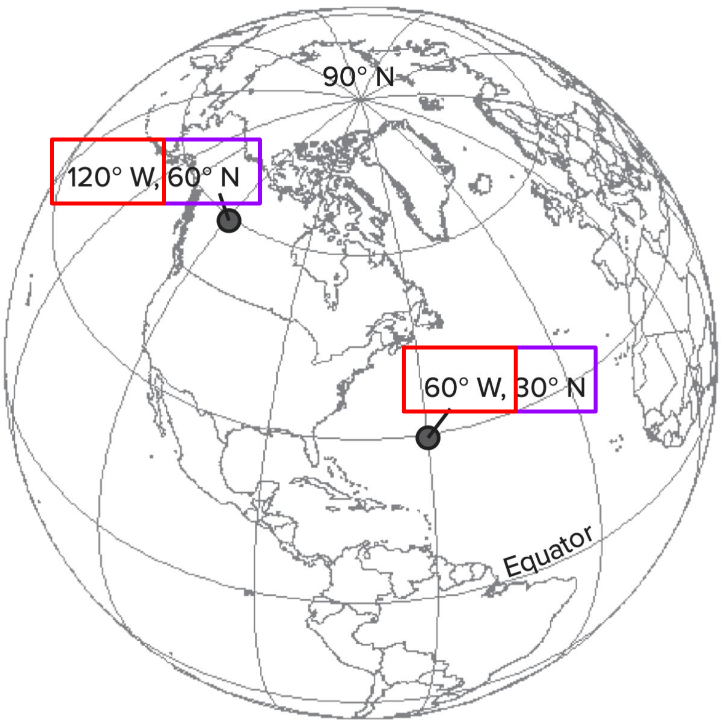
Geographic Coordinate Systems

reference system for locating points on Earth’s surface

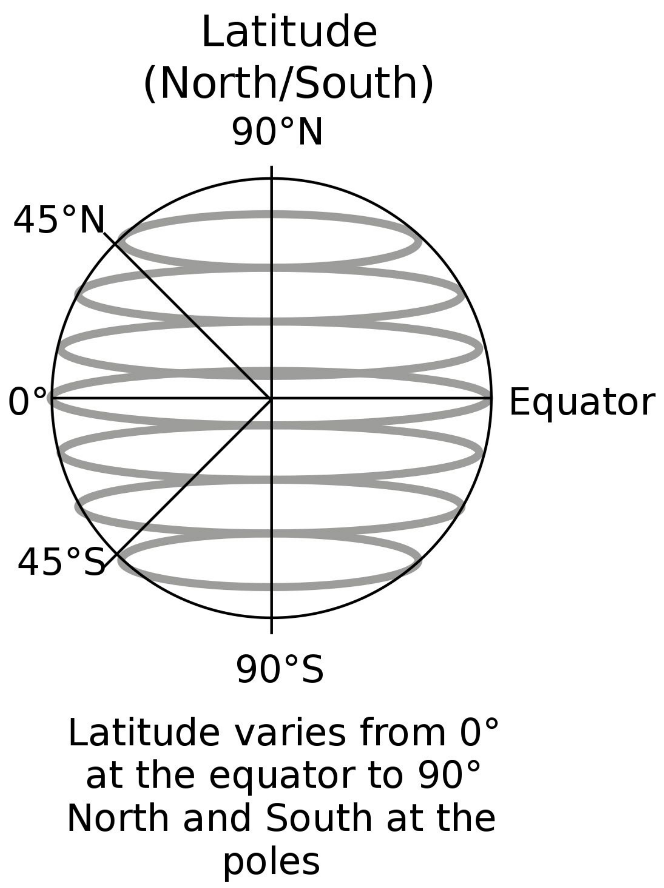
Geographic Coordinate System (GCS)

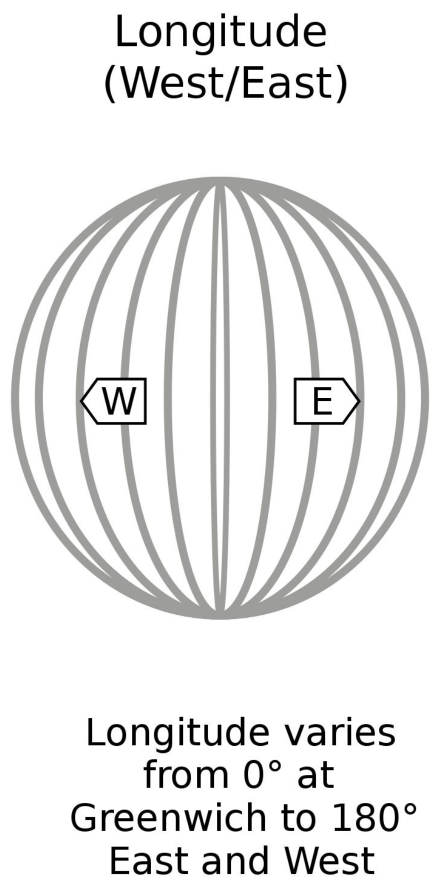
Reference system for locating points on the Earth's surface, based on two angles:

- **longitude** (Long)
  - angle E/W from the Prime Meridian (PM)
- **latitude** (Lat)
  - angle N/S of the equatorial plane



Introduction to Geographic Information Systems, Kang-Tsung Chang, MacGrawHill

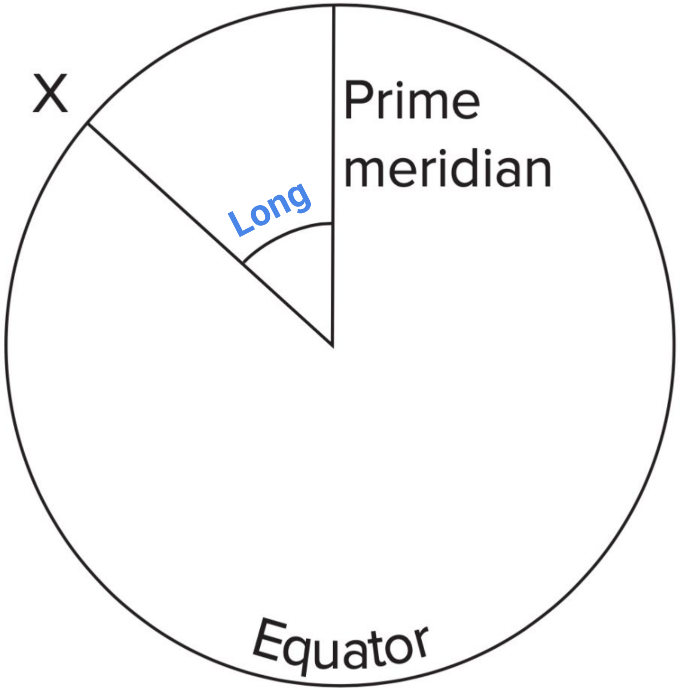

Latitude
(North/South)
90°N
45°N
0°
Equator
45°S
90°S
Latitude varies from 0° at the equator to 90° North and South at the poles


Longitude
(West/East)
W
E
Longitude varies from 0° at Greenwich to 180° East and West

source

Geographic Coordinate System (GCS)

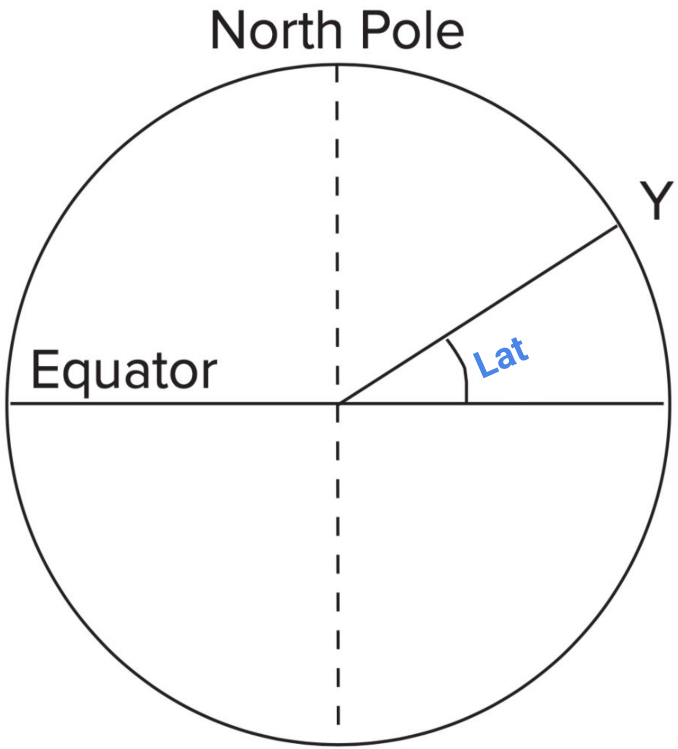
- **Meridians**: lines of equal Long
  - PM passes through Greenwich, UK (0°)
  - Long ∈ [0°, 180°] E/W of PM



Introduction to Geographic Information Systems, Kang-Tsung Chang, MacGrawHill

Geographic Coordinate System (GCS)

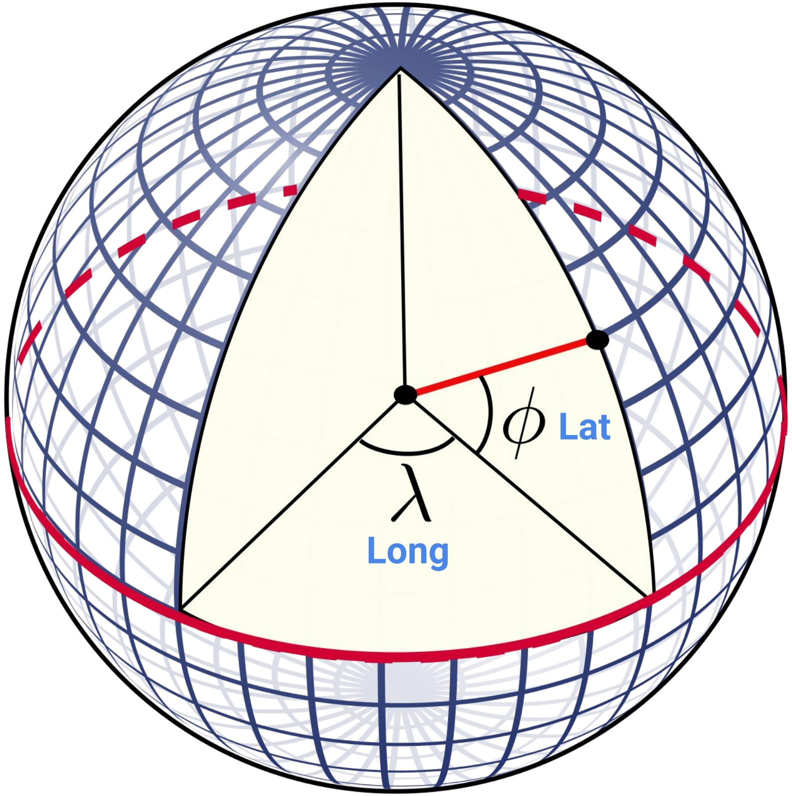
- **Parallels**: lines of equal Lat
  - Lat ∈ [0°, 90°] N/S of the equator (0°)



Introduction to Geographic Information Systems, Kang-Tsung Chang, MacGrawHill

Geographic Coordinate System (GCS)

- **Meridians**: lines of equal Long
  - PM passes through Greenwich, UK (0°)
  - Long ∈ [0°, 180°] E/W of PM
- **Parallels**: lines of equal Lat
  - Lat ∈ [0°, 90°] N/S of the equator (0°)



source

Geographic Coordinate System (GCS)

In a *plane coordinates* perspective:

- Long values are *x* values
  - Positive in eastern hemisphere, negative in western
  - $\in [-180°, 180°]$

- Lat values are *y* values
  - Positive if north of equator, negative otherwise
  - $\in [-90°, 90°]$

Geographic Coordinate System (GCS)

Long and Lat may be expressed in:

- **decimal degrees (DD)**
- degrees-minutes-seconds (DMS)
- radians (rad)

Conversion DMS to DD:

- 1 degree = 60 minutes; 1 minute = 60 seconds
- 45°52'30" (Lat) is 45.875° (45 + 52/60 + 30/3600)

Geographic Coordinate System (GCS)

Long and Lat may be expressed in:

- **decimal degrees (DD)**
- degrees-minutes-seconds (DMS)
- radians (rad)

Conversion DMS to rad:

- rad = DD * Pi/180
- one radian equals 57.2958°
- one degree equals 0.01745 rad

Geodetic Datum

Mathematical model of the Earth, serving as the *reference* for calculating the geographic coordinates

1. Long & Lat of an initial point (origin)
2. reference ***ellipsoid*** model, approximating Earth’s shape

Several local and global geodetic datums have been proposed

- A point on Earth can have different coordinates depending on the datum used

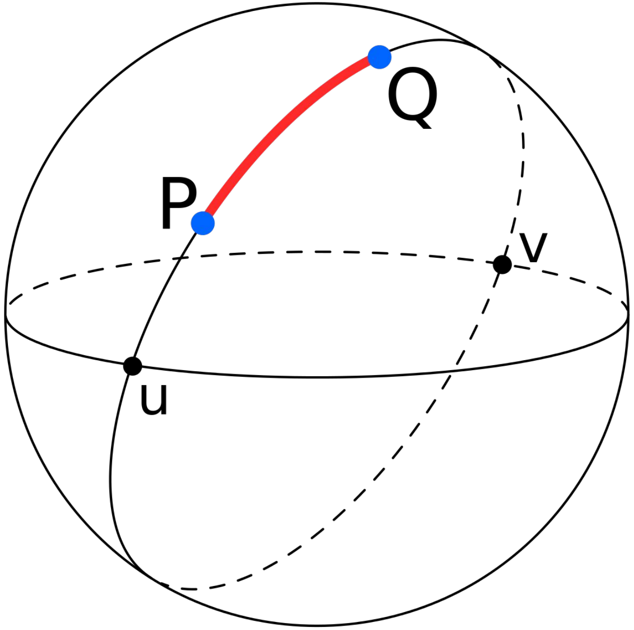
World Geodetic System 1984 (WGS84 or EPSG:4326)

- Used by the U.S. Department of Defense as global reference system for supporting positioning and navigation
- Datum for GPS readings:
  - satellites send their positions in WGS84 coordinates,
  - calculations in GPS receivers are based on WGS84
- Meridian of zero Long is the IERS Reference Meridian (close to Greenwich meridian) at the Lat of the Royal Observatory
- Earth surface is an ***oblate spheroid*** with equatorial radius:
  - a = 6 378 137 m at the equator
  - flattening f = 1/298.257223563.

Distance on the Earth surface

Straight lines are replaced by geodesics (great circles):

- Through two points (not antipodal) there is a unique great circle
- The two points separate the great circle into two arcs:
  - the shortest **(red)** is the great-circle distance
- The Earth is nearly spherical: great-circle distance are correct to within about 0.5%



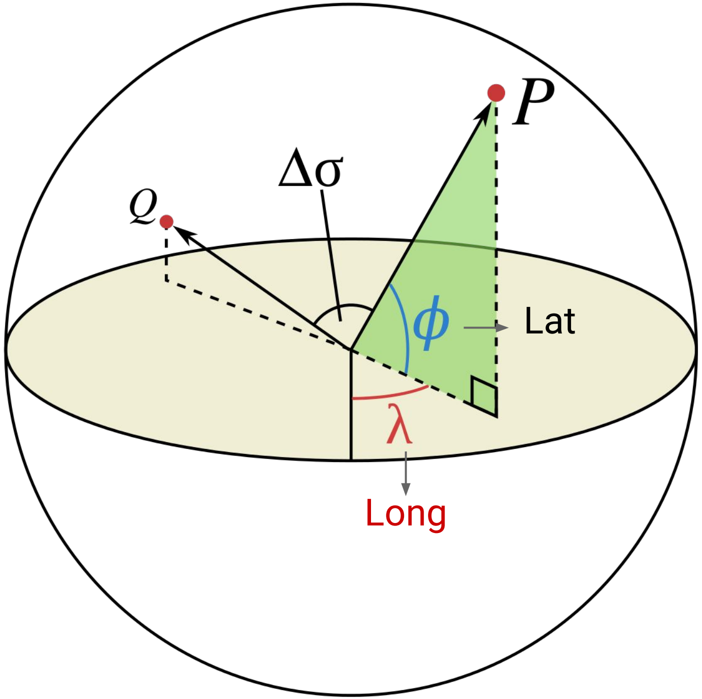
Distance on the Earth surface

Let P and Q be two points, the central angle between them is:

$$\Delta\sigma = \frac{d}{r}$$

spherical distance

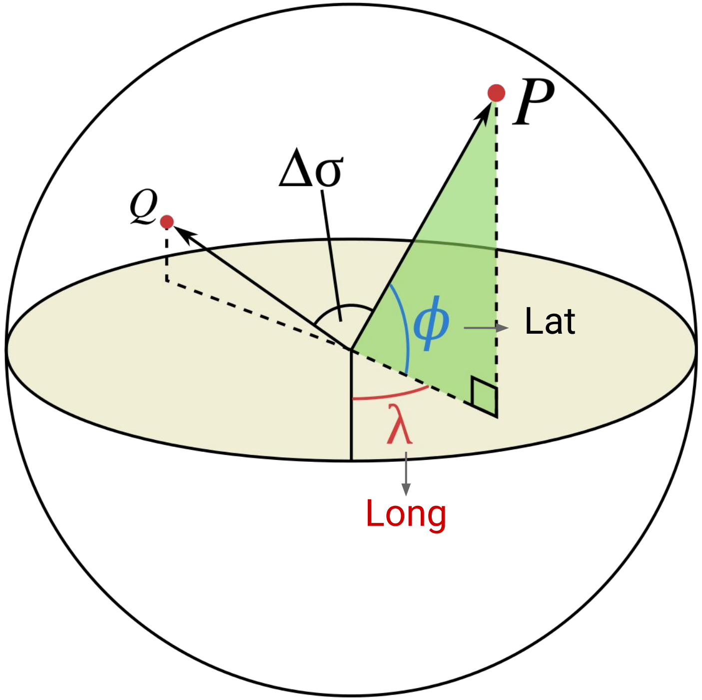
sphere's radius



Haversine formula

We can use the ***haversine formula*** to compute the great-circle distance between any two points given their Long and Lat:

$$\mathrm{hav}(\Delta\sigma) = \mathrm{hav}\left(\frac{d}{r}\right)$$



Haversine formula

Apply the arcsine (inverse sine) function:

$$d = 2r \arcsin\left(\sqrt{\operatorname{hav}(\Delta\sigma)}\right)$$

$$d = 2r \arcsin\left(\sqrt{\sin^2\left(\frac{\Delta Lat}{2}\right) + \cos Lat_P \cdot \cos Lat_Q \cdot \sin^2\left(\frac{\Delta Long}{2}\right)}\right)$$

Haversine formula

- An *approximation*, because the Earth is not a perfect sphere:
  - Earth radius varies from 6356.752km at the poles to 6378.137 km at the equator
  - the radius of curvature of a north-south line on the Earth's surface is 1% greater at the poles (≈6399.594 km) than at the equator (≈6335.439 km)

- The haversine formula and law of cosines cannot be guaranteed correct to better than 0.5%

What if…Earth would be flat?

just an Euclidean distance between points would be ok

$$D = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Did you know that...

It is not true that ancient and medieval people believed in flat Earth!

Globo crucigero

- A golden sphere with a cross on the top
- It represents the supremacy of Christ over the *world* and, if held in the hand of a ruler, the divine legitimacy of power




It is not true that ancient and medieval people believed in flat Earth!

Globo crucigero

- A golden sphere with a cross on the top
- It represents the supremacy of Christ over the *world* and, if held in the hand of a ruler, the divine legitimacy of power

The Flat Earth theory is recent!

- **XIX century:** the belief of flat Earth starts diffusing in England and the US (Rowbotham, Blount, Carpenter)

- **England, 1956:** Samuel Shenton found the Flat Earth Society

The Flat Earth theory is recent!

- **XIX** **century:** the nd and the US

- **Eng** **1956:** Sa

Insights

AB
BARBERO
mito della
terra piatta

BEHIND
THE CURVE
Behind the Curve
2018 | T | 1h 35m | Documentary Films
Meet the growing, worldwide community of theorists who defend the belief that the Earth is flat while living in a society who vehemently rejects it.
Flatearth

Vector data model

representation of spatial features in GIS

Vector data model

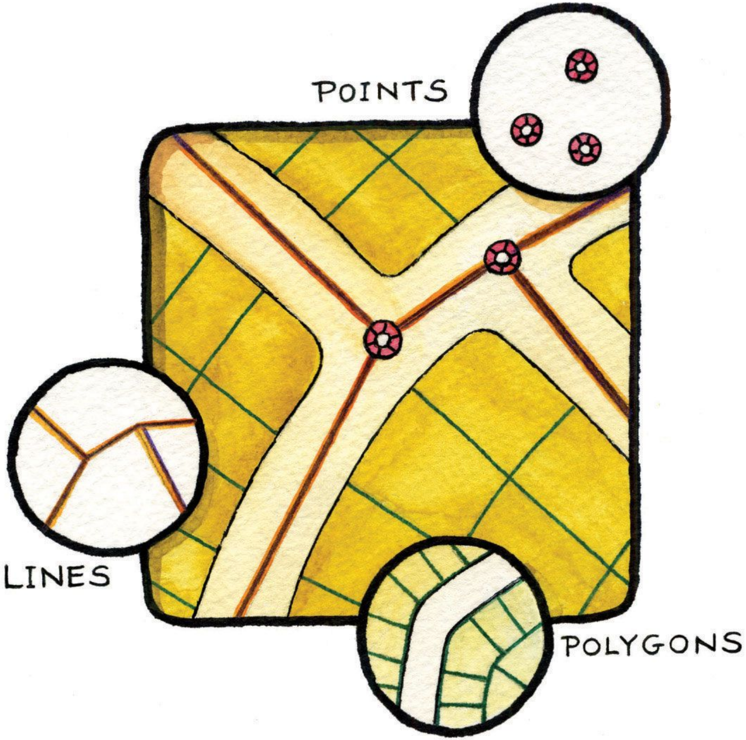
It uses discrete objects to represent spatial features:

1. representing `points`, `lines`, and `polygons` on an empty space

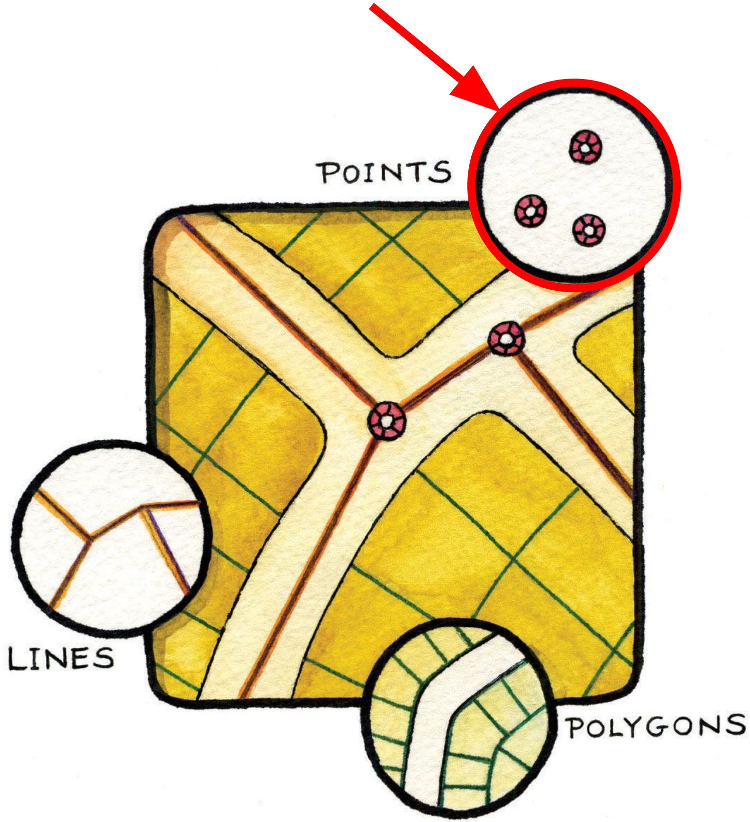
2. structuring the properties and spatial relationships of these geometric objects

3. coding and storing vector data in digital data files

Vector types

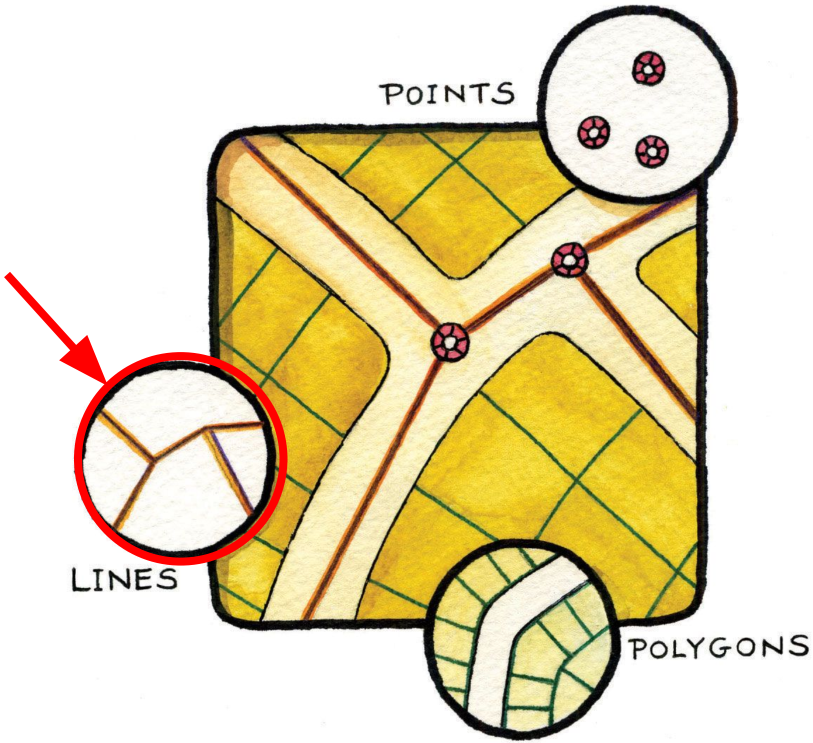


Vector types



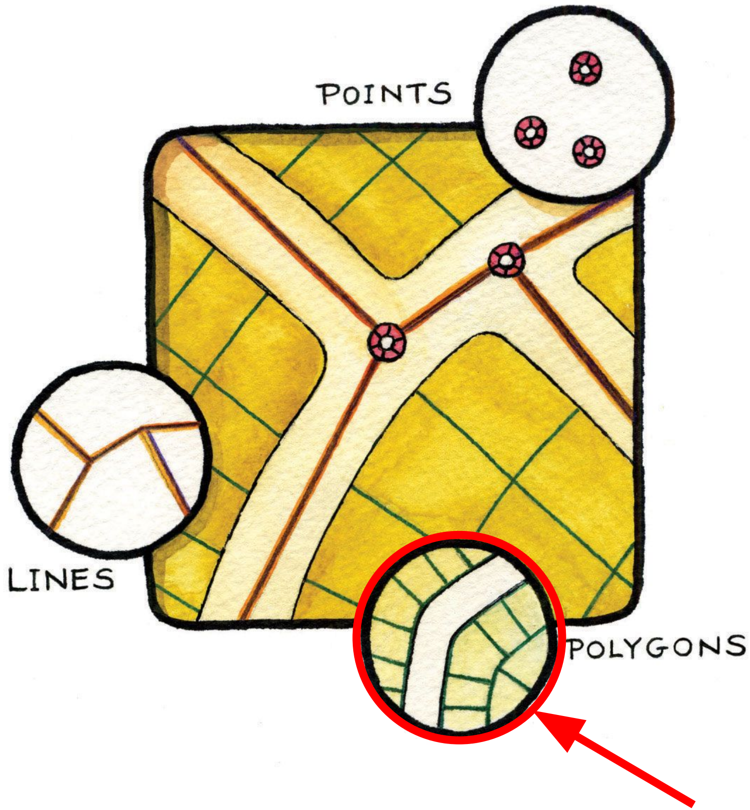
- `Point`: zero dimension
  - properties: *location* (xy coords)

Vector types

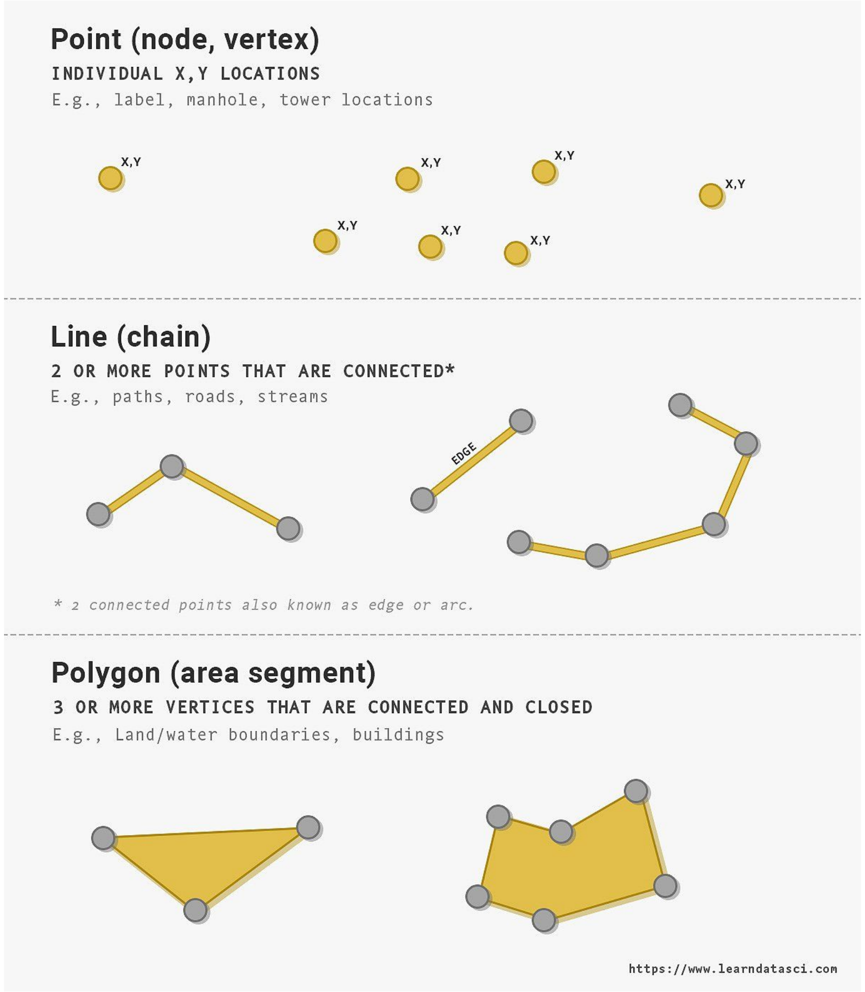


- `Point`: zero dimension
  - properties: *location* (xy coords)
- `Line`: one-dimensional
  - properties: *location* and *length*
  - has two end `Point`s
  - straight-line or curve

Vector types



- `Point`: zero dimension
  - properties: *location* (xy coords)
- `Line`: one-dimensional
  - properties: *location* and *length*
  - has two end `Point`s
  - straight-line or curve
- `Polygon`: two-dimensional
  - properties: *location*, *area*, *perimeter*
  - made of connected closed lines



- Google Maps
  - Point, Linestring, Linering, Polygon
- GeoJSON
  - Point, LineString, Polygon
- **Shapely**
  - Point, LineString, Polygon

Georelational Data Model

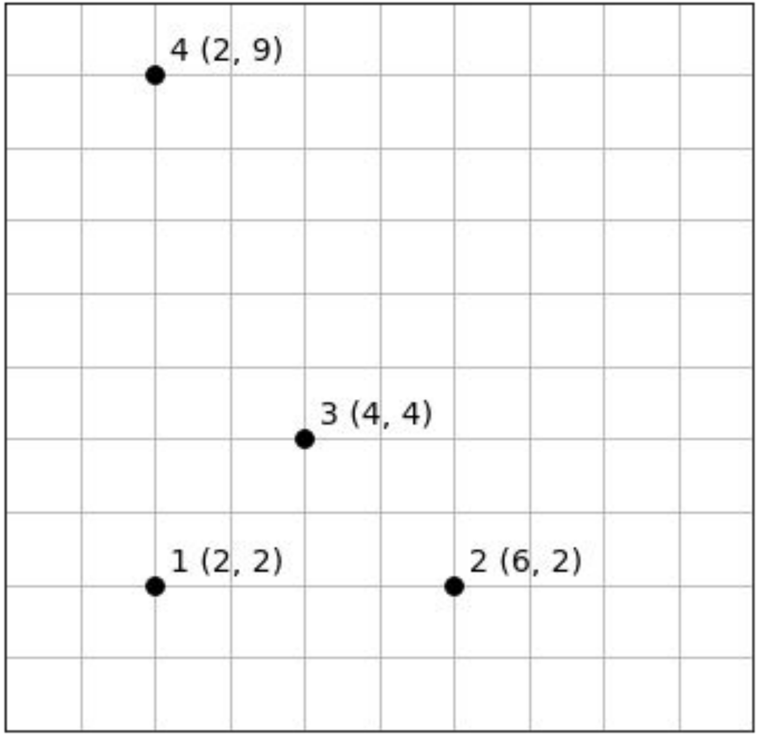
How do we represent geometric objects in a computer?

- Georelational data model: stores geometries and attributes separately
  - Topological: coverage
  - Non-topological: shapefile

Coverage: Point

Point list

ID	x, y
1	(2, 2)
2	(6, 2)
3	(4, 4)
4	(2, 9)



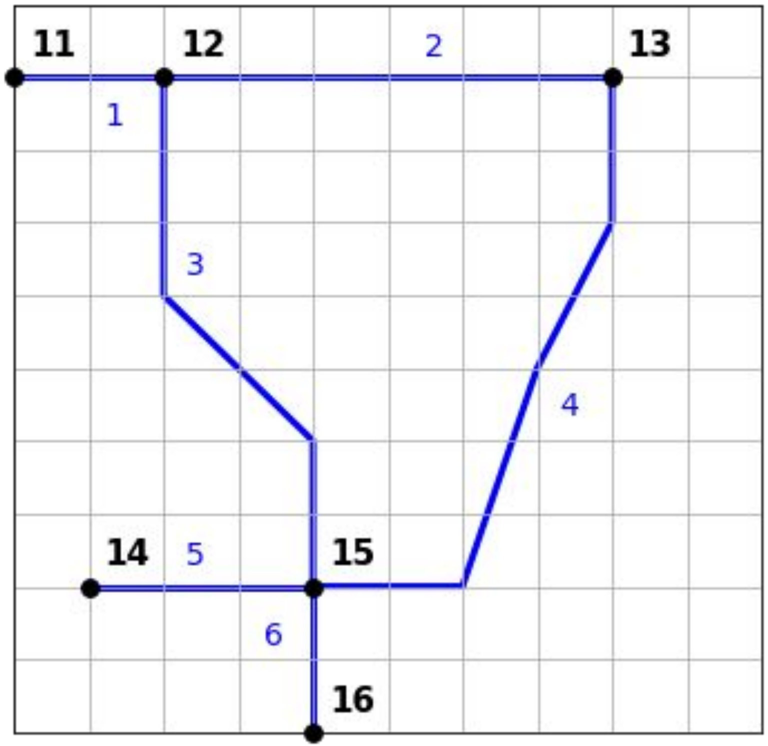
Coverage: Line

Point list

ID	x, y
11	(0, 9)
12	(2, 9)
13	(8, 9)
14	(1, 2)
15	(4, 2)
16	(4, 0

Arc-node list

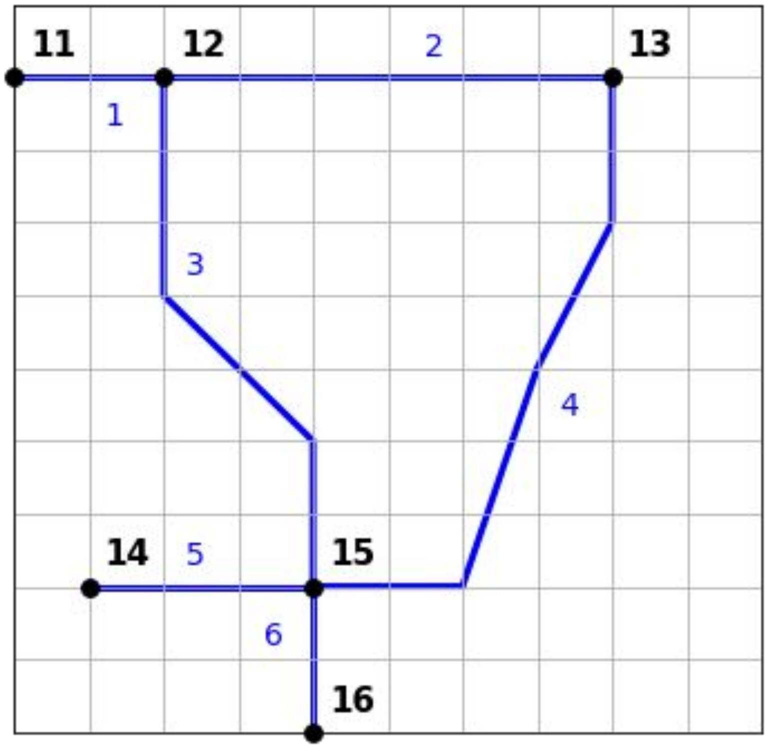
Arc	F-node	T-node
1	11	12
2	12	13
3	12	15
4	13	15
5	15	14
6	15	16



Coverage: Line

Arc-coordinate list

Arc	coordinates
1	(0,9) (2, 9)
2	(2, 9) (8, 9)
3	(2, 9) (2,6) (4, 4) (4, 2)
4	(8, 9) (8,7) (7, 5) (6, 2) (4, 2)
5	(4, 2) (1, 2)
6	(4, 2) (4, 0)



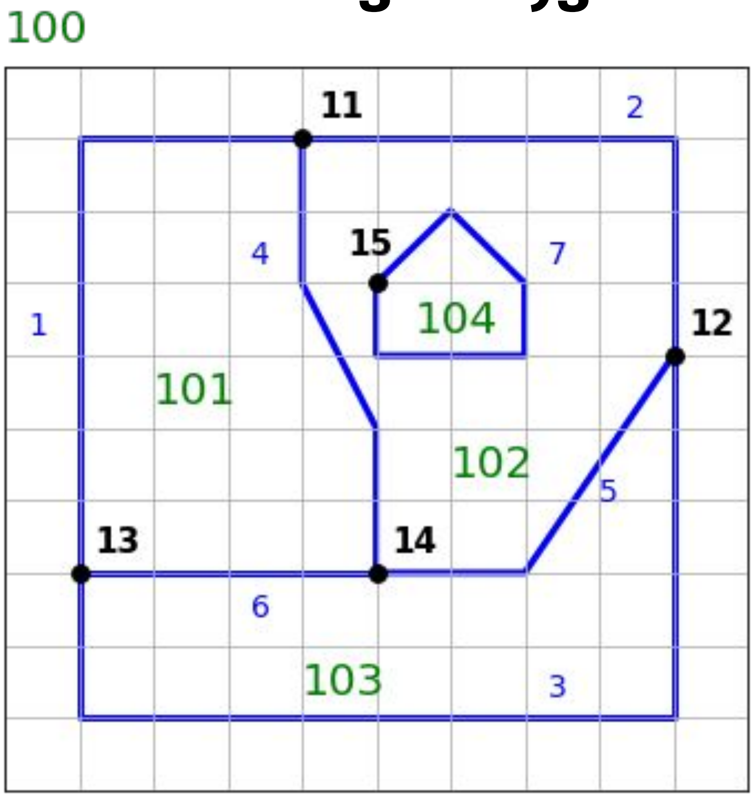
Coverage: Polygon

Point list

ID	x, y
11	(4, 9)
12	(9, 6)
13	(1, 3)
14	(5, 3)
15	(5, 7)

Arc-coordinate list

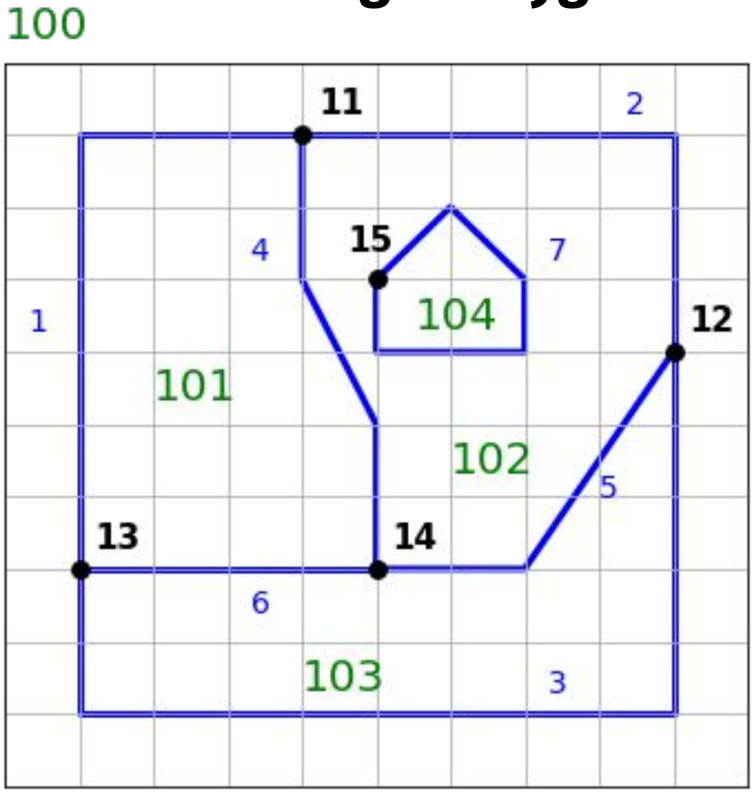
Arc	coordinates
1	(1, 3) (1, 9) (4, 9)
2	(4, 9) (9, 9) (9, 6)
3	(9, 6) (9, 1) (1, 1) (1, 3)
4	(4, 9) (4, 7) (5, 5) (5, 3)
5	(9, 6) (7, 3) (5, 3)
6	(5, 1) (1, 3)
7	(5, 7) (6, 8) (7, 7) (7, 6) (5, 6) (5, 7)



Coverage: Polygon

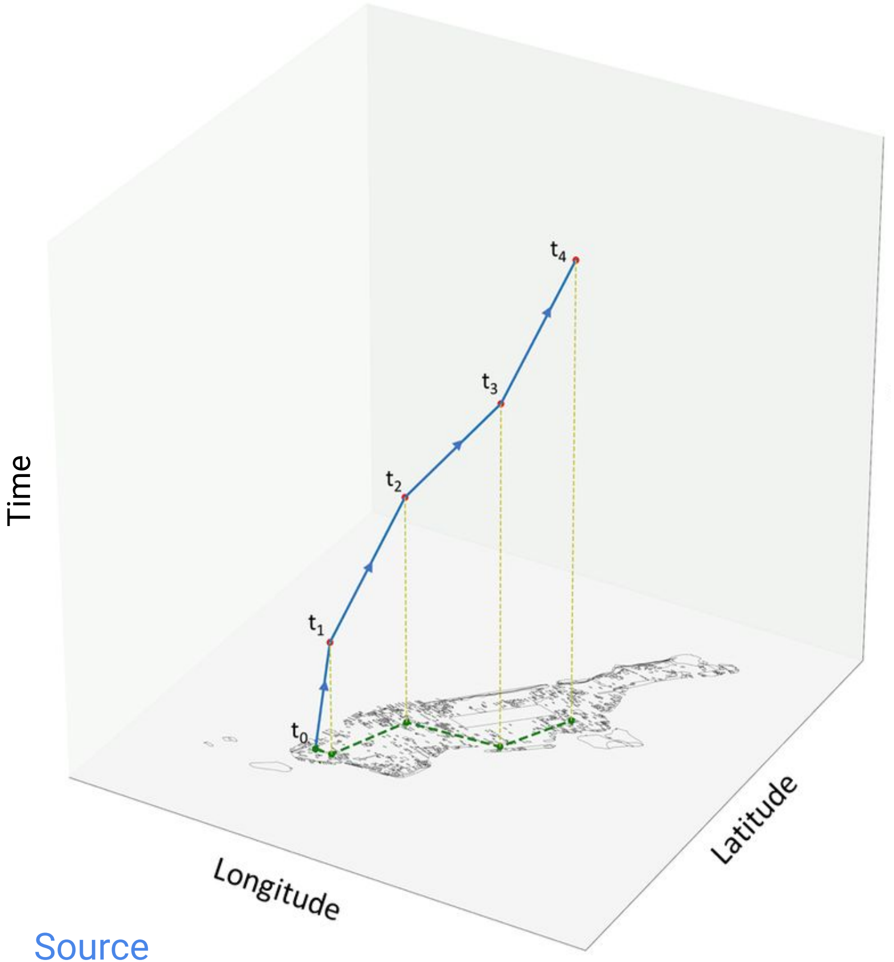
Polygon-arc list

Polygon	arc
101	1, 4, 6
102	4, 2, 5, 0, 7
103	6, 5, 3
104	7



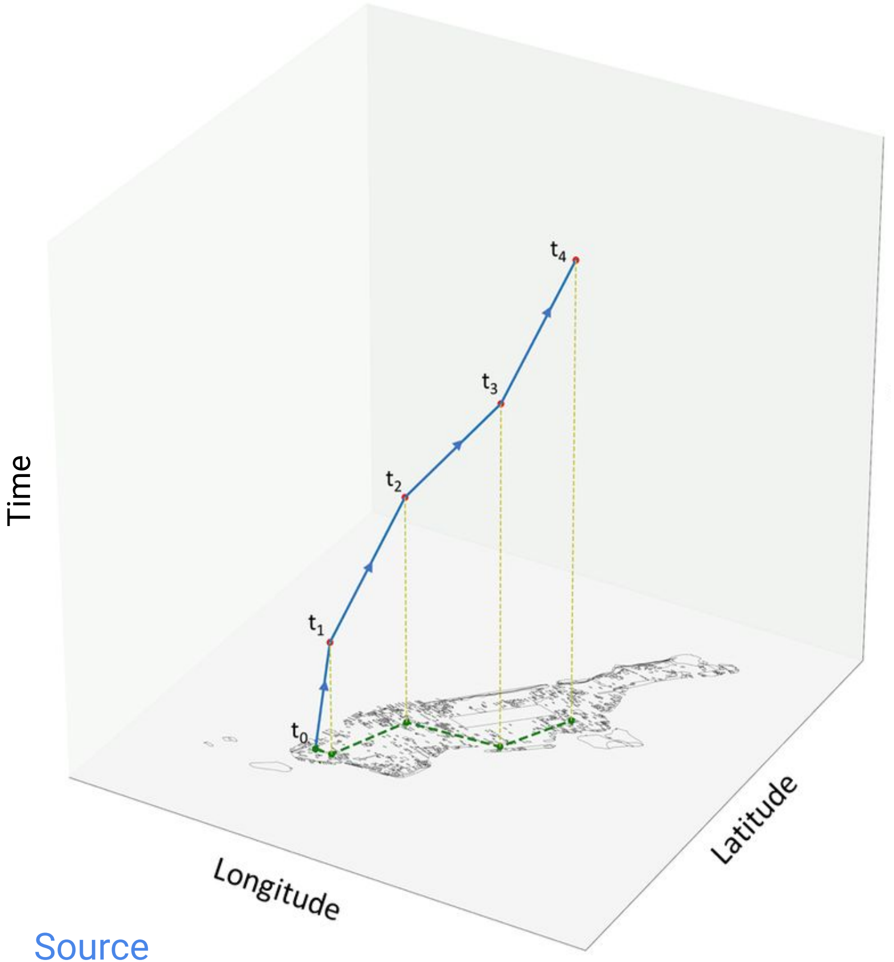
Trajectory

sequence of points that describe people’ movements



Source

What is a trajectory?



Source

Definitions

Trajectory

Let u be an individual, a **trajectory**

$$T_u = \langle p_1, p_2, \ldots, p_{nu} \rangle$$

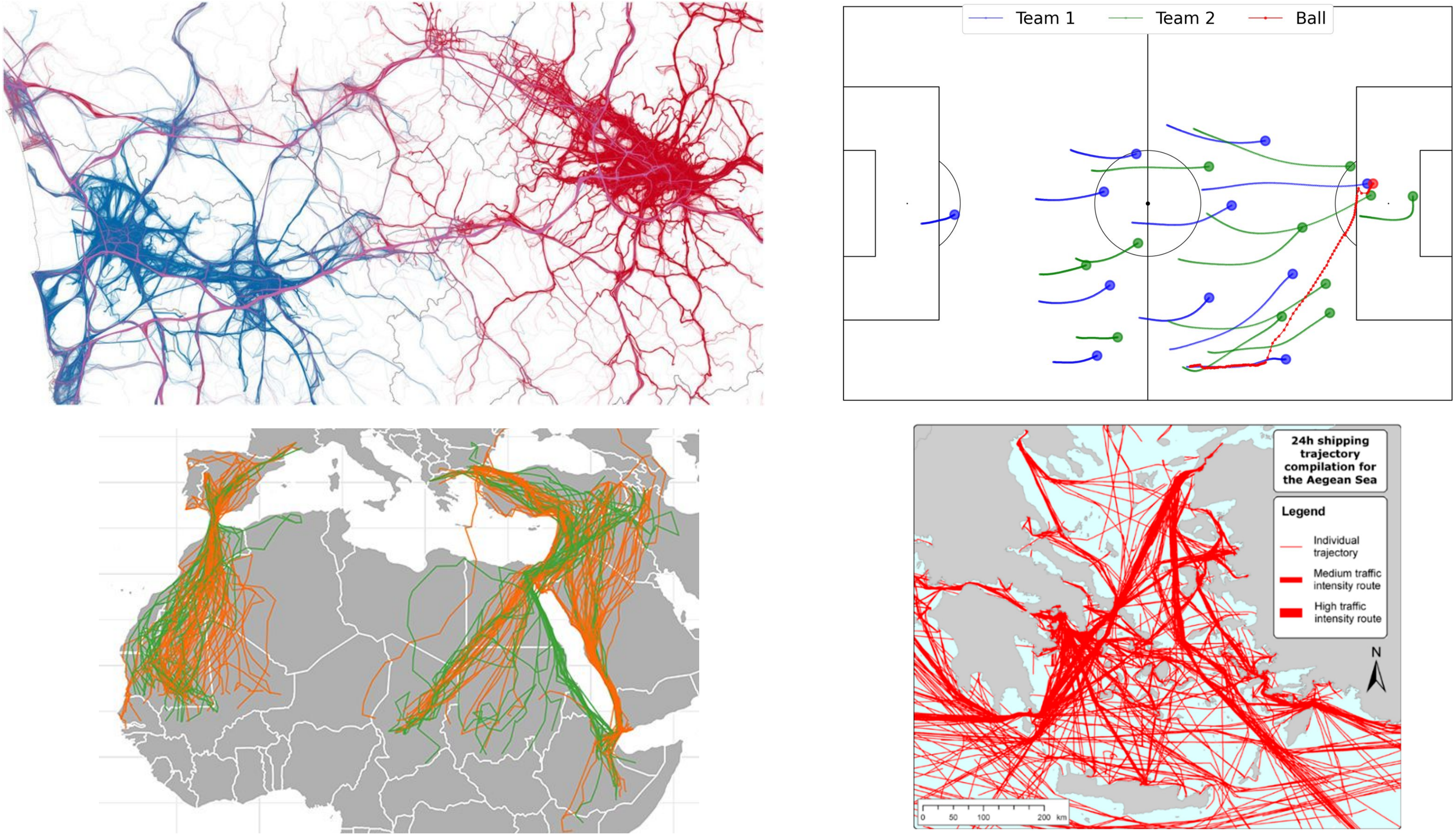
is a *time-ordered sequence* composed by the spatio-temporal points u visited.

A spatio-temporal point is a pair

$$p = (t, l)$$

where:

- t is the time
- $l = (x, y)$ is the point visited
  - x and y are spatial coords

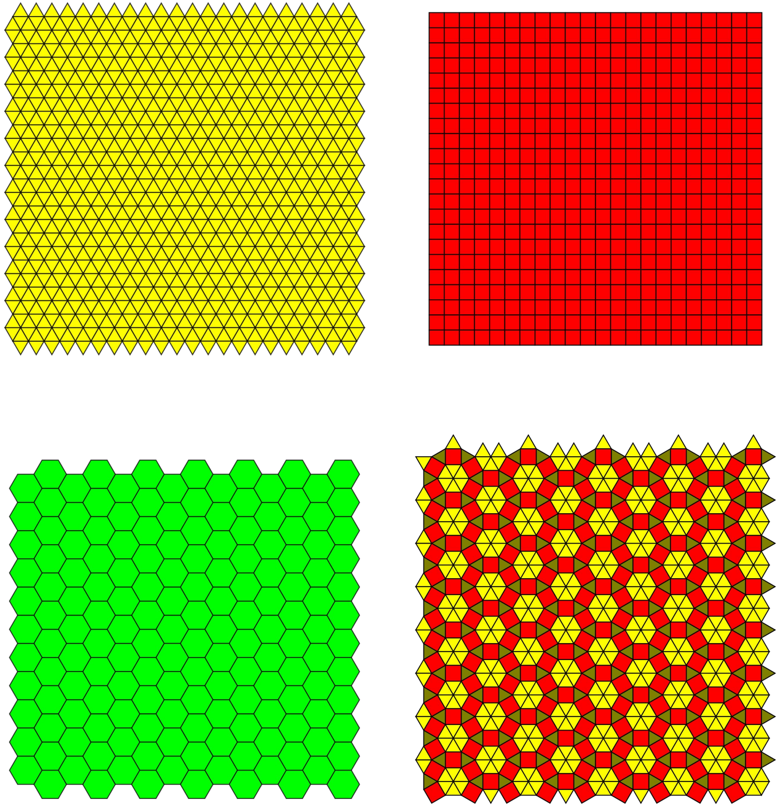


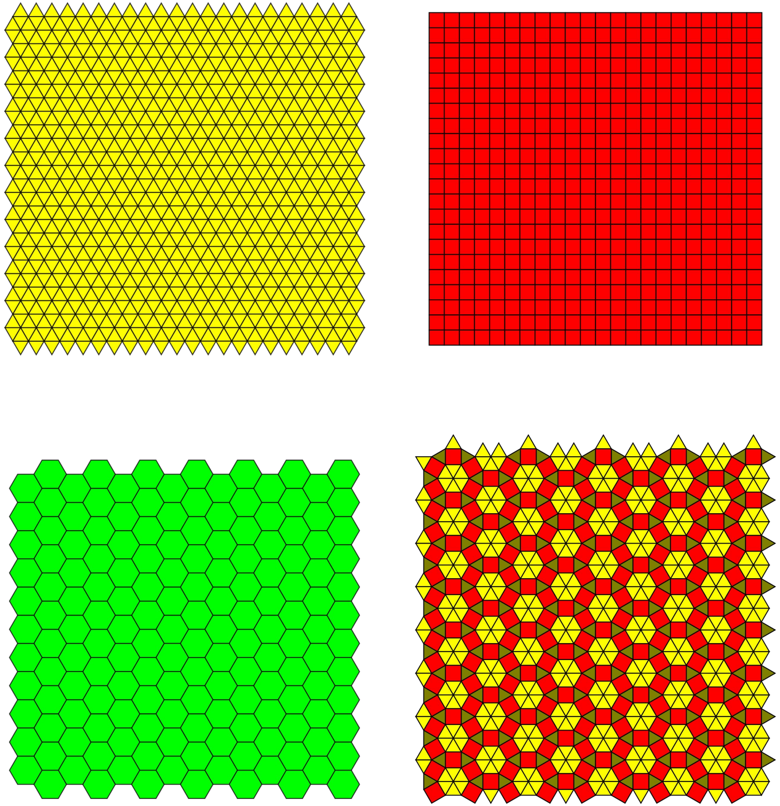
Team 1
Team 2
Ball
24h shipping trajectory compilation for the Aegean Sea
Legend
Individual trajectory
Medium traffic intensity route
High traffic intensity route
N
0
50
100
200 km

Spatial tessellation

division of the space into non-overlapping tiles

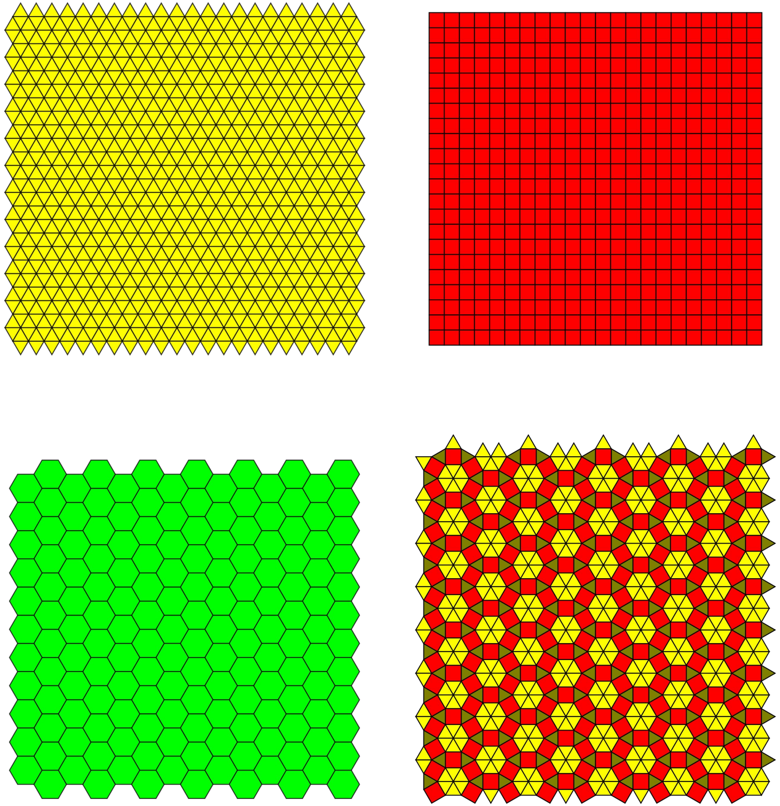
Definition **Spatial tessellation**





Definition Spatial tessellation

Given an area A, a tessellation is a set of geographical polygons with the following properties:

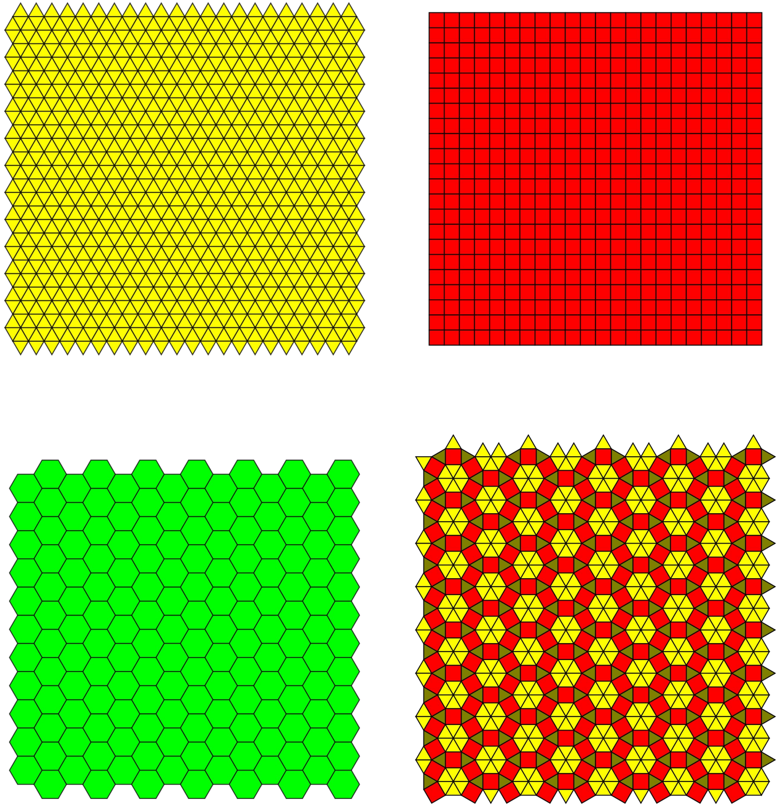


Definition Spatial tessellation

Given an area A, a tessellation is a set of geographical polygons with the following properties:

1. it contains a finite number of polygons, called tiles

$$\mathcal{G} = \{g_i : i = 1, \ldots, n\}$$



Definition **Spatial tessellation**

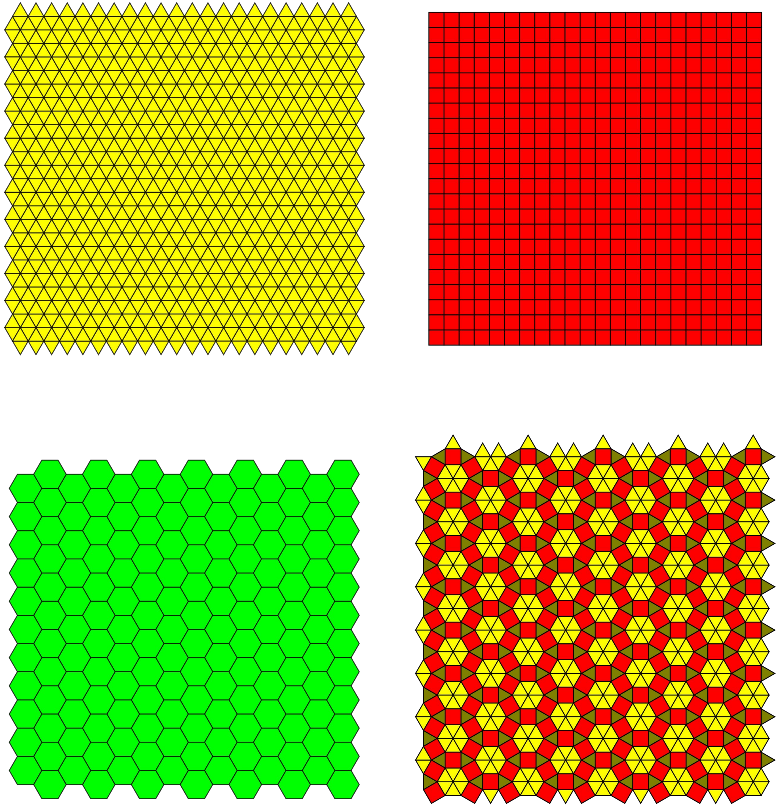
Given an area A, a tessellation is a set of geographical polygons with the following properties:

1. it contains a finite number of polygons, called tiles

$$\mathcal{G} = \{g_i : i = 1, \ldots, n\}$$

2. the tiles are non-overlapping

$$g_i \cap g_j = \emptyset, \forall i \neq j$$



Definition **Spatial tessellation**

Given an area A, a tessellation is a set of geographical polygons with the following properties:

1. it contains a finite number of polygons, called tiles

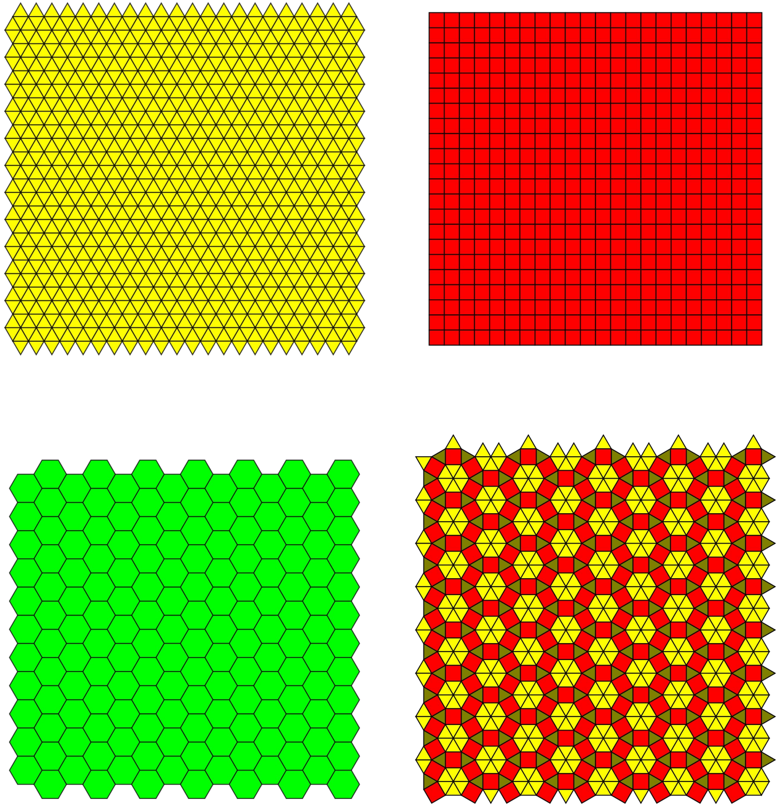
$$\mathcal{G} = \{g_i : i = 1, \ldots, n\}$$

2. the tiles are non-overlapping

$$g_i \cap g_j = \emptyset, \forall i \neq j$$

3. the union of all tiles completely covers the tessellation

$$\bigcup_{i=1}^{n} g_i = A$$



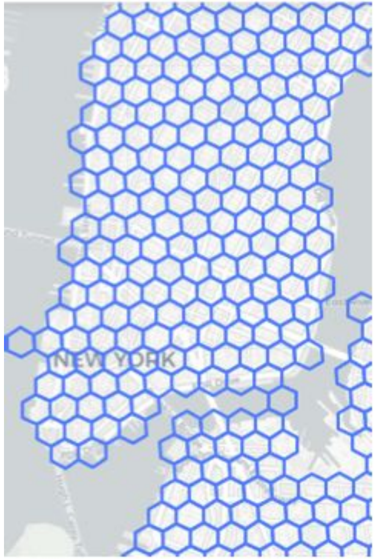
A tessellation may be:

- **Regular**
  - equilateral triangular, squared, quadrilateral, hexagonal
- **Irregular**
  - buildings, census cells, administrative units

- A spatial join can be used to associate a point with the tile that contains it

- Since the tessellation has no overlapping tiles and no gaps, each point is assigned only to one tile

Uber H3

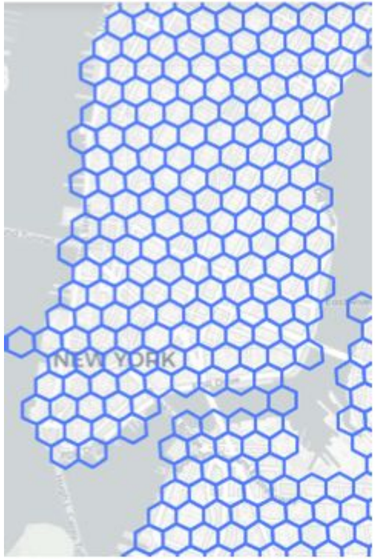


Administrative

Squared

S2 Geometry

Uber H3



- Uber H3 recursively creates increasingly higher precision hexagon grids

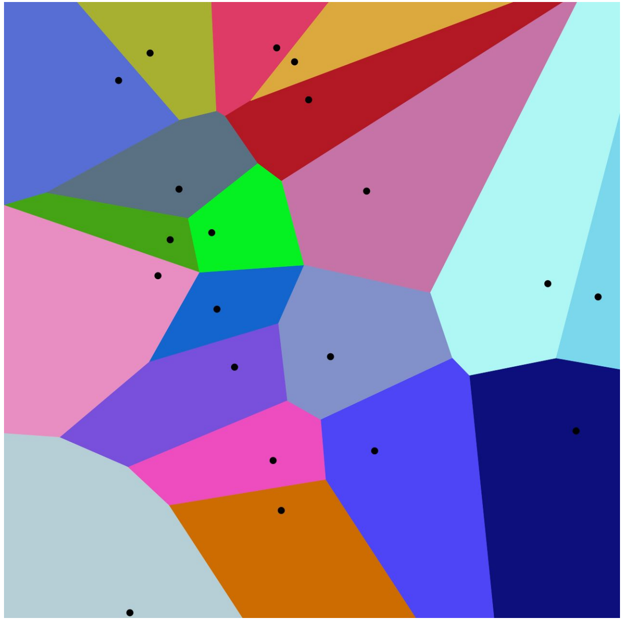
- Each subsequent resolution is created splitting each cell into seven children recursively

- 16 grid resolutions:
  - resolution 0: 122 base cells
  - resolution 15: cells with an area of less than 1 m^2

S2 Geometry

- S2 Geometry decomposes the unit sphere into a hierarchy of cells (quadrilateral bounded by four geodesics)
- The top level projects the six faces of a cube into the unit sphere
- Lower levels subdivides each cell into four sub-cells recursively
- Each cell in the hierarchy has a level, defined as the number of times the cell has been subdivided (starting with a face cell).
- Cells' levels range from 0 to 30. The smallest cells at level 30 are called leaf cells; there are 6×4^{30} cells in total, each about 1cm across on the Earth's surface

VORONOI TESSELLATIONS



- A partition of a plane into regions close to each of a given set of objects

- For object there is a corresponding region, called a Voronoi cell, consisting of all points of the plane closer to that seed than to any other.

INTERVALLO

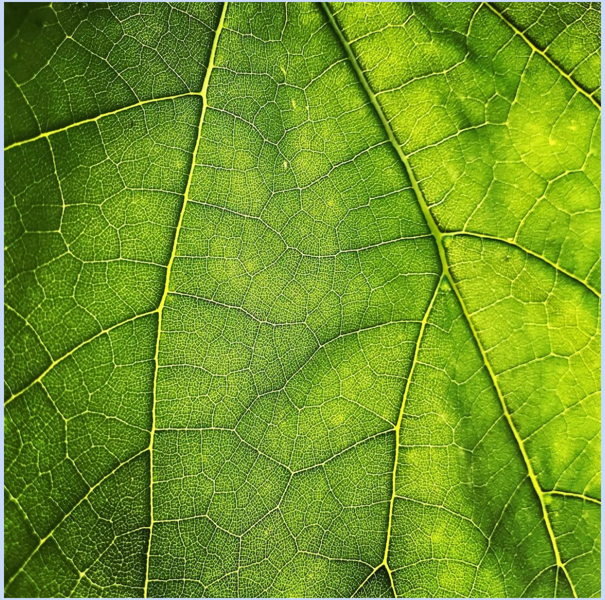
Tessellations in natural systems

INTERVALLO

Tessellations in natural systems

INTERVALLO

Tessellations in natural systems

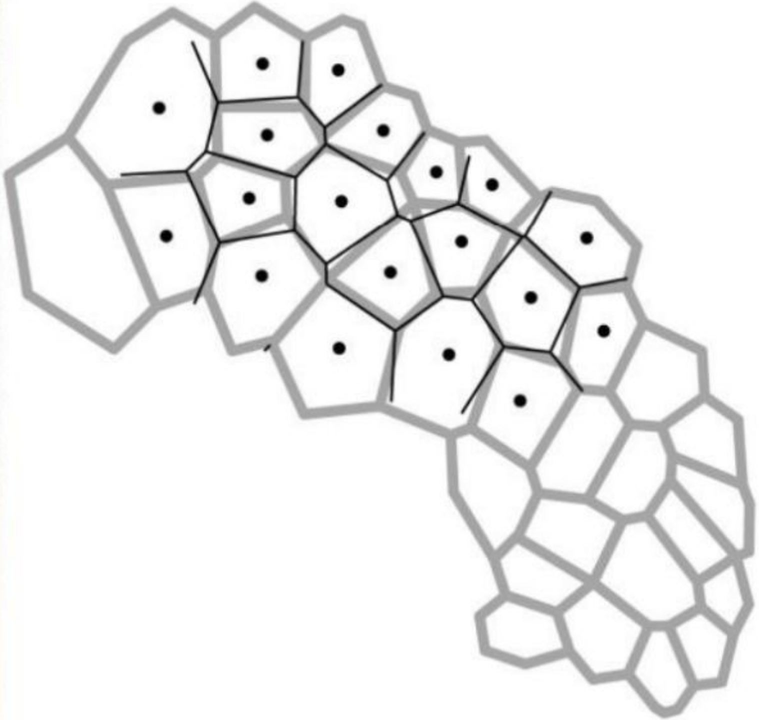


INTERVALLO

Tessellations in artificial systems

Giant’s causeway and foot ball

Real-world tessellations



References

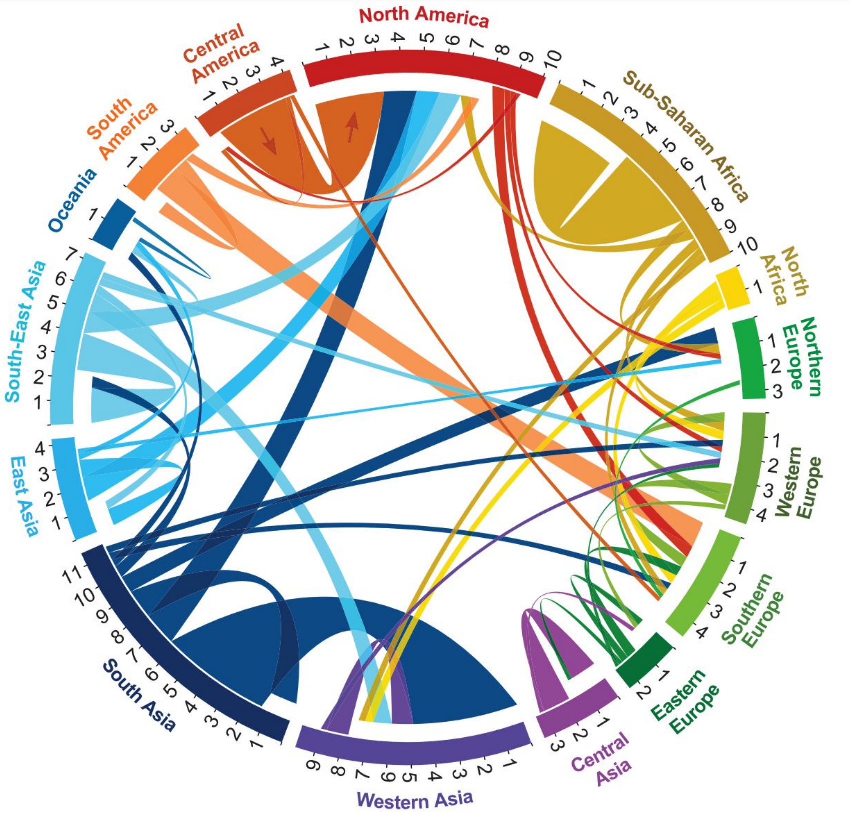
[article] Voronoi Tessellations and Scutoids Are Everywhere, Scientific American, 2019

Flows

movements of groups of people between places

Flows

Definition



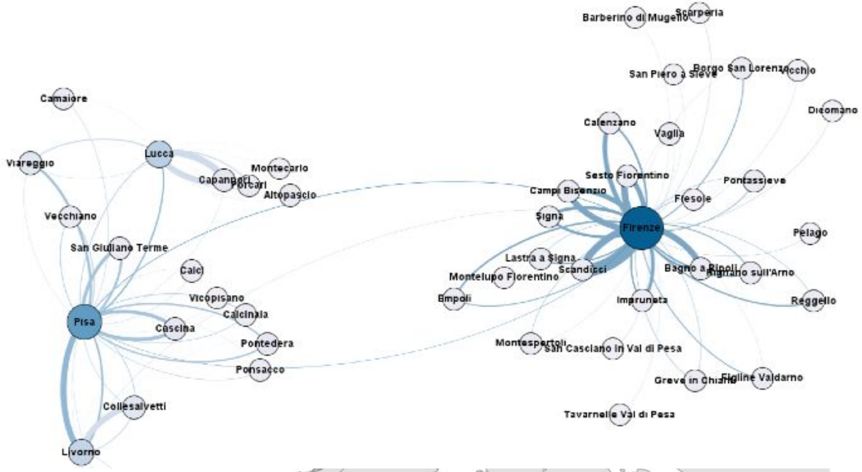
Given a tessellation, the flow:

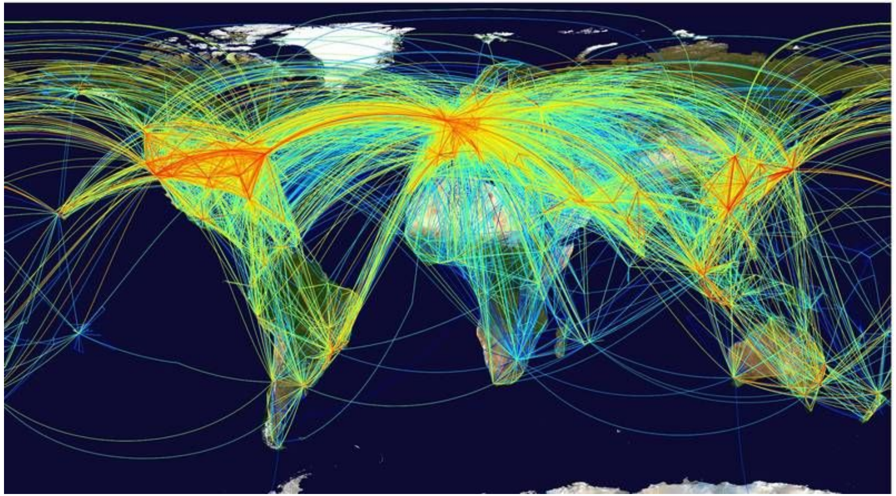
$$y(g_i, g_j)$$

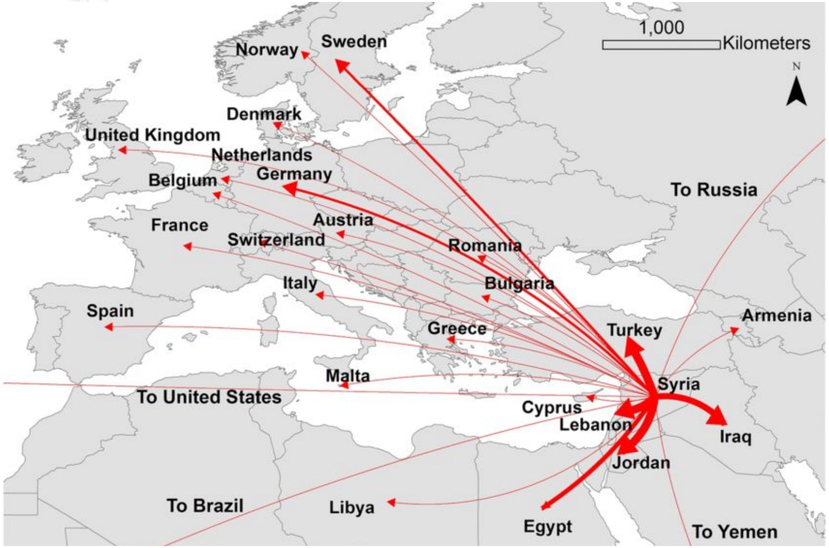
represents the number of people (or in general objects) moving between g_i and g_j.

Trajectory vs Flows:

- A trajectory corresponds to a single individual
- A flow corresponds to the total amount of people that move between two points
- Flows *can be* obtained from a set of trajectories
- Trajectories *cannot* be obtained from flows


Barberino di Mugello
Scarperia
San Piero a Sieve
Vicchio
Dicomano
Camaiore
Calenzano
Vaglia
Viareggio
Lucca
Montecarlo
Altopascio
Sesto Fiorentino
Pontassieve
Campi Bisenzio
Fiesole
Vecchiano
Signa
Firenze
Pelago
San Giuliano Terme
Lastra a Signa
Calci
Montelupo Fiorentino
Scandicci
Empoli
Vicopisano
Impruneta
Reggello
Calcinaia
Pisa
Cascina
Pontedera
Ponsacco
Greve in Chianti
Tavarnelle Val di Pesa
Collesalvetti
Livorno




1,000
Kilometers
N
Norway
Sweden
Denmark
United Kingdom
Netherlands
Belgium
Germany
France
Austria
Switzerland
Romania
Bulgaria
To Russia
Italy
Spain
Greece
Turkey
Armenia
Malta
Syria
To United States
Cyprus
Lebanon
Iraq
Jordan
To Brazil
Libya
Egypt
To Yemen

Homework 2.1

Compute the distance from your home to the five largest capitals in the EU, using both the Earth distance and the Euclidean distance

- https://www.itilog.com/ provides Lat, Long of an address
- Create a bar chart with the absolute difference between the Earth and Euclidean distances for each capital
- Are the Earth and Euclidean distances coherent? Make a plot to show that

- Submit a (well-commented) python notebook

Homework 2.2

What is the most "central" EU capital, i.e., the one with the lowest average Earth distance with the other EU capitals?

- Create a bar chart with the average distance for each EU capital, sorted in increasing order
- Repeat the exercise for at least another continent

- Submit a (well-documented) python notebook

Homework 2.3

Find at least three examples of regular or irregular tessellations in natural or artificial ecosystems (other than those presented in class). Discuss the reason (if any) why the tiles have precisely that shape.

- Write a blog post (2-3 pages max.) about it
  - Include the references you used (to papers, blog posts, newspaper articles, videos, or whatever)

Homework 2.4

Find objects (other than humans, animals, and transportation means) that may be described by trajectories. Do the same for flows.

Could clouds be represented as trajectories?

- Write a blog post (2-3 pages max.) about it!
  - Include references (to papers, blog posts, newspaper articles, videos, or whatever)

Homework 2.5

Create your own trajectory! Track your movements for an entire week, tracking all points of interest you visited (e.g., home, friends' home, university, supermarket, gym, bars).

- Use https://www.itilog.com/ to detect Lat & Long of places based on their address.
- Create a `TrajDataFrame` in scikit-mobility and visualize it.
    - If you want, add to `TrajDataFrame` the place's name or address
- Submit the notebook and the data, or a link to them

Homework 2.6

Use scikit-mobility to create a squared tessellation over Florence (500m). For any pair of tiles, generate a random flow choosing a number uniformly at random in [0, 1000]. Repeat for h3 tessellation (500m) and a Voronoi tessellation (with n=100 random points).

- Create a `FlowDataFrame` and visualize it in skmob.
- Are the flows realistic? Why? Comment on it.
- Save the tessellations into a shapefile using GeoPandas

- Submit a (well-commented) notebook.

Homework 2.7

What is the theoretic upper bound of the mapping error of a trajectory of n points into a squared tessellation of size s meters? And what is this upper bound for a hexagonal or triangular tessellation?

- Use a real trajectory dataset and map the trajectory into a tessellation. Compute the error. Is it actually lower than the theoretic upper bound?

- Submit a (well-documented) notebook

Homework 2.8

Compute the area of Italian subdivisions using GeoPandas and plot their area as a bar chart

- Download a shapefile that describe Italian regions (e.g., here)
- Make a bar chart, put the regions in increasing order of area (put the region's name on the x axis)
- Repeat for provinces and municipalities (plot only the 100 municipalities with the highest area)
- Plot the shape of each region (in blue), with the shape of its capital municipality (in red)

- Submit a (well-documented) notebook

Homework 2.9

Create and plot a `GeoDataFrame` with the top 1% and the bottom 1% municipalities in Italy based on their area

- Download a shapefile that describe Italian regions (e.g., here)
- Create a `GeoDataFrame` with two rows (top 1% and bottom 1%) and the corresponding multipolygons
- Plot the multipolygons with folium

- Submit a (well-documented) notebook

Homework 2.10

Create a `GeometryCollection` *with Shapely regarding an imaginary squared Island with a lake, a house, and a road.*

- Island: a square centered at (0, 0) with side 10
- Lake: rectangle base 3, height 2, bottom-left corner at (-2.5, 0)
- House: `Point` centered at (3, 2) with size (buffer) 1
- Road: line connecting the lake's center and the house's center
- Plot the collection using folium

- Submit a (well-documented) notebook

Homework 2.11

Several "Los Pollos Hermanos" vans, carrying large quantities of methamphetamine (meth), were attacked by a drug cartel 10 times in an area in New Mexico. The DEA thinks the meth lab is at the centroid of this area.

- Compute the smallest polygon that contains all the points corresponding to the attacks
- Create a `GeometryCollection` that contains New Mexico, the polygon, and the points within it
- Visualize the collection and the centroid in folium (use markers for points, color the centroid differently)
- Randomly generate the attacks' points in New Mexico
- Submit a (well-commented) notebook

Homework 2.12

Download the geojson files describing the world of "A Game of Thrones" from this repository. Plot the places mentioned in three chapters of your choice.

- Create some `GeoDataFrames` to store info and geometries of continents, islands, lakes, rivers, roads, the Wall, and the locations.
- Plot them using geopandas
- Select 3 chapters from here, and visualize all places mentioned in the chapters that are present in the locations `GeoDataFrame`.
- Plot each chapter's points differently and make a legend with the Chapter number and title.
- Submit a (well-commented) notebook

Homework 2.13

Compute the probability of a randomly generated lat/long pair to fall on the land.

- Generate n=10k points randomly
- Select only the points that fall on the land and compute the probability of that a random point falls into the land.
- Repeat this experiment 100 times and make a boxplot showing the distribution of probabilities.
- What is the mean and the std of these probabilities?
- Does the average probability decrease if you execute 1000 experiments?

- Submit a (well-commented) notebook

Homework 2.14

Download parking meters (parchimetri) *in Rome and the shapes of the* "municipalità" of Rome. *Make a cloropleth math in folium describing how many parking meters each municipalità has.*

- Do the same for the accidents in Rome. Make another folium cloropleth map.
- Submit a (well-commented) notebook

Homework 2.15

To plan a reduction of the environmental impact, UEFA aims to compute the total distance traveled by all clubs in the Champions League during their matches in the first round of the competition.

- Take the position of the city of each club qualified to the Champions League (season 2022/2023)
- Compute the sum of the distance traveled by each club to play the matches in its group
- Make a bar chart to show the total distance traveled by each club
- What's the club the travel the most? And the club traveling the least?
- Compute and visualize the mobility flows generated by the first round (each club traveling counts as 1)

- Submit a (well-commented) notebook and the data used.

to study for the exam

Material

- [book chapter] Introduction to geographic information systems, Kang-Tsung Chang, McGraw-Hill
  - Chapter 3: Vector Data model

- [lesson] Automating GIS-processes, Department of Geosciences and Geography, University of Helsinki, Finland
  - Lesson 1, Shapely and geometric objects

- [book chapter] Essentials of Geographic Information Systems, Saylor Academy
  - Chapter 4, Section 4.2: Vector Data Models

- [article] Analyze Geospatial Data in Python: GeoPandas and Shapely, learndatasci.com

to study for the exam

Material

- [book] Introduction to geographic information systems, Kang-Tsung Chang, McGraw-Hill
  - Chapter 2 (Section 2.1)

- [paper] A survey of deep learning for human mobility, Pappalardo et al., ACM Computing Surveys, 2021
  - Section 2.1, Appendix A

- [paper] scikit-mobility: a Python library for the Analysis, Generation, and Risk Assessment of Mobility Data, Pappalardo et al., Journal of Statistical Software
  - Sections 1, 2

to study for the exam

Material

- [video] Intro to coordinate systems and UTM projection, Middlebury Remote Sensing, 2016