DEPARTMENT OF TRANSPORT TECHNOLOGY
AND TRANSPORT ECONOMICS

Intelligent Transportation Systems
Lab course

Introduction to QGIS software

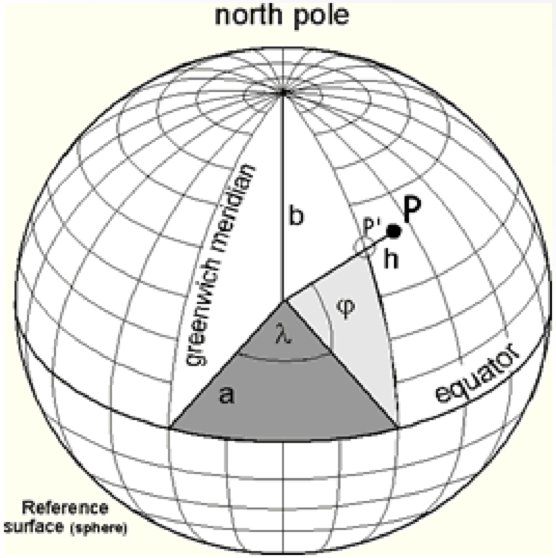
Tamás Soltész

Basics of cartography and Gis

History

- Shape of the Earth: nearly spherical, which was known (or assumed) even in antique times
  - Philosophical consideration (perfection)
  - Experience: shadow of the Earth on the Moon; later sight of ship poles on the see
  - Route of Phoenician voyagers around Africa (~500 BC)
  - Calculation of the radius by Eratosthenes, with 2-15% error (~250 BC)
  - Calculation of Persian Al-Biuni had only 0,2% error (~1000 AD)

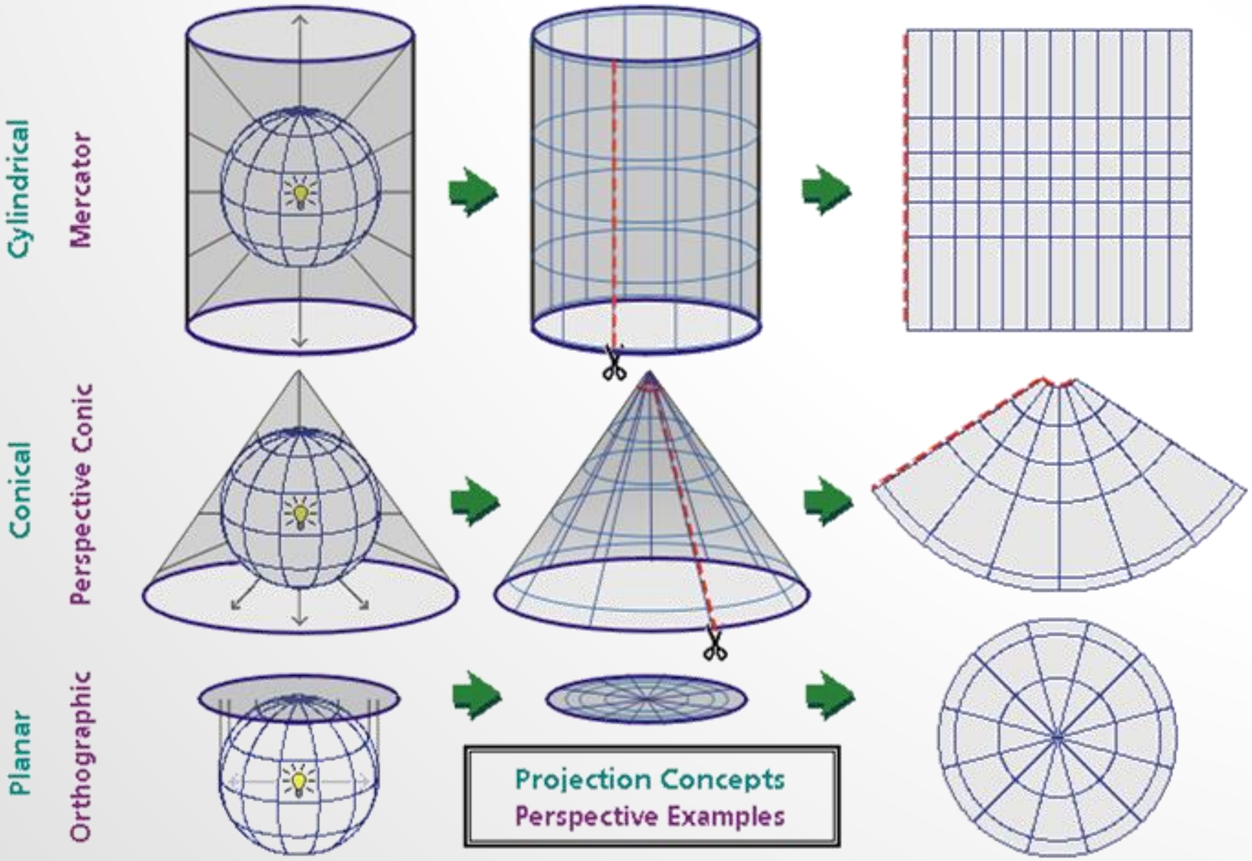
Basics of Cartography I



- Coordinate system:
  spherical coordinates
  - Latitudes:
    - angle of the radius of the position and the plane of the Equator
    - each one is (approximately) a full circle, with different size (from 40 000 km to 0)
    - distance between them is constant
      1' = 1852 m = 1 nautical mile; 1° = 111 km
  - Longitudes:
    - angle of meridian of the position and a starting meridian
    - each one is (approximately) a half circle, with the same length 20 000 km (first definition of meter)
    - distance between them is different (from 111 km to 0)

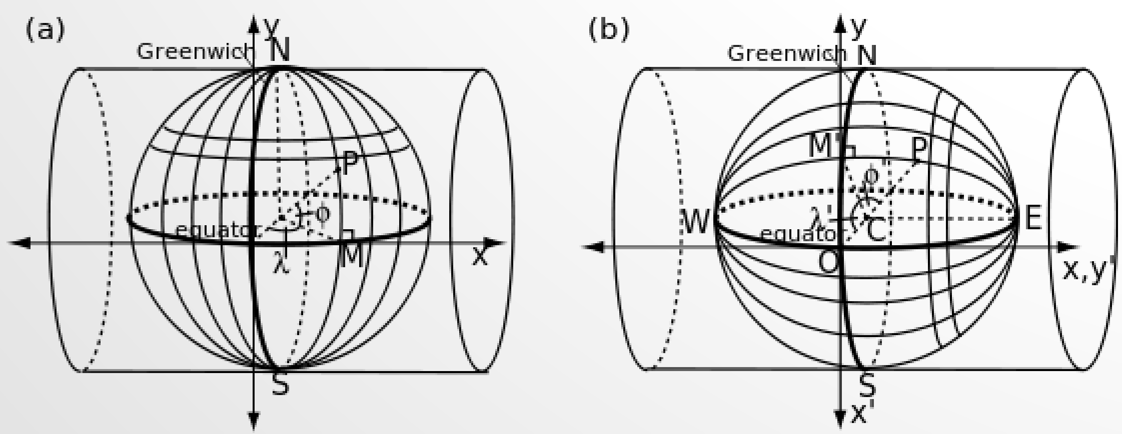
Basics of Cartography II

- Representation of a spherical surface in plane: by projections



Basics of Cartography III

- Projection types (main aims):
  - Equidistant
  - Conformal (equal angles)
  - Equal area
  - **Compromise**
- Traversed Mercator Projection



Geographical Information Systems – GIS

- Simplified definition:
  - Information systems which contain – and can work with – spatial (geographical) data
  - They can be seen as digital maps (like Google Maps etc.)
- Application area:
  - registries (estates, utilities etc.)
  - cartographic systems
  - environmental and meteorological systems
  - transport information systems

GIS DATA TYPES

- GIS data are organized in **layers** which contain elements from the same type (e.g. schools, roads, green areas etc.)
- Layers have two main categories:
  - **Raster** layers can be considered as map images – each pixel represent a small area of the map
  - **Vector** layers are based on points and polygons – each object is represented by its (approximate) geometry and position
- Very important property of each layer is their **projection** (defines their coordinate system)

GIS vector layer types

- **Point** layers:
  - spatial data of points only consist of 2 (or 3) coordinates
  - examples: POIs (like shops, cafés), PT stops etc.
- **Polyline** (or linestring) layers:
  - all lines are represented by their end- and intermediate points (section between points are straight)
  - examples: rivers, borders, roads, railways etc.
- **Polygon** layers:
  - areas are represented by their polygon border(s)
  - examples: lakes, parks, industrial areas etc.

QGIS software – Basics

QGIS SOFTWARE

- (Previously called Quantum GIS)
- Open-source software for managing and visualizing GIS data
- Plenty of sophisticated GIS functions and many plugins are available
- Self-made scripts can be added
- However, some errors and bugs may appear in rarely used functions
- We work with version **3**.xx

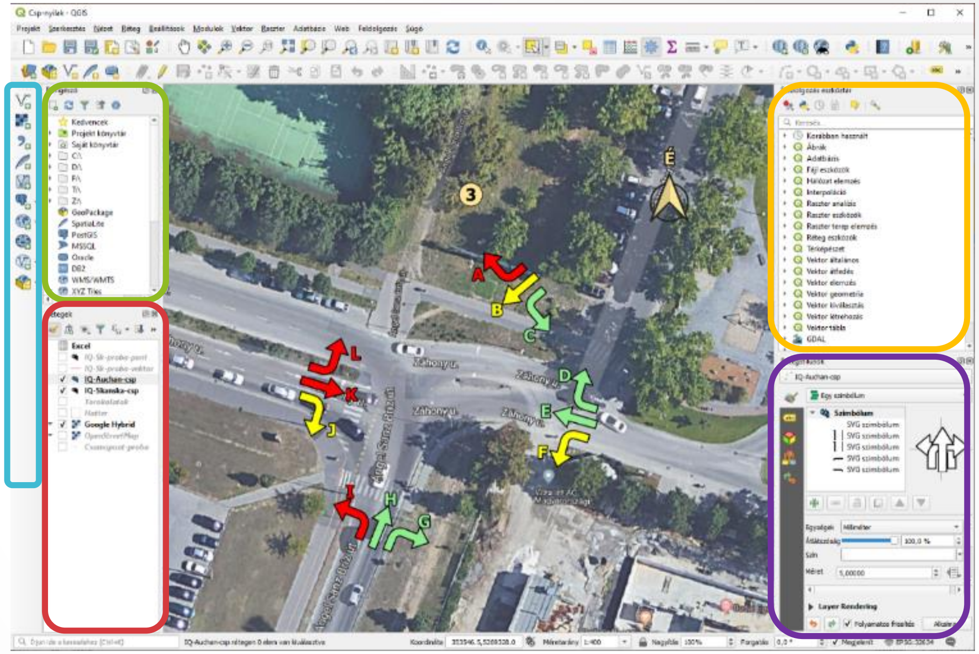
File handling in QGIS

- Maps can be managed in a Project, but layers are stored **separately**
  - to open a map, all project and layer files are needed
- Layer files store:
  - spatial (and other) data of each element of the layer
  - separate file or files (!) for each layer
- Project file stores:
  - general project data (e.g. scale, visualization settings)
  - file paths for layer files (absolute / relative)
  - style settings for layers

QGIS USER INTERFACE I

- Change language:
  - form Menu: Settings → Options… → General tab, Override system locale ☑
  - *in Hungarian: Beállítások → Beállítások… → Általános tab, Helyi beállítás felülbírálása* ☑
  - choose from User Interface Translation list
- Enable hidden functions:
  - from Menu: Plugins → Manage and Install Plugins…, filter to Installed, then ☑ the needed ones
  - check whether Processing module is enabled
  - from this window, additional plugins can be installed (change filter)

QGIS USER INTERFACE II



- Usual layout:
  - Top: Menu line, toolbars
  - Left: *Browser panel* – for opening files

    Mange layers toolbar *(if enabled)* – to add layers

    Layers panel – to handle layer (order, visibility etc.)
  - Right: *'Toolbar' / Processing Toolbox* panel *(if enabled)*
  - Bottom: Status bar (cursor position, scale, projection)
- Additional panels can be added in View menu
  - e.g. *Layer Styling* panel is useful

Layers in QGIS

ADDING AND STYLING LAYERS

- Import vector data
  - ➔ Add Vector Layer / Open Data Source Manager
- Styling a whole layer *(1)* or elements by attributes *(2)*
  - ➔ Layer Properties → Style
    → Single symbol *(1)* / Categorized *(2)*
- Import text (csv) data
  - ➔ Add Delimited Text Layer
  - ➔ pay attention to settings: Encoding, Format (delimiters), Header, Geometry
  - ➔ give/check layer's coordinate system (CRS)

Handling layers

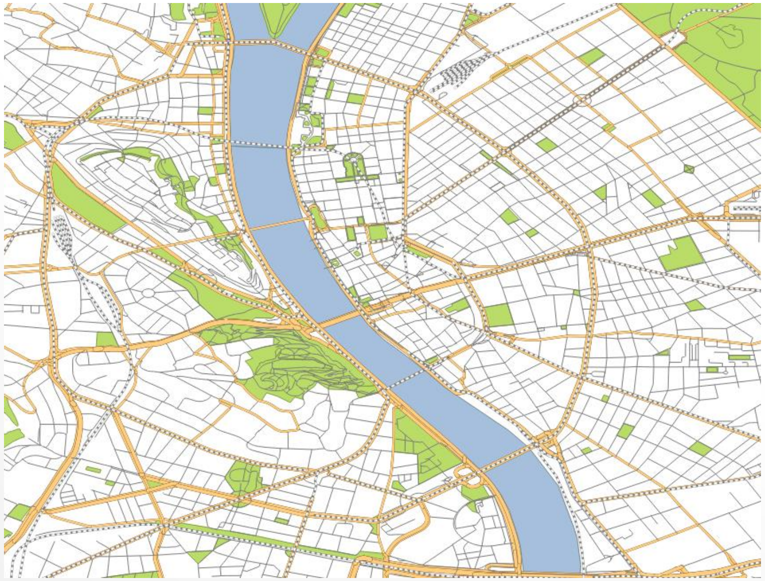
- Useful Interface functions:
  - ➔ Zoom to Selection / Layer,
  - ➔ Select Features by area or single click / expression
  - ➔ Deselect all
- Attribute table:
  - stores additional data of features (e.g. name, code)
  - does not show geometry
- Editing: Edit mode has to be on (pencil icon)
- Decomposing: Select (filter) features, Copy to clipboard and Paste as new layer

CRS (Coordinate Reference System) of layers

- CRS defines:
  - projection (shape and viewpoint)
  - the reference shape of Earth (nominal sea level)
  - coordinate system (origin and unit)
- Layer CRS: the system in which geometry is **stored in layer files**
- Project CRS: current 'view' of the map, layer CRS can differ from it
  (QGIS is capable of 'on the fly' transformation)
- Layer CRS can be changed only by saving it as a new layer

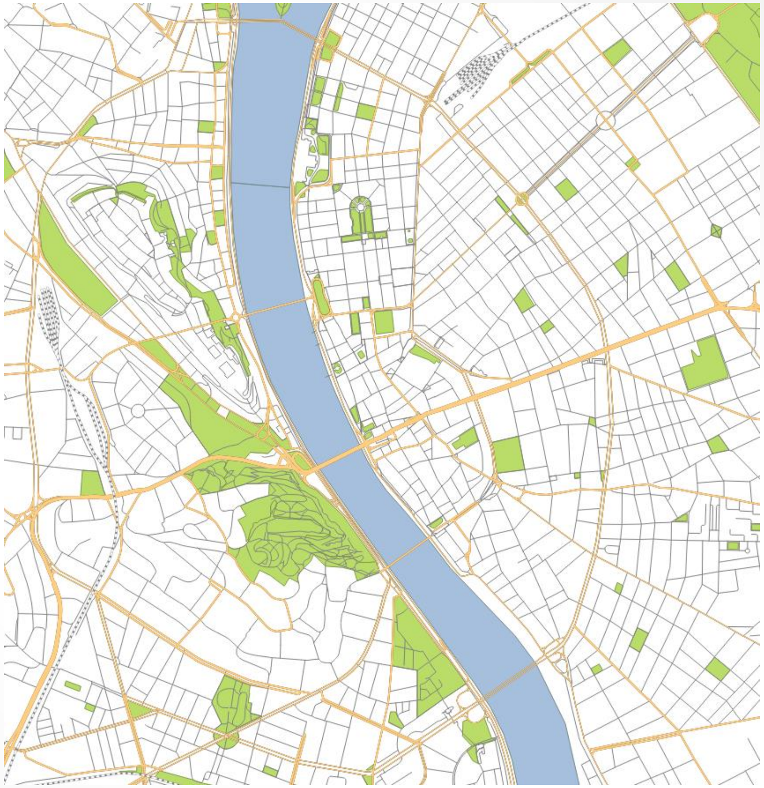
WGS 84 COORD. SYSTEM (EPSG:4326)

- Name from the reference shape of Earth (latest in use)
- Data: spherical coordinates in decimal degrees (most common format; e.g. 47.48201, 19.0580)
  → map unit is degree (not equidistant for longitudes)
- As default, QGIS use Mercator projection with it
  → objects in higher latitudes have considerable distortion



UTM COORDINATE SYSTEMS

- We use WGS 84 / UTM 34N (EPSG:32634)
- Traversed Mercator projection → low distortion (34N: central meridian code)
- Data: x, y coordinates in meter (from an origin) → distances can be calculated without transformation



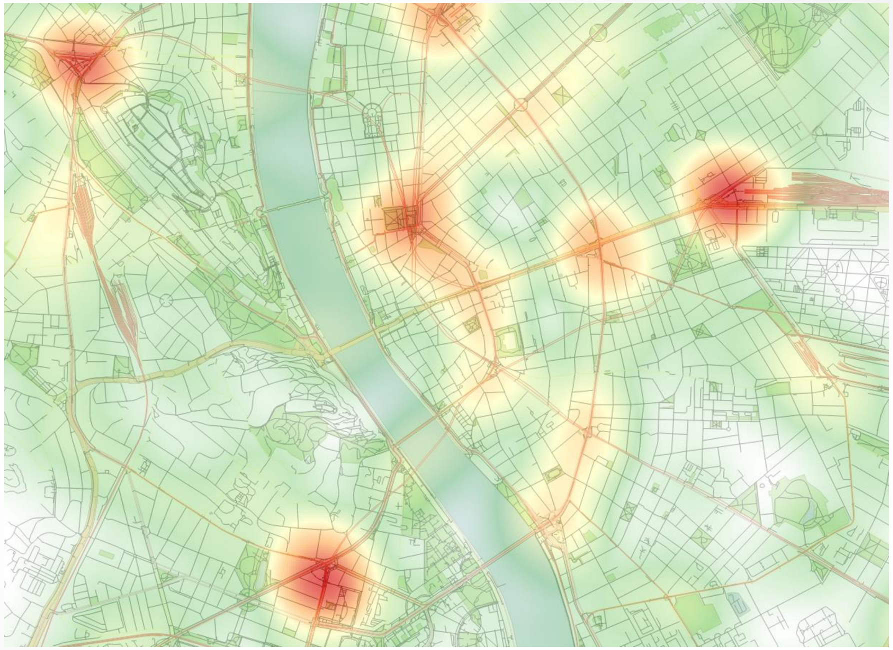
- EOV (EPSG:23700): coord. system optimized for the area of Hungary (cylindrical, but not trav. Mercator projection)

Introduction to QGIS functions

Useful data sources for GIS projects

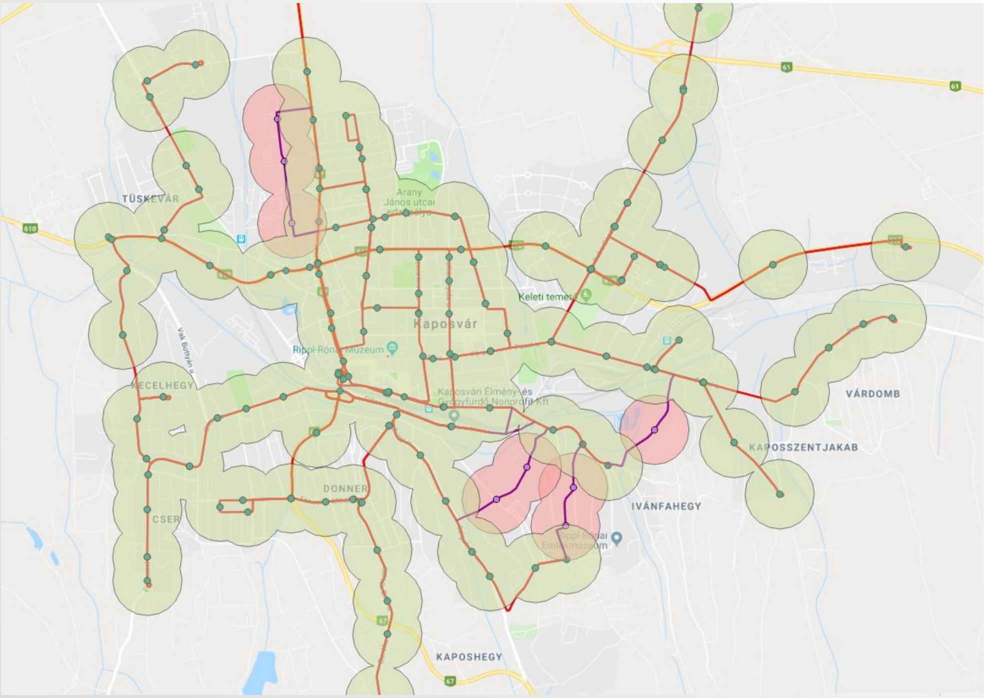
Calculations with points I

- Creating Heatmap
  - ➔ Layer Properties → Style → Heatmap
  - ➔ Set colour scheme
  - ➔ Set Transparency
  - ➔ Set size (Radius)
    Unit is important:
    - mm and pixel are measured on screen (like line widths) – result depends on map scale / zoom
    - Map unit is the project CRS unit, e.g. meter – real distance of objects; does not depend on scale
  - ➔ Set Weighting (importance of points)

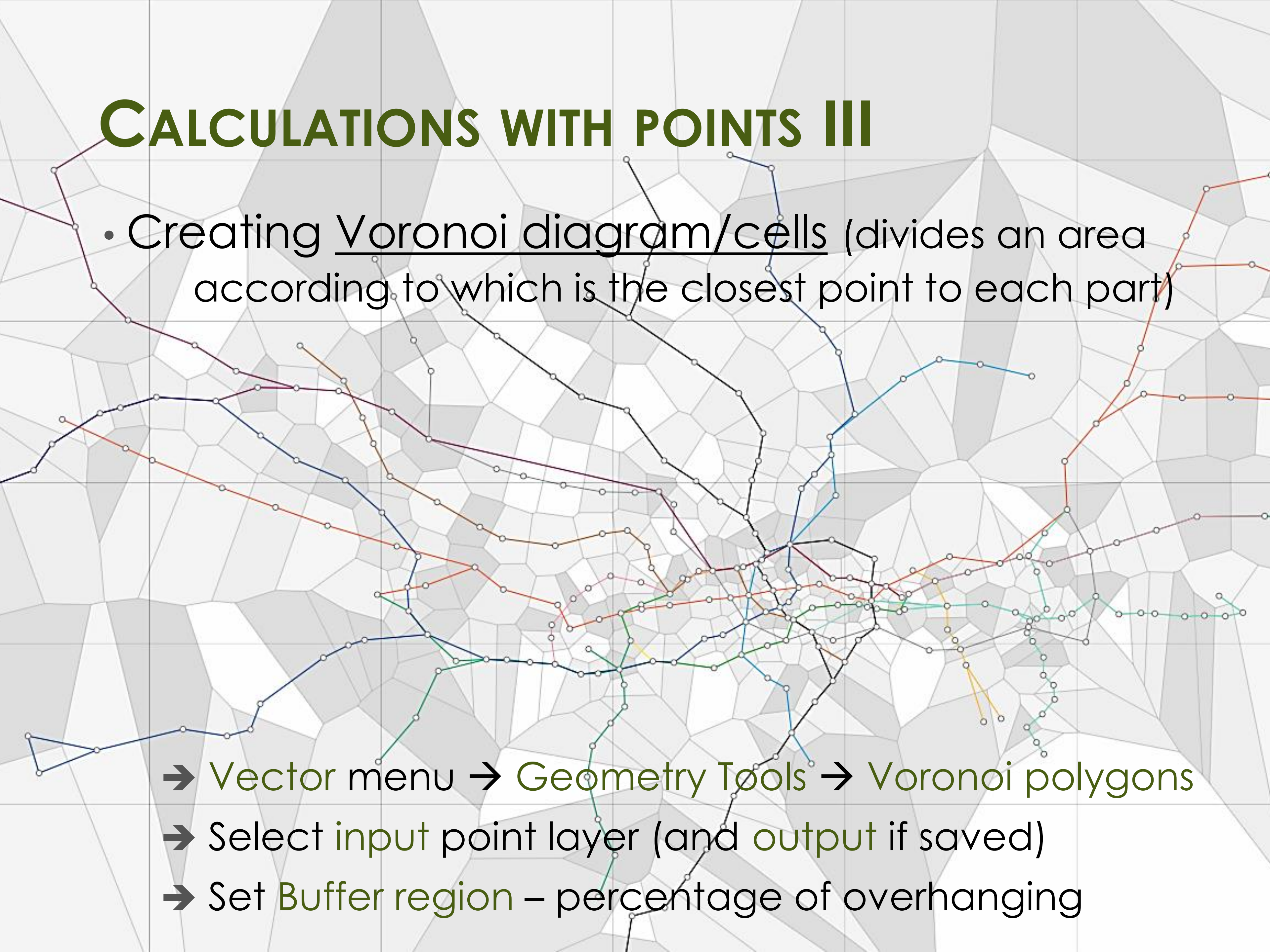


Calculations with points II

- Creating Coverage diagram
  - ➔ Processing Toolbox → Fixed distance buffer
  - ➔ Select point layer (as input)
  - ➔ Set pattern size (Distance) – unit is based on the layer's(!) CRS
  - ➔ Select output if saved (if not, it will be a temporary layer, which disappears after closing the project)



Calculations with points III



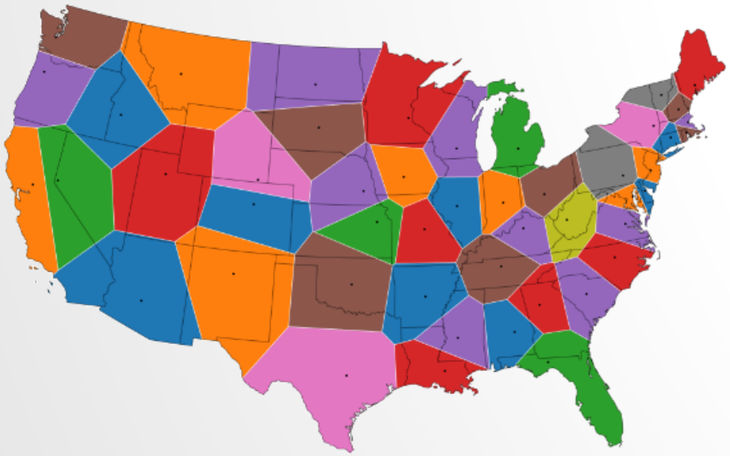
- Creating <u>Voronoi diagram/cells</u> (divides an area according to which is the closest point to each part)

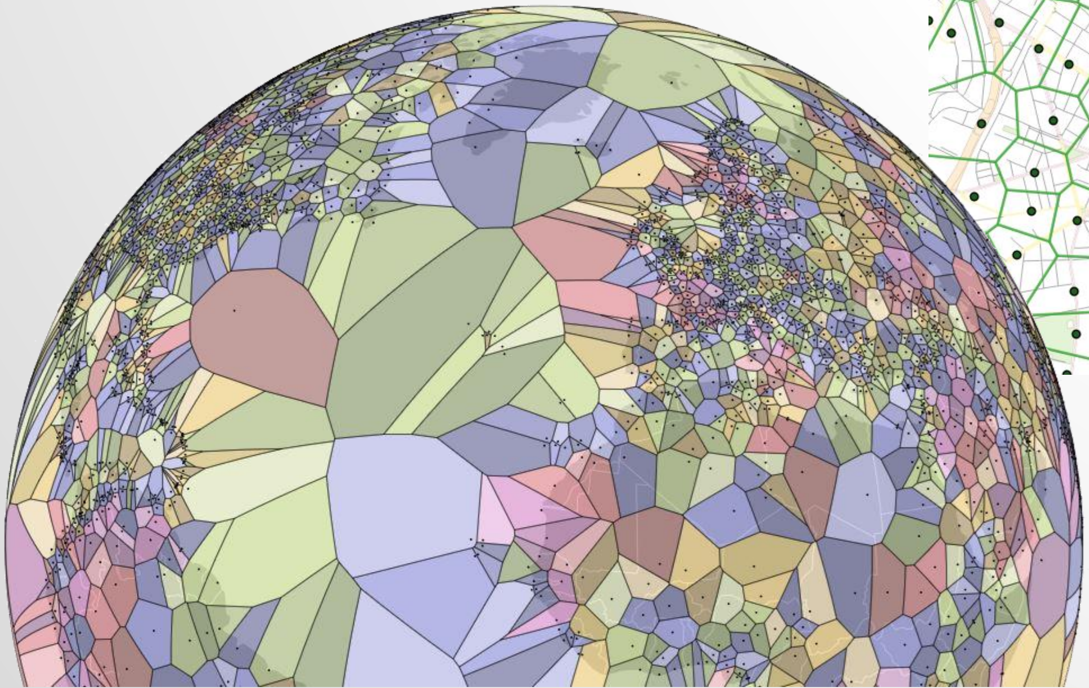
➔ Vector menu → Geometry Tools → Voronoi polygons

➔ Select input point layer (and output if saved)

➔ Set Buffer region – percentage of overhanging

VORONOI DIAGRAM EXAMPLES





Handling OSM maps I

- OpenStreetMap: free map developed by community (www.openstreetmap.org)
- Simplified data structure: only 3 layers for points, lines and areas (+2 in addition)
- Most information about elements (incl. their type) are stored in 'tags', in key=value form
  - *highway* tag for roads, e.g. *highway=primary*
  - *railway* tag for railways, e.g. *railway=subway*
- Objects from different types (e.g. roads, rivers) can be separated only by their keys (attributes)

HANDLING OSM MAPS II

- Data can be downloaded from OSM website or with QGIS' OSM downloader plugin
  - download area has to be defined (boundaries of a rectangle)
  - downloaded *.osm* file can be opened as a vector file
- Downloaded data are saved to 3(-5) layers, with the following attribute fields:
  - Common OSM attributes: *osm_id*, *name*,…
  - Most frequent tag values are saved to separate attribute fields (named after their key e.g. *highway*)
  - Others are in *other_tags* field, in "key"=>"value" form

Useful OSM websites

- download.geofabrik.de
  - OSM extracts of cities, counties (and continents)
  - File formats:
    - *.osm.pbf* – compressed *.osm* files
    - *.shp.zip* – ESRI shapefiles zipped
      (separate files for each object type / layer!)
- overpass-turbo.eu
  - queries for OSM data can be created and run
  - result can be exported to *.kml, .gpx* etc.
- wambachers-osm.website/boundaries
  - administrative boundaries of countries, regions

Using Field calculator I

- For calculations in Attribute table (of the chosen layer)
- Expressions: built from operations (+,×,…) and functions
- Field values can be used as input
  - Format: "fieldname" (≠ text value format: 'text')
- Only result is stored, saving:
  - *Create a new field* → choose name, type & size (length)
  - *Update existing field* (type will not change)
- Example 1: Calculate area of objects (e.g. parks)
  - *Create a new field* (name: e.g. *area*, type: *decimal number (real)*, length: min. *8* digits, min. 2 decimals)
  - Expression: *$area* (from *Geometry* functions)

Using Field calculator II

- Example 2: Calculate length of a track

  Functions needed:

  - *distance()*: calculates it between 2 *geometries (!)*
  - *geometry()*: geometry (spatial data) of a feature
  - *get_feature()*; *get_feature_by_id()*: chooses a feature in a layer according to an attribute value or its ID

  1. Save point ID to a new field *ID* (with function *$id*)
  2. Calculate distance from the previous point, to field *dist*

     ➔ *distance(geometry(get_feature('track_layer', 'ID', $id – 1)), $geometry)*

  Or in one step (without saving ID):

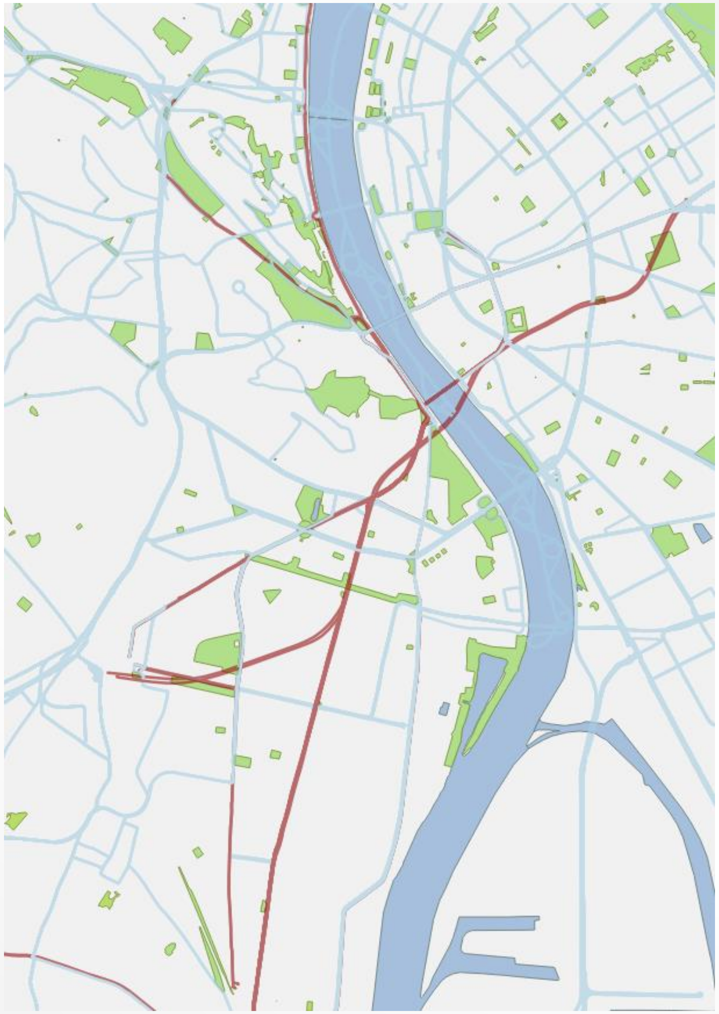
  ➔ *distance(geometry(get_feature_by_id('track_layer', $id–1)), $geometry)*

Handling GTFS spatial data

- General Transit Feed Specification:
  - Structure designed for public transport service data
  - Defined by Google (as an input for Google Maps route planner)
- Usually transit authorities release their data
  - like BKK: https://bkk.hu/gtfs/budapest_gtfs.zip
  - developers can use them in their apps
- Spatial data in GTFS:
  - stops.txt: Stop coordinates (Lat, Lon)
  - shapes.txt: PT routes (point coordinates of polylines)

Calculations with lines I

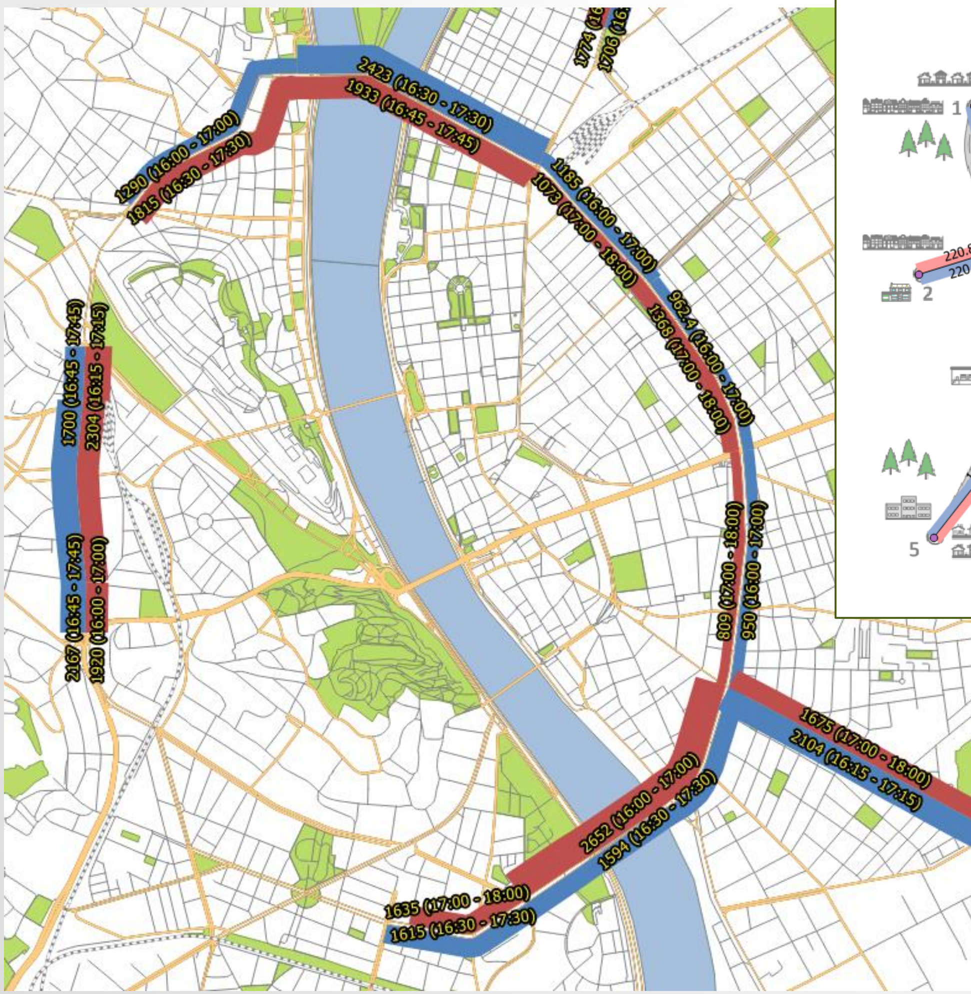
- Creating line layer from point coordinates
  - ➔ Processing Toolbox → Points to path
  - ➔ Select input point layer
  - ➔ Select fields:
    group (line feature ID);
    order (point order in a line)
  - ➔ Select output (if saved)

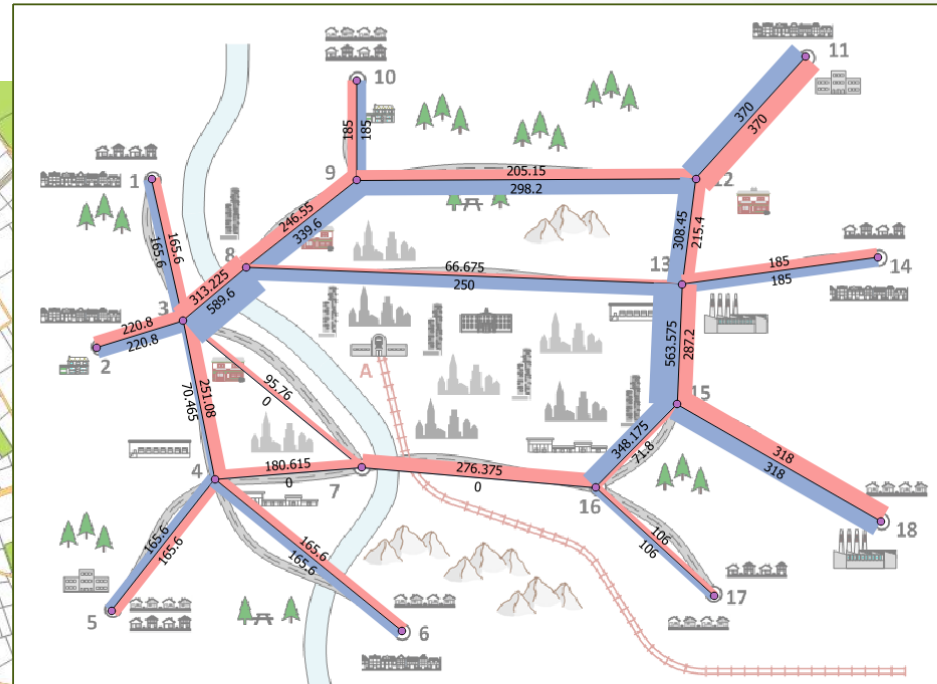


Calculations with lines II

- Creating Sankey (traffic flow) diagram
  - ➔ Add Delimited Text Layer (without geometry)
  - ➔ Create / add diagram layer (used routes)
  - ➔ Join diagram layer it to traffic data (by IDs)
  - ➔ Style: Rule based (visible or not)
    Set two symbols for the two directions
    Line width: traffic vol. / scale
    Offset: (+ or -) width/2
  - ➔ Pay attention to line directions
    (line bars may appear on the wrong side
    → offset sign might be calculated from a field)
  - ➔ Set Labels (if needed)

Sankey diagram examples





Print Composer

- Tool for exporting map images with customized layout (frame, legend etc.)
- Main Composer items:
  - Map: default view is the same as main window, but it can be modified
  - Legend: style and labels of layers; elements can be renamed or removed from list
  - Label (title), Scalebar, etc....
- Export formats: to image, SVG or PDF

GIS Home assignment

- Any transport related topic can be chosen
- Find some GIS data for your topic
- Make some **analysis** on your data
  - Visualisation/styling is not enough on its own *
  - For data procession you can use functions presented on classes or further ones, as well
  - Draw some **conclusion(s)** from the result
- Submission:
  - Documentation (5-10 pages), via e-mail
    Deadline: **30th April**
  - Presentation (max. 5 minutes), on the last 2 lessons

GIS Home assignment – Suggested topics I

Analysis of transport system in a chosen city / area

- Examples for possible measures / calculations:
  - Density (point count / network length by area unit)
  - Covered area by points (total coverage / size of Voronoi cells)
  - Distance analysis (dist. matrix, avg. values for points)
- Calculated measures can be compared by:
  - Transport modes (e.g. bus, metro),
  - Areas (e.g. districts, cities), ... etc.
- Relationship with traffic volumes (if you have), population etc. can be examined
- Visualization of results on map

GIS Home assignment – Suggested topics II

Detailed visualization of traffic volumes

- Station (stop) traffic (min. 3 figures for 1 assignment)
  - Traffic volumes (e.g. circles with different size or heatmap)
  - Shares (e.g. directions, boarding/alighting, peak/day)
  - Distribution / Histogram (e.g. volume by time/direction)
- Flow (Sankey) diagram
  - Traffic volumes by directions, on each section, with labels
- We can provide traffic data for some public transport lines in Budapest
  - Metros M1, M2, M3, M4
  - Trams 4-6, 47-49, 56-61

Thank you for your attention

Tamás Soltész
Assistant researcher
soltesz.tamas@mail.bme.hu
Room St 426

Further functions in QGIS
(supplement)

Extract OSM attribute (with Field calculator)

- Example: Type of railways from *other_tags* field

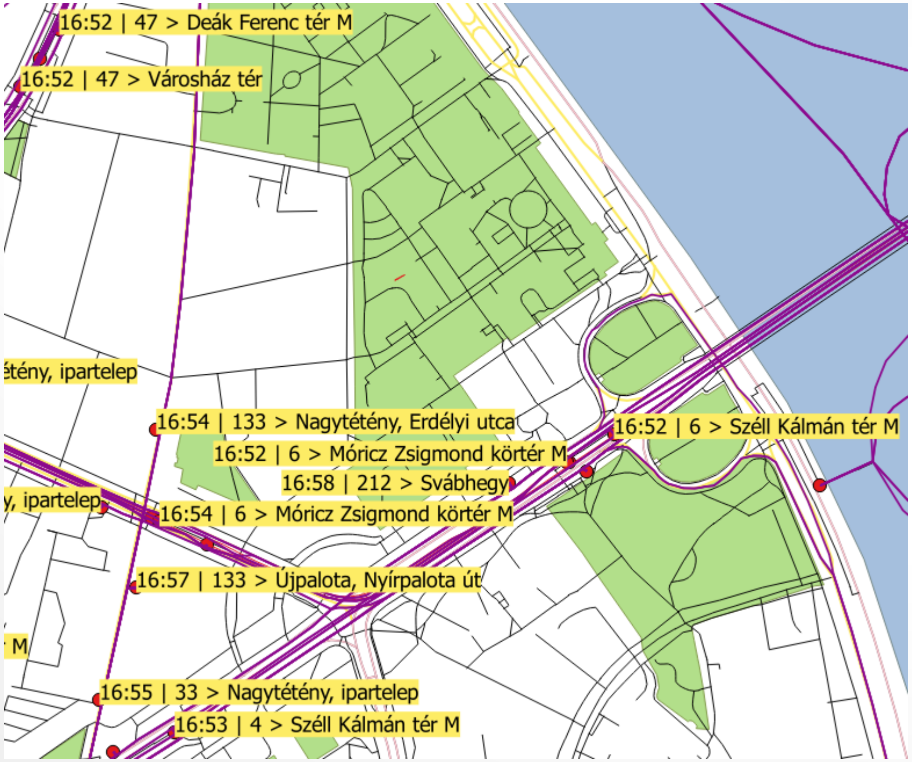
  which is like e.g.: …"*1435*","*railway*"=>"*rail*","*voltage*"…

  String functions: *strpos()* for finding a part; *substr()* to cut

  1. Cut text from *other_tags*, starting with the value after *railway* tag (find the word *railway*)
     a) position of 'r' (in *railway*) to new field *pos* (4-digit integer)
        ➔ *strpos("other_tags", 'railway')* → result e.g.: *107*
     b) cut part of *other_tags* starting with railway type (11 chars after 'r'), to new field *part* (50-character long text)
        ➔ *substr("other_tags", "pos" +11, 50)* → result e.g.: rail","volt…
  2. Cut type from the result of step 1 (text until " sign), to new field *railtype* (~20-char long text), in one step:
     ➔ *substr("part", 1, strpos("part",'"')-1)* → result e.g.: rail ✓

Database connection

- Several kinds of database connections can be built up
- Example: <u>Next PT departure</u> from stops
  - Timetable from GTFS to MS Access
  - Query in Access (next dep. from each stop)
  - Setup ODBC connection
  - Join the imported query and the stops layer
  - Dep. data into stop labels
  - Manual update

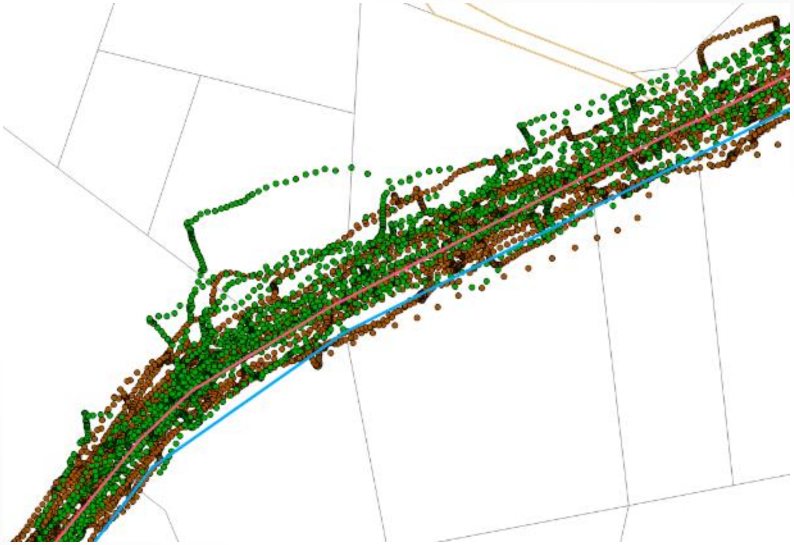


Shortest path search (routing)

- Enable tool panel:
  View → Panels → Shortest path
- Options are in:
  Vector → Road graph → Settings
  - Transportation layer has to be selected
  - Direction and speed attributes can be defined
- Start & End points can be selected on map
- Tool can search shortest or fastest route, and can export the result

Track data procession – Purpose

- All GPS devices have some oscillation (from position errors) around real coordinates that increases measured length
- This additional distance can be eliminated by <u>replacing</u> recorded <u>track points onto</u> the real (known) <u>route</u>, e.g. a street



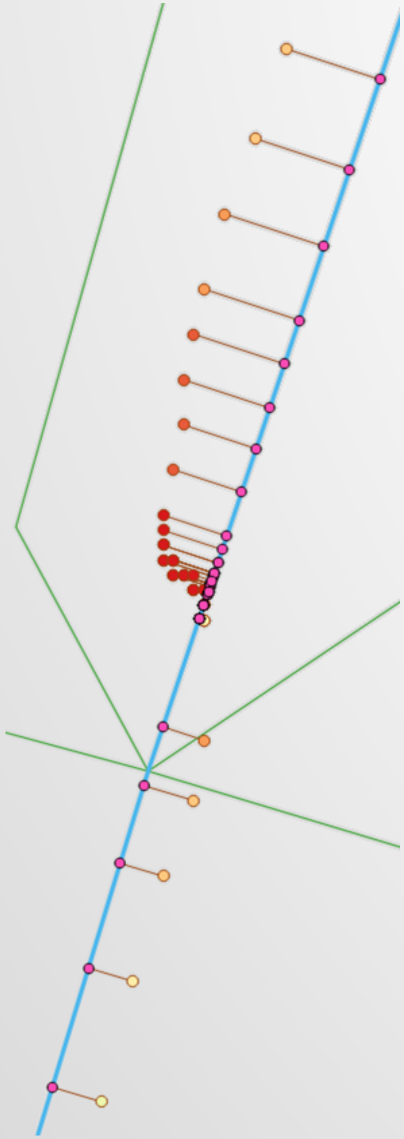
- Simplest way:
  1. Creating reference points (densely) on the route
  2. Replacing track points to the nearest reference point

- There is another, more accurate method for this in *GRASS* module called *v.distance*, but it needs more preparation

Track data procession – Steps (I)

1. Creating reference points on the route:
   a) Create new nodes on route polyline (Densify geometries given an interval, e.g. to 0.5 m)
   b) Extract nodes from densified route layer
   c) Store point geometry to a new field in attribute table (Field calculator, *geom_to_wkt* function)
2. Replacing track points to the nearest ref. point:
   a) If point layers are not already in meter-based CRSs (like UTM), they have to be transformed into that – due to right distance calculations

TRACK DATA PROCESSION – STEPS (II)

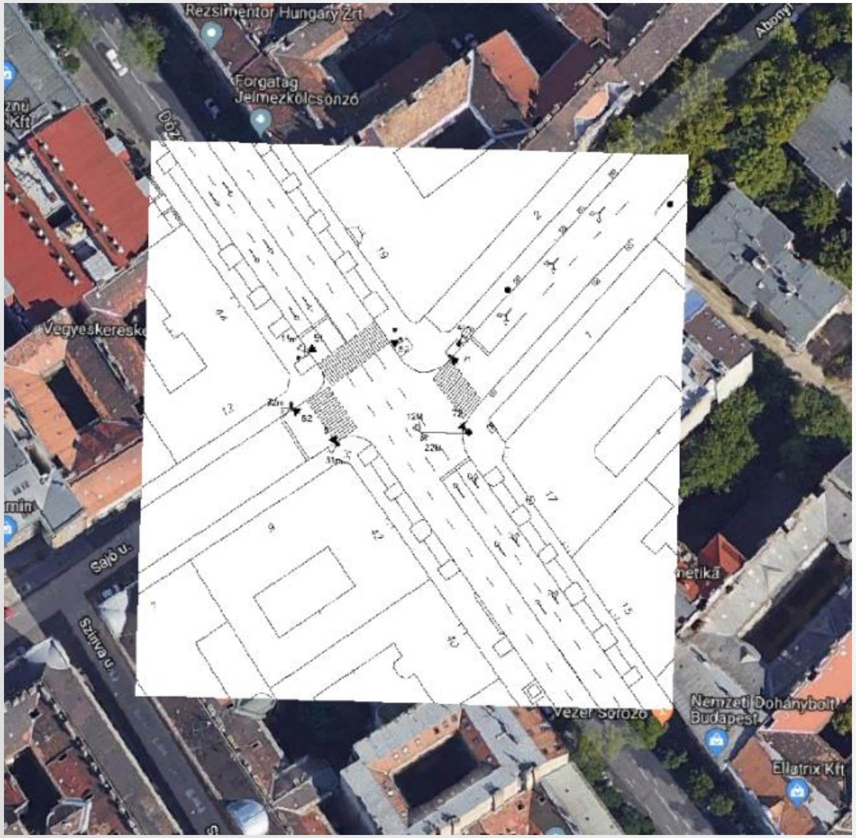


2. Replacing track points to the nearest reference point (continuation):
   b) To each track point, assign the nearest reference point and store its geometry field to the track point's attribute table (Distance to the nearest hub)
   c) Rewrite the coordinates of track points with the stored geometry (Field calc., rewrite *<geometry>* with *geom_from_wkt* function)

- Distance and speed calculations are easier in Excel (after CSV export)

CALCULATING INTERGREEN TIMES I

- At signalized intersections, intergreen times are calculated from the distances of conflict points and stop lines



- Measuring distances:
  1. Import the layout as a raster layer
     - Georeferencer plugin (plugin has to be enabled; min. 6 points are needed)

CALCULATING INTERGREEN TIMES II

2. Vehicle paths can be added manually (as polyline layer)
   - All paths have to start at the stop line
3. Conflict points: Line intersections on paths
   - Relevant ones have to be selected and identified
4. Distances: with *v.distance* function, *along_to* value
   - Separate calculation for each stage (by selection of paths)
   - Copy results back to the points' attribute table (or Excel)