Producing ‘fiets isochronen’, step by step

This document describes the steps to produce street network isochrones for pedestrians and cyclists, going from train stations along the street network for a given distance. The method is just describing general steps and not specific GIS tools. The method was developed using PostgreSQL database with the PostGIS and pgrouting extensions, and visualised using QGIS.

The overall sequence of steps is:

1. Prepare the street network
2. Prepare the train stations
3. Prepare the street network topology
4. Connect the ferry links
5. Locate the station origins on the street network
6. Calculate the catchment area from the station origins
7. Create the street network catchment
8. Produce the isochrone polygons

The main principle behind this method and these steps is that no manual selections or adjustments are made so that the method can be consistently reproduced by others.

Depending on the software used some steps might be skipped. Some steps have alternative approaches that are indicated but not explored further.

This method was developed as part of the the Knooppuntontwikkeling Datasysteem project for the Province Noord Holland. For further information contact:

Jorge Gil – gil.jorge@gmail.com

1. Prepare the street network

Input: TOP10NL wegdeel hartlijn

Alternative sources:

- OpenStreetMap – in principle much more complete for bikes and pedestrians, but not simple to use and has consistency issues
- NWB wegvakken – it's not so complete for bikes and pedestrians

Operations:

1. Select the street network segments with the following criteria:
    - *eindregistratie IS NULL* – excludes roads that have been removed
    - *AND status IN ('in gebruik', 'onbekend')* – similar to above, but also excludes those roads under construction
    - *AND hoofdverkeersgebruik NOT IN ('vliegverkeer','busverkeer','snelverkeer')* – excludes exclusive lanes for planes, buses and cars, not for cyclists or pedestrians
    - *AND verhardingstype != 'onverhard'* – excludes soft terrain in woodland and dunes
    - *AND typeweg NOT IN ('startbaan, landingsbaan','rolbaan, platform')* – excludes 'road' types for plane and rail
2. Select the segments intersecting the limits of the Province Noord Holland and MRA boundaries.
3. Add a column to store the metric length of the segments, which is used to calculate the travel distance later on.

Note: Two attributes were added (boolean data type) to indicate if the road is dedicated to pedestrians or bicycles:

- voetganger = True WHERE hoofdverkeersgebruik = 'voetgangers'
- viets = True WHERE hoofdverkeersgebruik = 'fietsers, bromfietsers'

In the end the same network was used for bicycle and pedestrian isochrones because the rules are not so strict, many segments are shared, and in rural areas pedestrians can use the bike lanes where there is no dedicated pavement.

Output: New layer with the bicycle and pedestrian street network only.

2. Prepare the train stations

Input: NWB spoor treinstations

Alternative sources:

- 9292.nl GTFS – has a more realistic location of stations, but it's not trivial to use and extract the train stations only (https://reisinformatiegroep.nl/ndovloket/Data/ShowDataCollection/5)
- OpenStreetMap

Operations:

1. Select all stations within the boundary of the Province Noord Holland and MRA.

Output: New layer with the selected train stations.

3. Prepare the street network topology

This operation is software specific. The general aim is to identify the connections (topology) between the street network segments' endpoints, where there are street intersections, so that they can be traversed.

Input: New street network and selected train stations layers.

Operations:

1. Create nodes (points) on the endpoints of line segments, keeping only one point per intersection
2. Add to the street network layer the ID of the two nodes at the start and end of each line segment.

Output: New layer with street network intersection nodes.

4. Connect the ferry links

Some of the ferry connection across the Ij in Amsterdam are free and frequent for use by pedestrians and cyclists. These offer a bridge like crossing of the river and as such should be considered in the pedestrian and cycling street network. This steps connects the the ferry

links to the street network, because in most cases they are disconnected and cannot be used for catchment analysis.

Input: Street network, street intersection nodes and train stations layers.

Operations:

1. Select the ferry network segments from the street network:
    - typeweg = 'veerverbinding'
    - Within 3000m from station with naam = ‘'AMSTERDAM CENTRAAL'
2. Select the street nodes that belong to the ferry segments
3. Select the street nodes that are **not** ferry segments
4. Join the ferry nodes to the single nearest street node
5. Update the node IDs of the ferry network segments with the IDs of the joined road network nodes

Alternative: The geometry of the ferry links can be modified to join with the street network segments by moving the ferry link endpoints. These modified links are then treated like normal street segments in the topology operation (step 3).

Output: Updated ids in the street network layer.

5. Locate the station origins on the street network

To calculate isochrones one must have a starting point. This point can be the station point, which gets connected to the nearest street segment. However, this is quite imprecise. The station point is not located near a street, nor on the station building where the entrance to the street is located, and often a station has multiple entrances on both sides of the rail tracks. For these reasons, getting the single nearest street segment to a station point is not appropriate. This step selects the two nearest street segments that are not connected to each other, to obtain street segments on opposite sides of the track.

Input: Street network and train stations layers.

Operations:

1. Select the street segments within 200m from each station
2. Insert the nearest point on the nearest segment into a station origin points table

3. Insert the nearest point on the next street segment where:
    - Start node and end node are not the **start node** of the nearest segment
    - Start node and end node are not the **end node** of the nearest segment

Alternative: For greater accuracy one could use the rail track geometry itself and test if the selected segments are on opposing sides. If not, it would look for the next nearest street segment.

Output: New layer with train station origin nodes.

6. Calculate the catchment area from the station origins

This step is dependent on the network analysis tool. It produces a separate catchment area for each station. It is different from producing a single catchment for all stations combined.

Input: Street network and train stations origins.

Operations:

1. Select each set of station origins and calculate the distance to all nodes on the network within a given metric distance:
    - 3000m for bicycles (10 minutes)
    - 800m for pedestrians (10 minutes)
2. Take the minimum distance result for each street intersection node, for the station origins from the same station.

Alternative: Select a maximum distance and produce multiple bands from the results.

Output: New layer with the shortest distance from each station to the surrounding street nodes.

7. Create the street network catchment

This step might be automatically produced depending on the software used. The aim is to select all street segments that are within the catchment. In addition one has to use linear referencing to extract a part of a segment when only part of a street is within the catchment.

Input: Street network and node distances layer.

Operations:

1. Select the intersection nodes that have a distance to the station below the threshold (3000m or 800m)
2. Select the street segments that have the start node **and** the end node in the nodes list from 1.
3. Insert these segments in a street network catchment table with an attribute for the station name/id.
4. Select street segments that have the start node **or** the end node in the nodes list from 1.
5. Join to each the distance from its node to the station.
6. Calculate the difference between the node distance and the segment length, and convert to a ration between 0 and 1.
7. Use linear referencing to create partial street segments that have the length of the ratio in 6.
8. Insert these partial segments into the street network catchment table.

Alternatives:

- Use only the segments that are entirely within the catchment distance. These have both start and end points with a distance to the station below the threshold.
- Use the segments that have at least one node within the catchment distance. This will include parts of segments that are beyond the desired distance, and in some cases this can be rather long and inaccurate.
- Both these alternatives are much faster to calculate because they skip the linear referencing step and are simply based on attribute queries.

Output: New layer with street network segments with a station id/name and the distance to the station.

8. Produce the isochrone polygons

This step might be automatically produced depending on the software used. The aim is to create a single polygon representing the catchment area around each station. This polygon can be used for visualisation or spatial analysis later on.

Input: Street network segments with a station id and distance.

Operations:

1. Select the street segments that have the same station id
2. Create a buffer around the selection:
    - Buffer distance = 100m
    - Buffer cap should be round if straight cap does not join between different segments
    - Use only the outline of the buffer, excluding all holes.

Alternatives:

- Create the **convex** hull of the street nodes that are within the desired distance of the station. The result is crude but very fast to calculate. Puts areas without street segments within catchment which can be desirable for planning but not for evaluation of the current situation. This does not require step 7.
- Create a **concave** hull of the street nodes that are within the desired distance of the station. The result is better than the convex hull but depends on an abstract user defined threshold. This does not require step 7.

Output: New layer with catchment area polygons, one per station and per travel mode.