Linear Referencing with Floating Car Data

Felix Kunde
Jérémy Surbier
Petra Sauer
ITS term: Sensors that „float“ with the traffic. Or GPX data.

Detect traffic where static sensors cannot see it. „Installment“ is also way cheaper.

Important source to estimate travel times, turn costs, origin-destination analysis etc.

Extenced xFCD = Detect acceleration, breaking, stearing, fuel consumption etc.

Very sensible data. Not many open data sets available (e.g. https://envirocar.org)
FCD trajectory model

(1,1)  (2,1)  (3,1.2)  (4,1.2)  OSM
FCD trajectory model

{11.07.2018 17:00, 1.1, 1.1, 40km/h}

FCD

{11.07.2018 17:01, 3.4, 1.3, 34km/h}

OSM

(1,1)

(2,1)

(3,1.2)

(4,1.2)
FCD trajectory model

FCD

{11.07.2018 17:00, 1.1, 1.1, 40km/h}

OSM

{11.07.2018 17:01, 3.4, 1.3, 34km/h}

Takes road coordinates from OpenStreetMap
Start and end as in the FCD
Interpolated time and speed FCD
Stored in a PostGIS database as LineStringZM

(1,1,40,1531321200) (2,1,29,1531321220) (3,1.2,33,1531321250) (4,1.2,35,1531321270)
SELECT ST_LocateBetween(geom, 1531321230, 1531321260) FROM fcd;

Extract by time

(2.3, 1.05, 30, 1531321230)
(3.1.2, 33, 1531321250)
(3.5, 1.2, 34, 1531321260)
SELECT ST_LocateBetweenElevations(geom, 0, 30)
FROM fcd;

Extract by speed

(1.9, 1, 30, 1531321219)

(2, 1, 29, 1531321220)

(2.3, 1.05, 30, 1531321230)
< 15 km/h
> 19h & < 24h
```
SELECT ST_LocateBetween(geom,
    ST_InterpolatePoint(geom, <point_A>),
    ST_InterpolatePoint(geom, <point_B>)
) 
FROM fcd;
```
SELECT ST_LineSubstring(geom, 
    ST_LineLocatePoint(geom, <point_A>), 
    ST_LineLocatePoint(geom, <point_B>) 
) 
FROM fcd;
How to use a spatial index with trajectories

- A spatial Index on tracking data has its pitfalls
SELECT

-- no need for 4D here
ST_Force2d(geom)

FROM

matched_tracks;

(13.96 51.18 42 1531350000)

(13.58 50.97 39 1531320000)
How to use a spatial index with trajectories

```
SELECT
    ST_Segmentize(
        ST_Force2d(geom),
        0.0005  -- max_segment_length
    )
FROM
    matched_tracks;
```
How to use a spatial index with trajectories

```sql
SELECT
  ST_SubDivide(
    ST_Segmentize(
      ST_Force2d(geom),
      0.0005
    ),
    8 -- max_vertices
  )
FROM
  matched_tracks;
```
SELECT
    ST_Envelope(  -- box is enough
        ST_SubDivide(
            ST_Segmentize(
                ST_Force2d(geom),
                0.0005
            ),
            8
        )
    ) AS fcd_segment_bbox
FROM
    matched_tracks;
How to use a spatial index with trajectories
WITH find_segments AS (  
    SELECT DISTINCT trip_id  
    FROM matched_tracks_segments  
    WHERE fcd_segment_bbox && '<point_A>'  
          OR fcd_segment_bbox && '<point_B>'  
  )  
SELECT id, ST_LineSubString(...)  
FROM matched_tracks t  
JOIN find_segments s ON s.trip_id = t.id  
WHERE _ST_DWithin('<point_A>', geom, 0.0005)  
    AND _ST_DWithin('<point_B>', geom, 0.0005);
FCD Miner

A  B

FCD History

Single trips
FCD Miner

Single trips

- Dump points
- Make lines with 2 points
- Group by line segment
- Use count to filter outliers
Road segments with statistics

- Find unique line endings near start
- Recursive query to find line IDs
- Collect geoms and ST_LineMerge
Produce GeoJSON Output

Recommendations

Node
Express
pg
Socket.io
Leaflet
Differences during the day

- Shortest way is the best between 6 pm and 6 am
- The higher the density the greater the trip variety
- Even taxis don’t take short cuts through residential neighbourhoods
Aggregating trip infos per road segment
Have trajectory data? Try LineStringM and PostGIS functions

Spatial lookups on tracks by indexing smaller chunks of the linestring

Still, too slow for route recommendations (especially on-the-fly aggregates)

PostGIS 2.5: ST_FilterByM – Removal of stop points (if you cannot use map matching)

PostGIS 3.0: Idea to have more dimensions for geometry – Interesting for xFCD

App: github.com/MAGDa-BeuthHS/fcd-miner
Thank you for your attention!

Questions?

The work was supported by the Federal Ministry for Economic Affairs and Energy (BMWi) under grant agreement 01MD15001B (Project: ExCELL).