Managing Mobility Data in PostgreSQL

Contact: Esteban Zimányi (ezimanyi@ulb.be)
Mahmoud Sakr (mahmoud.sakr@ulb.be)

MobilityDB: Architecture

MobilityDB

PostGIS

PostgreSQL

tgeompoint, tgeogpoint,
tint, tfloat, ttext, tbool, ...

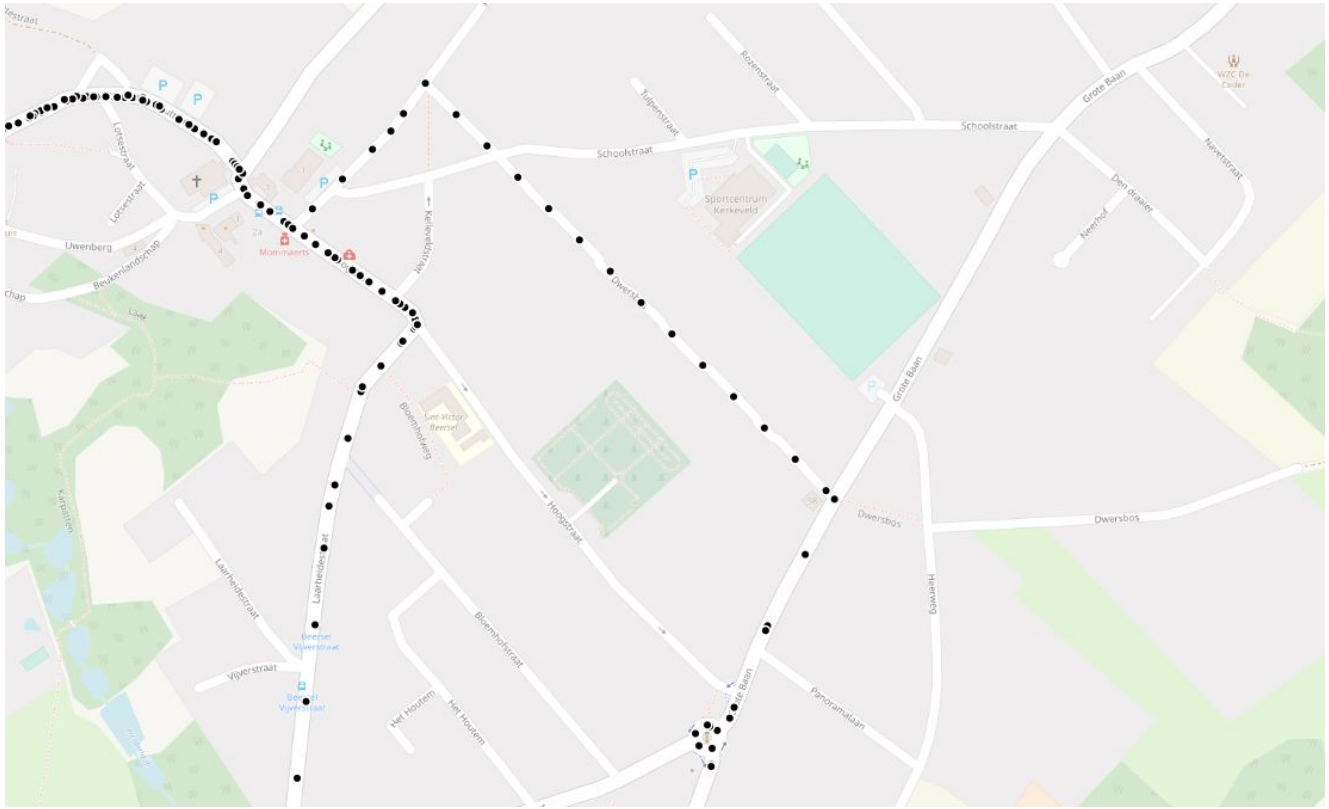
geometry, geography

numeric, monetary, character,
data/time, boolean, enum,
arrays, ranges,
XML, JSON, ...

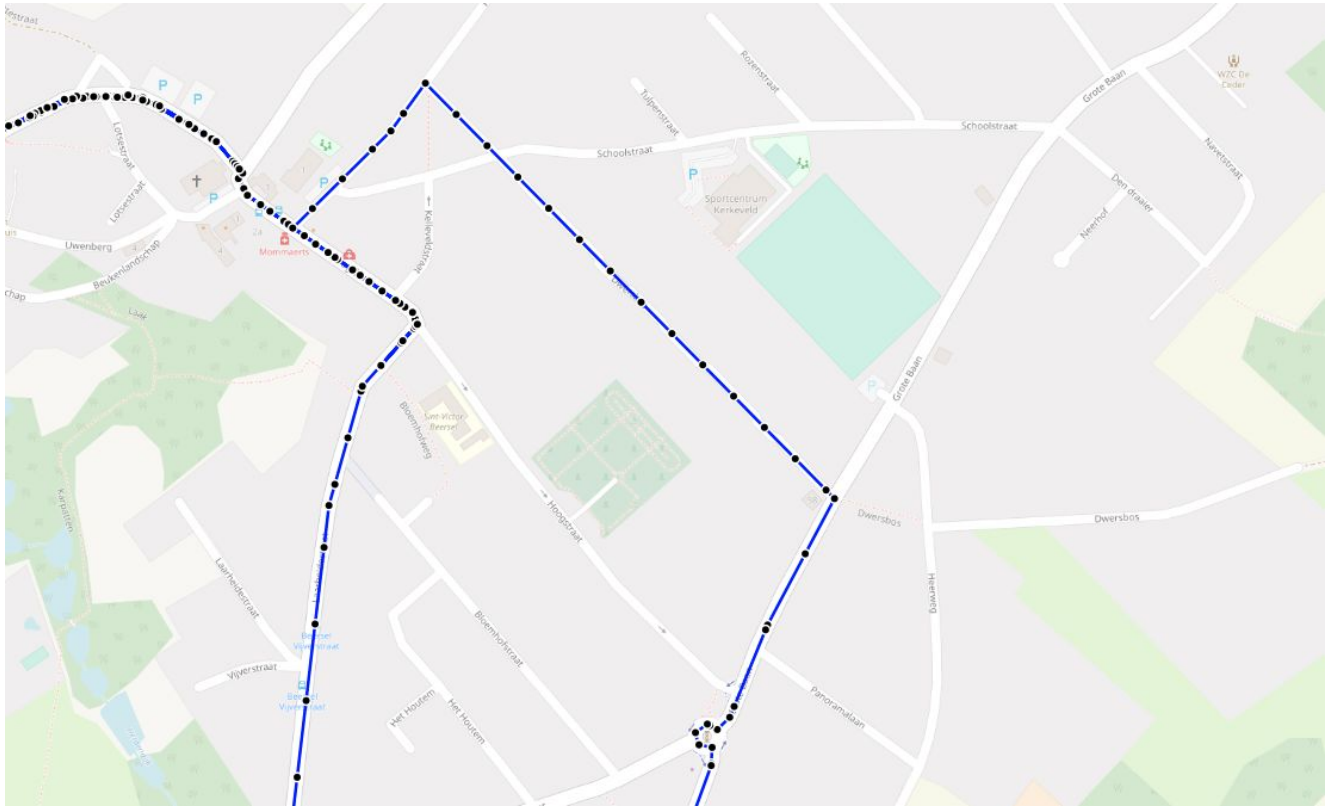
Mobility Data: GPS, DCM

	vehicle integer	day date	seq integer	source bigint	target bigint	t timestamp with time zone	geom text
370	1	2020-06-01	1	16690	34728	2020-06-01 10:01:20.978+02	POINT(490985.514477288 6604283.85305868)
371	1	2020-06-01	1	16690	34728	2020-06-01 10:01:28.081029+02	POINT(490956.213031904 6604299.02553383)
372	1	2020-06-01	1	16690	34728	2020-06-01 10:01:30.978+02	POINT(490940.739620434 6604303.45266601)
373	1	2020-06-01	1	16690	34728	2020-06-01 10:01:34.491879+02	POINT(490921.971156538 6604308.82255297)
374	1	2020-06-01	1	16690	34728	2020-06-01 10:01:39.062744+02	POINT(490911.18429788 6604299.23736115)
375	1	2020-06-01	1	16690	34728	2020-06-01 10:01:40.978+02	POINT(490901.048941147 6604301.05718891)
376	1	2020-06-01	1	16690	34728	2020-06-01 10:01:42.592551+02	POINT(490892.504887324 6604302.5912944)
377	1	2020-06-01	1	16690	34728	2020-06-01 10:01:47.131132+02	POINT(490872.022101019 6604317.29566003)
378	1	2020-06-01	2	34728	16690	2020-06-01 17:53:26.791+02	POINT(490872.022101019 6604317.29566003)
379	1	2020-06-01	2	34728	16690	2020-06-01 17:53:31.929581+02	POINT(490892.504887324 6604302.5912944)
380	1	2020-06-01	2	34728	16690	2020-06-01 17:53:36.791+02	POINT(490910.008155174 6604299.44854041)
381	1	2020-06-01	2	34728	16690	2020-06-01 17:53:37.117666+02	POINT(490911.18429788 6604299.23736115)
382	1	2020-06-01	2	34728	16690	2020-06-01 17:53:39.828856+02	POINT(490921.971156538 6604308.82255297)
383	1	2020-06-01	2	34728	16690	2020-06-01 17:53:46.239706+02	POINT(490956.213031904 6604299.02553383)
384	1	2020-06-01	2	34728	16690	2020-06-01 17:53:46.791+02	POINT(490958.875710594 6604297.64678183)
385	1	2020-06-01	2	34728	16690	2020-06-01 17:53:56.791+02	POINT(491007.174416728 6604272.63740401)
386	1	2020-06-01	2	34728	16690	2020-06-01 17:54:06.791+02	POINT(491055.473122862 6604247.62802619)
387	1	2020-06-01	2	34728	16690	2020-06-01 17:54:10.336527+02	POINT(491072.597559528 6604238.76088375)

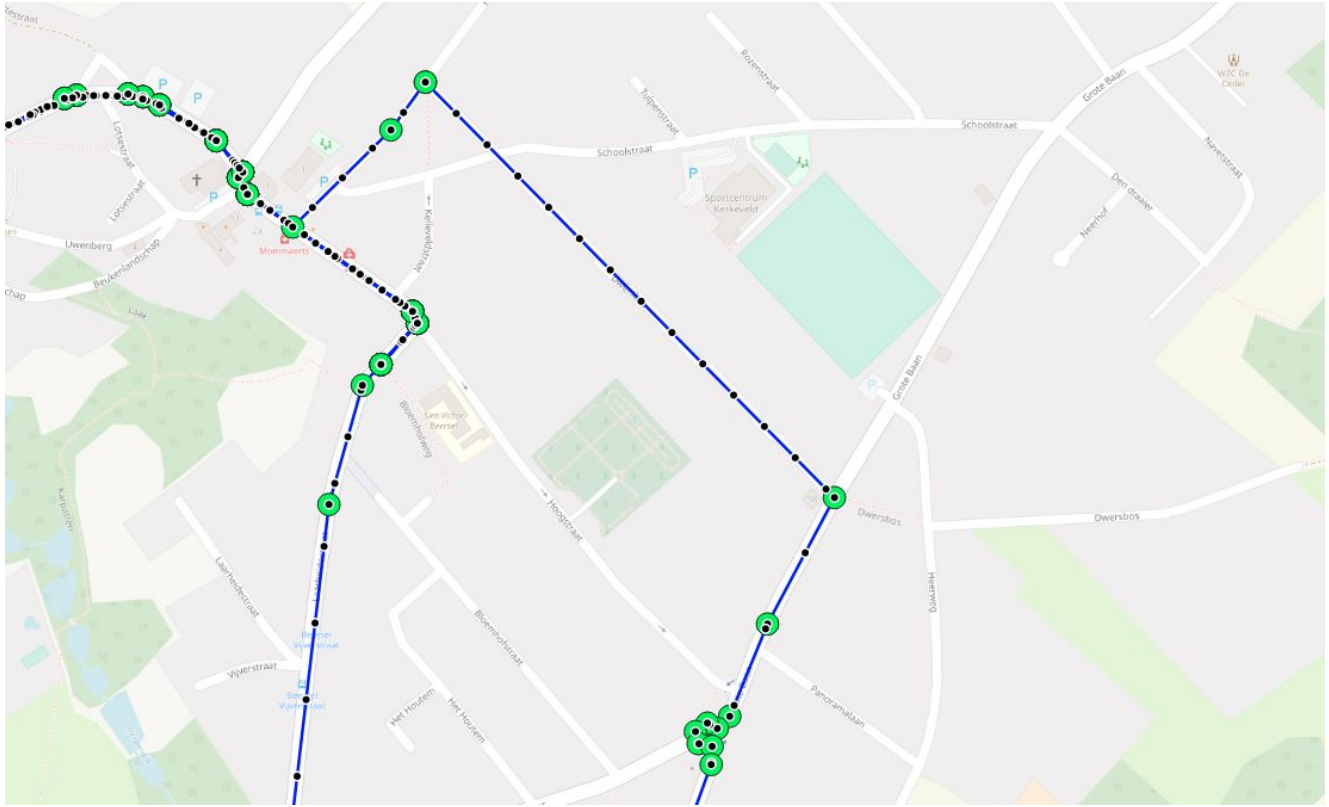
Mobility Data: GPS, DCM



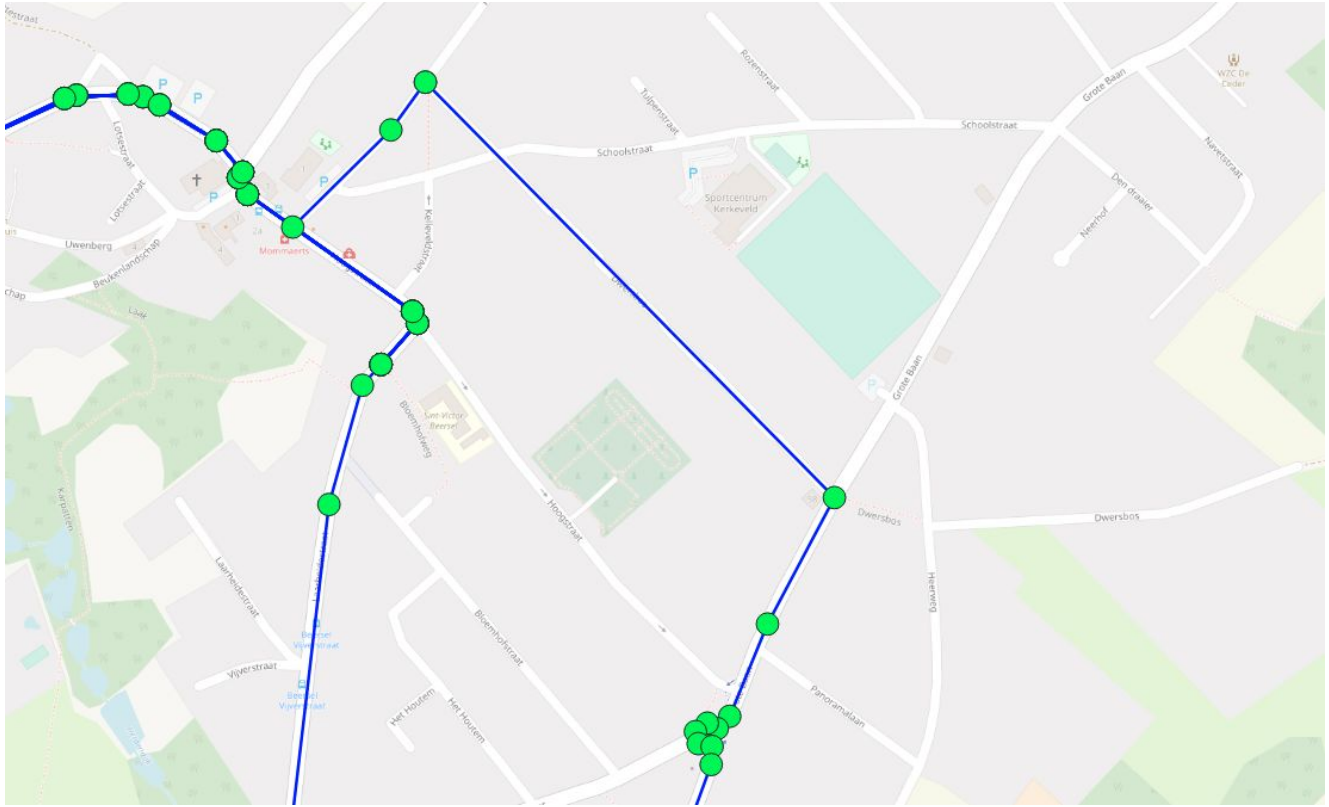
MobilityDB Compact Format (1)



MobilityDB Compact Format (2)



MobilityDB Compact Format (3)



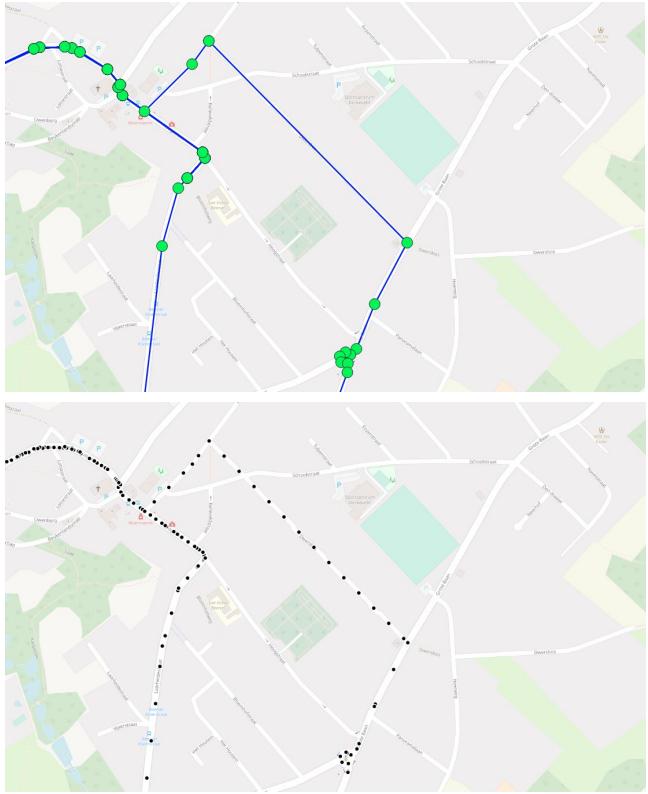
MobilityDB Compact Format (4)

Improvement over existing tools

- 10 - 100x reduction in size
- From millions of rows into thousands
- Query speedup

Novel benefits

- Interpolation
- Native database types
- Rich mobility analytics API
- Significant reduction in development time



PostGIS -> MobilityDB migration

44384	2015-04-06 06:38:00	POINT(37.3826816 55.7937783)
44384	2015-04-06 06:38:30	POINT(37.3826816 55.7937783)
44384	2015-04-06 06:39:00	POINT(37.3826816 55.7937783)
44384	2015-04-06 06:39:30	POINT(37.3826816 55.7937783)
44384	2015-04-06 06:40:00	POINT(37.3826816 55.7937783)
44384	2015-04-06 06:40:30	POINT(37.3826816 55.7937783)
44384	2015-04-06 06:41:00	POINT(37.3826816 55.7937783)
44384	2015-04-06 06:41:30	POINT(37.3826816 55.7937783)
44384	2015-04-06 06:42:00	POINT(37.3826816 55.7937783)
44384	2015-04-06 06:42:30	POINT(37.3826816 55.7937783)

44384	[POINT(37.3826816 55.7937783)@2015-04-06 06:38:00+03, POINT(37.3826816 55.7937783)@2015-04-06 0...
44399	[POINT(37.6126166 55.7274032)@2015-04-06 07:14:29+03, POINT(37.6118683 55.7274732)@2015-04-06 0...
44399	[POINT(37.6127783 55.7265099)@2015-04-06 05:32:14+03, POINT(37.6127783 55.7265099)@2015-04-06 0...
62736	[POINT(37.6078283 55.7158566)@2015-04-06 05:35:17+03, POINT(37.607475 55.71504)@2015-04-06 05:35:...
62771	[POINT(37.6124233 55.7264416)@2015-04-06 05:07:57+03, POINT(37.6124233 55.7264416)@2015-04-06 0...
67756	[POINT(37.608135 55.7163933)@2015-04-06 04:47:23+03, POINT(37.60777 55.7153983)@2015-04-06 04:47:...
67762	[POINT(37.6093483 55.7190449)@2015-04-06 16:58:07+03, POINT(37.6094966 55.7188982)@2015-04-06 1...
67762	[POINT(37.6099266 55.7209099)@2015-04-06 04:41:30+03, POINT(37.60921 55.7190516)@2015-04-06 04:4...

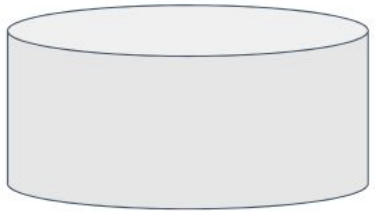
10 billion rows a day

> 500 MB per day

~ 300 GB per year

15 thousand rows

~ 5 MB per day



2GB per year

Temporal Aggregations: Travel Time

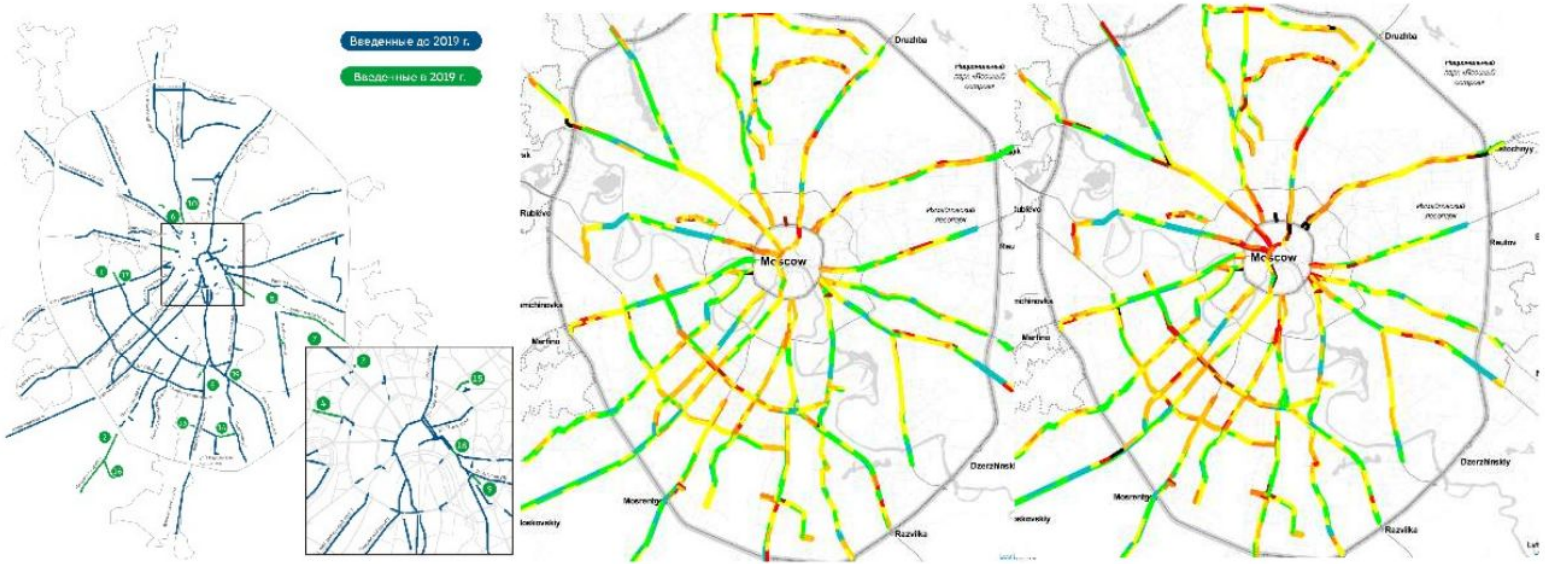
Trolleybus №49

September 2017

September 2018

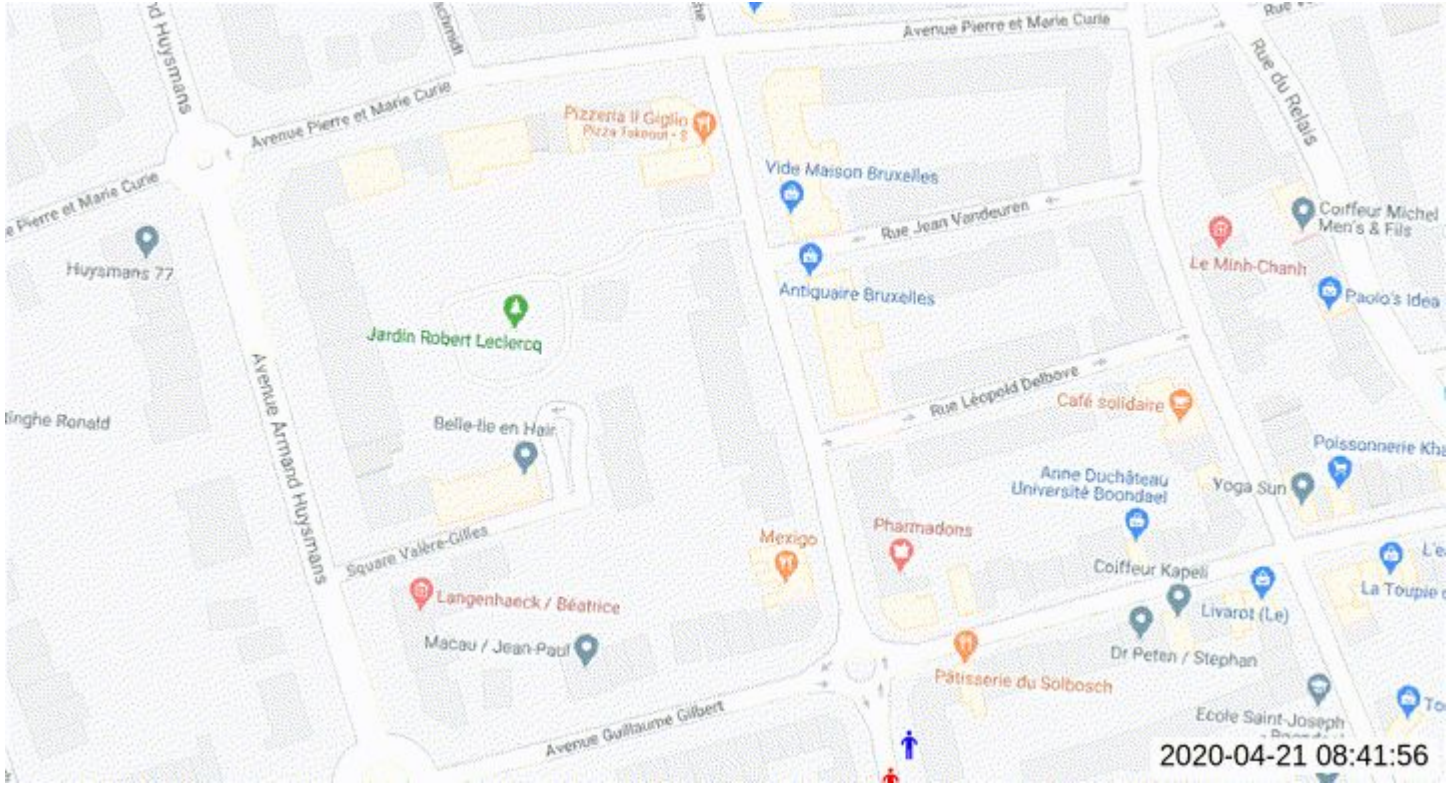
Temporal Aggregations: Velocity Maps

Moscow bus lanes



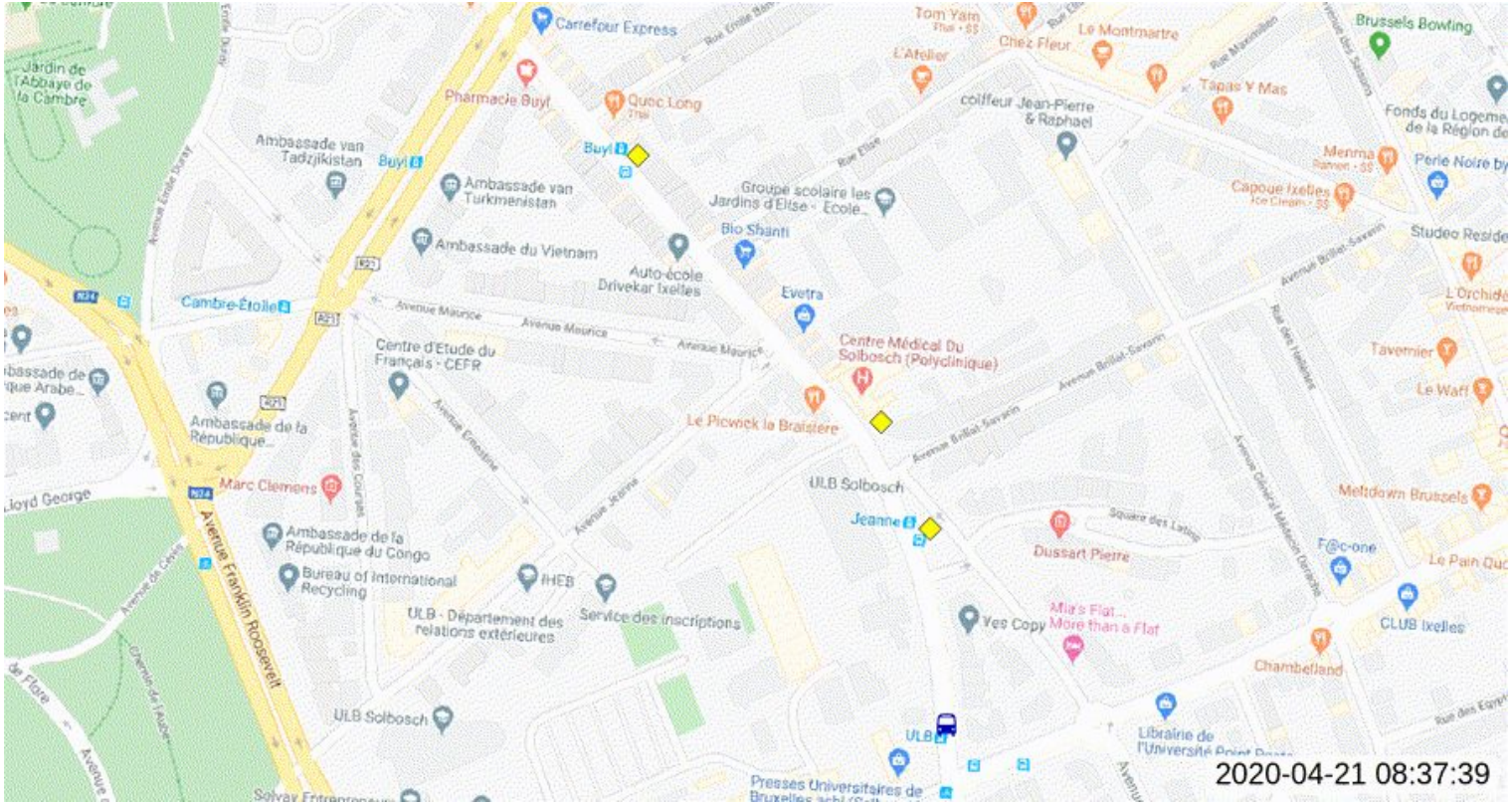
Spatiotemporal Proximity: Marine Data

Spatiotemporal Proximity: COVID-19

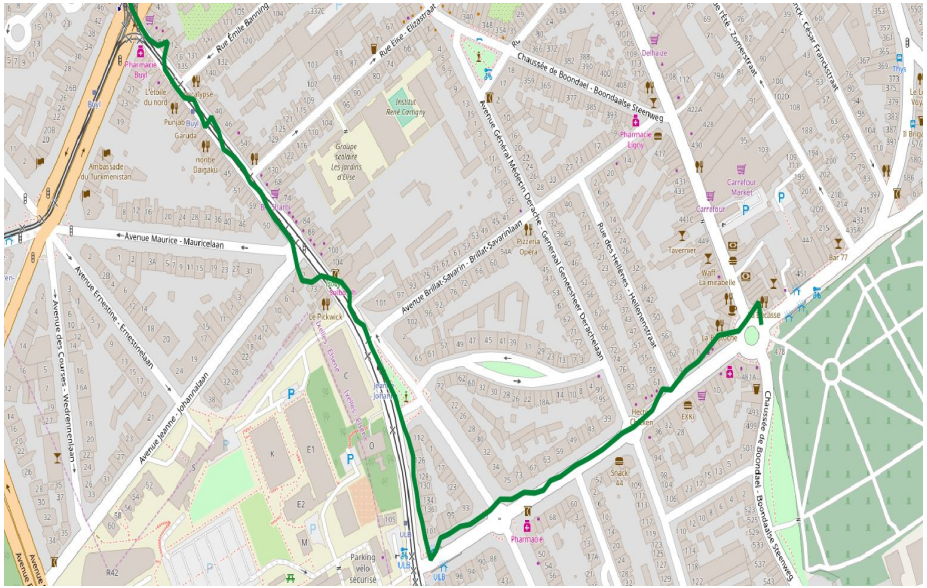


https://ec.europa.eu/info/sites/info/files/recommendation_on_apps_for_contact_tracing_4.pdf

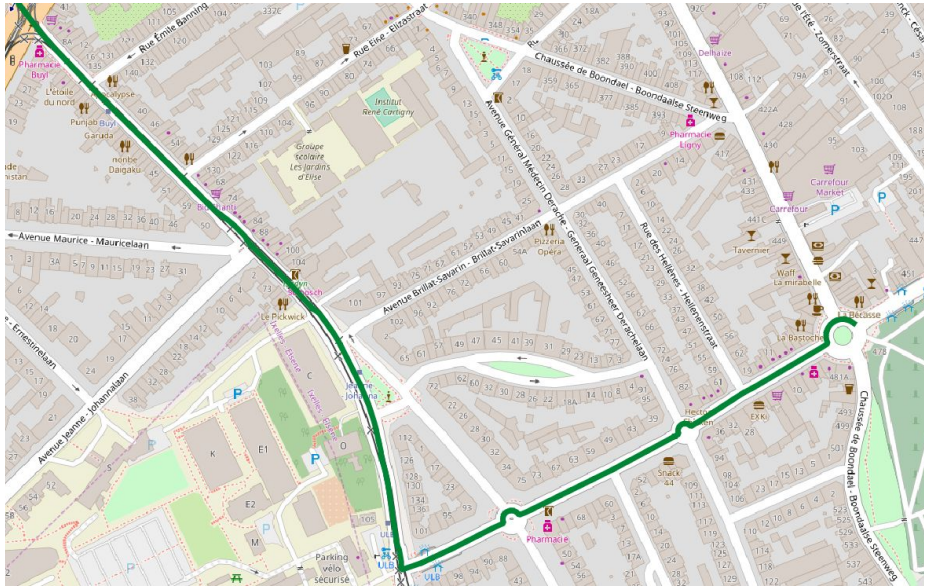
Spatiotemporal Proximity: Smart Advertising



Map Matching: Continuous Trajectories



Original



Map-matched

MobilityDB Ecosystem

MobilityDB Network	MobilityDB Geometry	MobilityDB MapMatch	MobilityDB Stream	MobilityDB View	MobilityDB Generator
pgRouting	AWS MobilityDB Cloud		kafka	MobilityDB Python	MobilityDB Java
docker	kubernetes	citusdata	psycopg	asyncpg	PostgreSQL JDBC
kepler.gl mapbox	QGIS	Grafana	plotly	python	Java
	MobilityDB	PostGIS	PostgreSQL	ubuntu	

MobilityDB

An open source geospatial trajectory data management & analysis platform

Location tracking devices, such as GPS, are nowadays widely used in smart phones, and in vehicles. As a result, geospatial trajectory data are currently being collected and used in many application domains. MobilityDB provides the necessary database support for storing and querying such geospatial trajectory data.

MobilityDB is implemented as an extension to PostgreSQL and PostGIS. It implements persistent database types, and query operations for managing geospatial trajectories and their time-varying properties.

A geospatial trajectory is generally collected as a sequence of discrete location points and timestamps, as illustrated in the top most figure. In reality, however, the movement is continuous. Therefore MobilityDB interpolates the movement track between the input observations, as illustrated in the figure in the middle. As such, the moving object location and properties can be queries, effectively approximated, at any time instant. While this interpolation restores the movement continuity, it does not correspond to an increased storage

Moving Features SWG

Chair(s):

Ishimaru, Nobuhiro (Hitachi, Ltd., Defense Systems Division)

Kim, Kyoung-Sook (National Institute of Advanced Industrial Science & Technology (AIST))

SAKR, Mahmoud (Université Libre de Bruxelles (ULB))

Group Charter:

Download Charter document

Group Description:

Moving Features

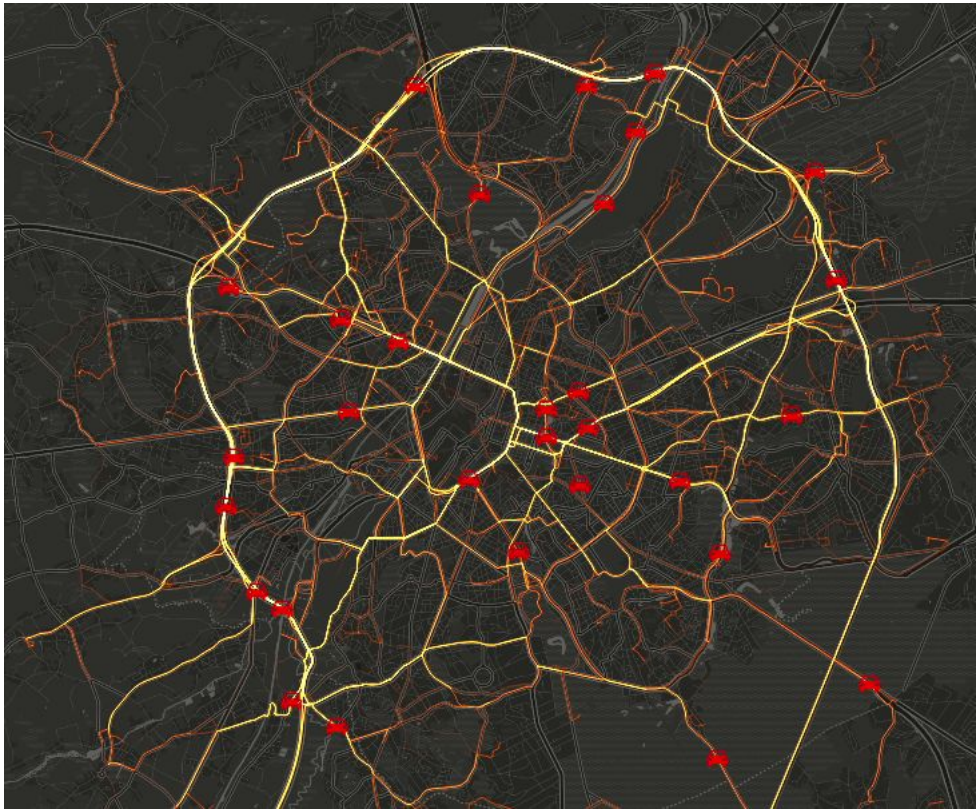
With the development of communication and positioning technology such as Global Navigation Satellite System (GNSS), Wi-Fi, and Beacon, collecting movement data for moving features, typically on vehicles and pedestrians, has become easy. A moving feature is a feature whose location continuously changes over time. A moving feature is widely used in several application domains, such as LBS (Location Based Service), marketing, and public health. These applications have considered not only the current location of features but also the historical data of a feature's movement. These data are used to analyze the patterns of moving features and provide input to predictive models. Moreover, innovative applications for smart cities require the overlay and integration of moving feature data from different sources to create enhanced social and business values. For example, sharing moving feature data widely and seamlessly helps organizations to handle marketing at the micro-level, trace people contacts in penedemics, make an efficient evacuation plans in the case of a sudden disaster, control autonomous vehicles and personal mobility, and more based on people's activities and movement conditions.

Purpose of the Standards Working Group

The goals for the Moving Features SWG are to develop and maintain enhancements to the moving features standard based on the following issues:

- Data models and encoding formats to exchange the moving features data including GNSS-logged, network-constrained, semantic, and region-based moving features.

BerlinMOD Data Generator*



* Visualization done by Maxime Schoemans as part of his PhD Thesis

Introduction to MobilityDB: Tables

```
CREATE TABLE trips (
  vehicle integer NOT NULL,
  day date NOT NULL,
  seq integer NOT NULL,
  source bigint,
  target bigint,
  trip tgeompoint,
  trajectory geometry,
  CONSTRAINT trips_pkey
  PRIMARY KEY (vehicle, day, seq)
)
```

```
CREATE TABLE municipalities (
  id integer,
  name text,
  population integer,
  percpop numeric,
  popdensitykm2 integer,
  noenterp integer,
  percenterp numeric,
  geom geometry
)
CREATE TABLE planet_osm_point (
  ...
  Name text
  way geometry(Point, 3857),
  ...
)
```

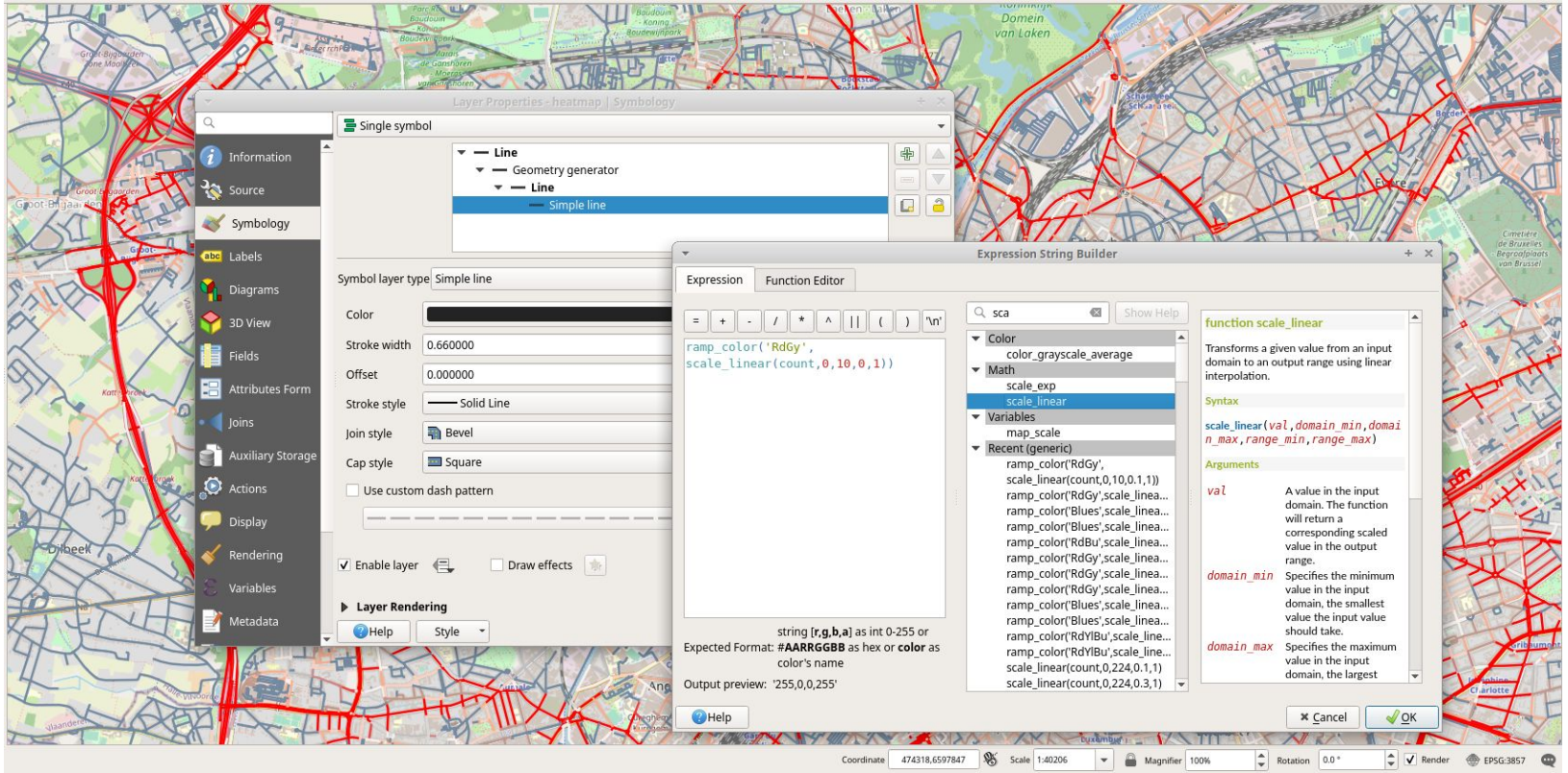
Querying the Database

- Compute dataset summaries

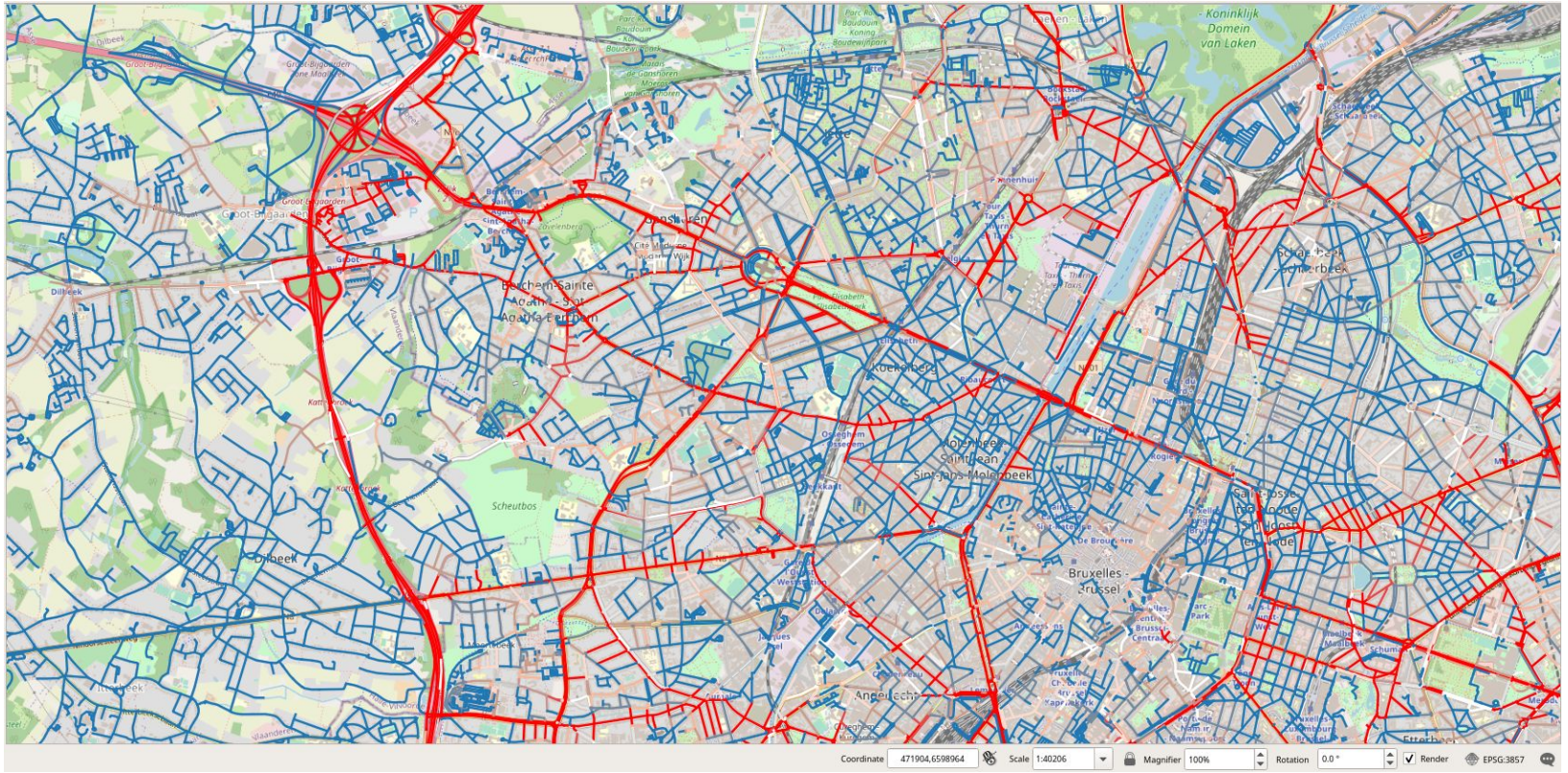
```
SELECT MIN(startTimestamp(trip)), MAX(endTimestamp(trip)),
     MIN(length(trip)), MAX(length(trip)), AVG(length(trip)),
     MIN(duration(trip)), MAX(duration(trip)), AVG(duration(trip)),
     MIN(numInstants(trip)), MAX(numInstants(trip)), AVG(numInstants(trip)),
     AVG(numInstants(trip) * 60 / extract (epoch from timespan(trip)))
          AS avgPointsPerMinute
FROM trips;

2020-06-01 08:00:09.334+02 | 2020-06-03 00:25:53.92958+02 |
461.88900092268483 | 37543.95951933945 | 14914.107113980963 |
00:01:15.052723 | 01:10:16.893662 | 00:25:25.772065 |
25 | 3093 | 1244.1916666666666667 |
45.63301083678095
```

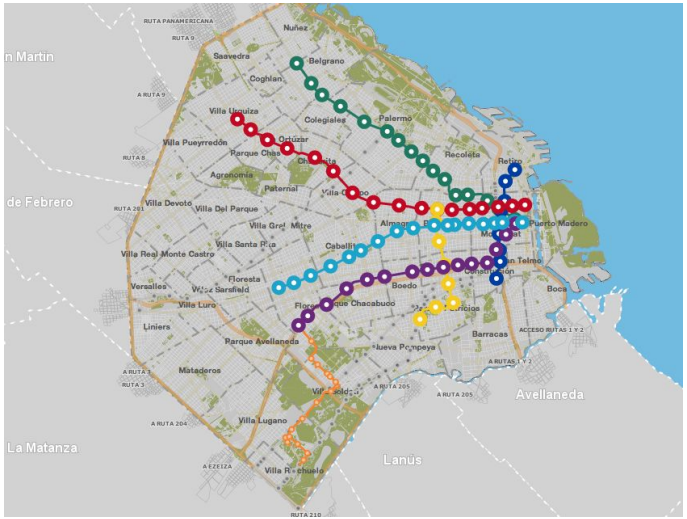
Defining Heat Maps in QGIS



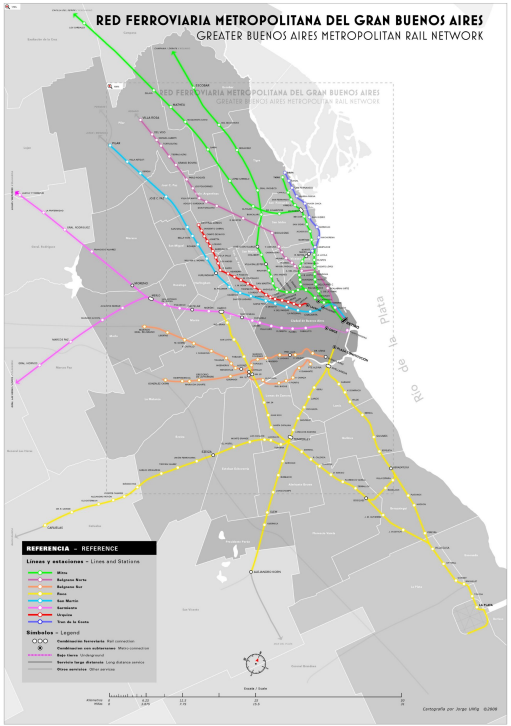
Visualizing Heat Maps in QGIS



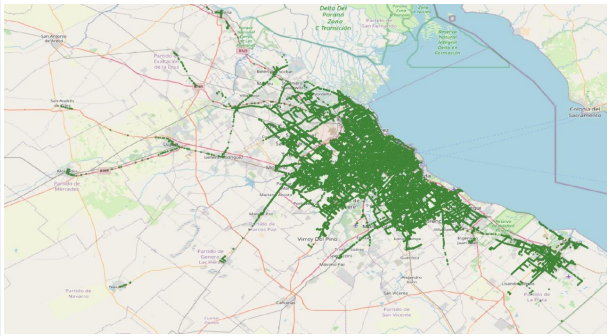
MobilityDB and GTFS: Public Transport in Buenos Aires, Argentina



Subway network



Train network



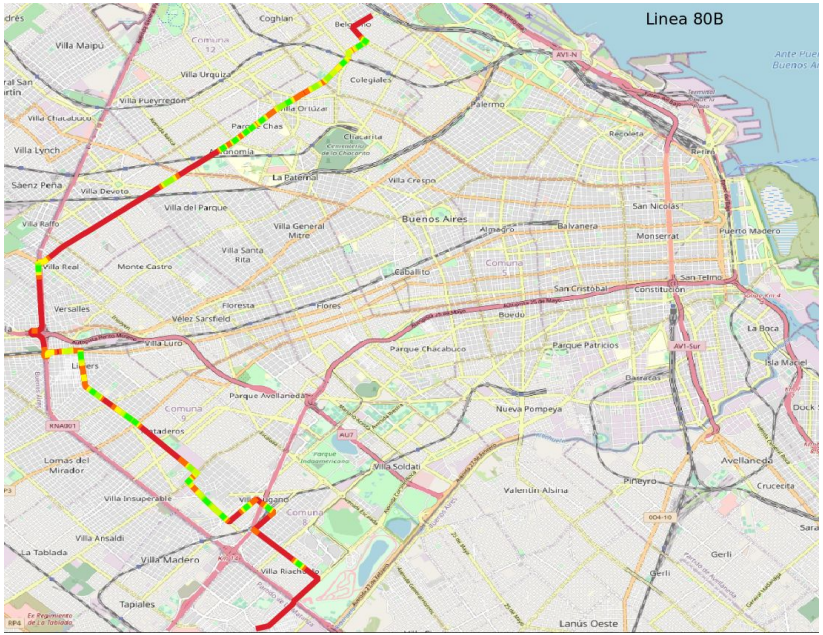
Bus network

- Juan Godfrid, Pablo Radnic, Alejandro Vaisman, *Instituto Tecnológico de Buenos Aires, Argentina*
- Esteban Zimányi, *Université Libre de Bruxelles, Belgium*

MobilityDB and GTFS: Public Transport in Buenos Aires, Argentina

- A first study using moving object databases and GTFS Static and Realtime
- GTFS Static: scheduled trajectories of trains, subways and buses in Buenos Aires during a week
- GTFS Realtime: real trajectories of some buses (the bus network is very large in Buenos Aires)
- Data stored in MobilityDB as trips (spatiotemporal data)
- MobilityDB used to analyze public transport data: delays, speed, etc.

Heatmap: Predicted vs Real Schedule

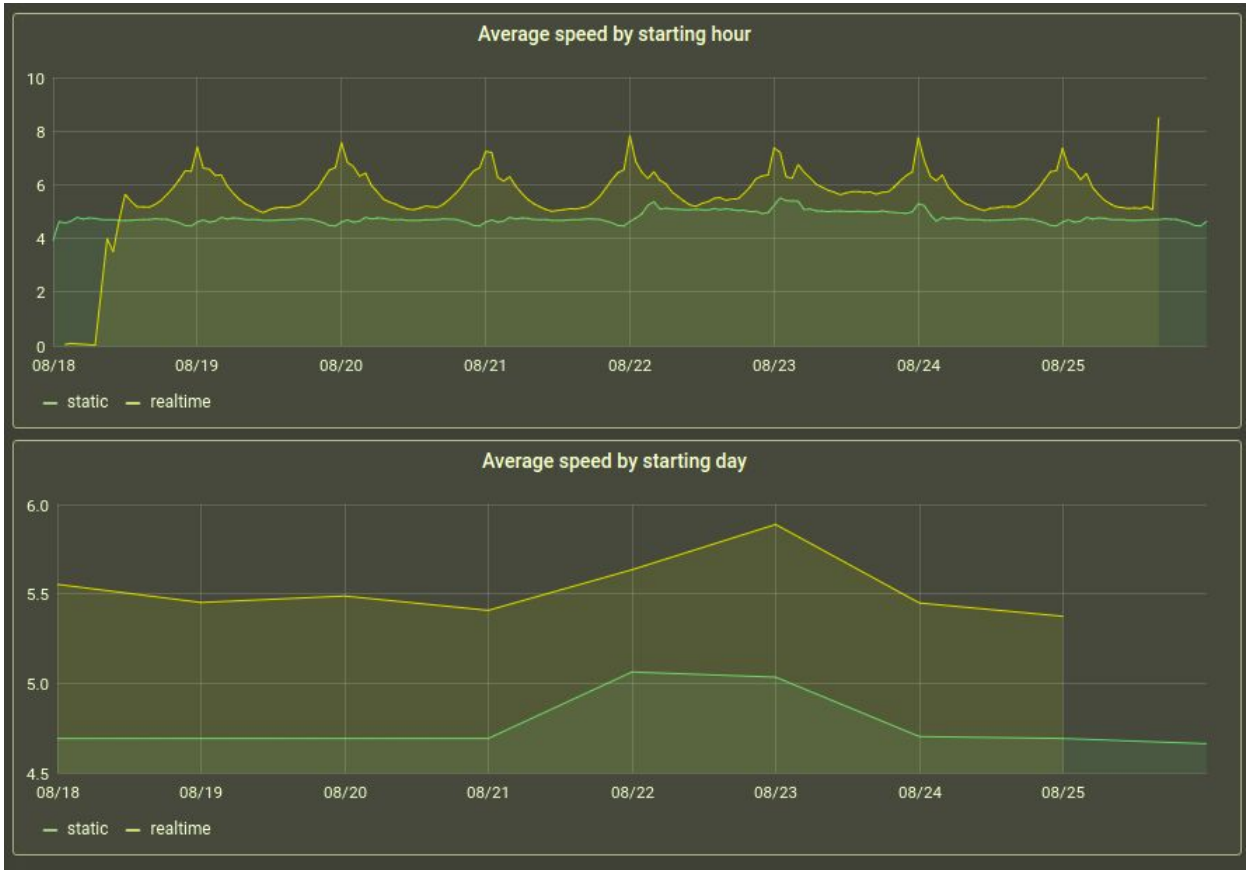


Bus Line #
152

Out of schedule
Close to schedule
On time trajectory

Bus Line #
80

Visualizing Speed Comparison with Grafana

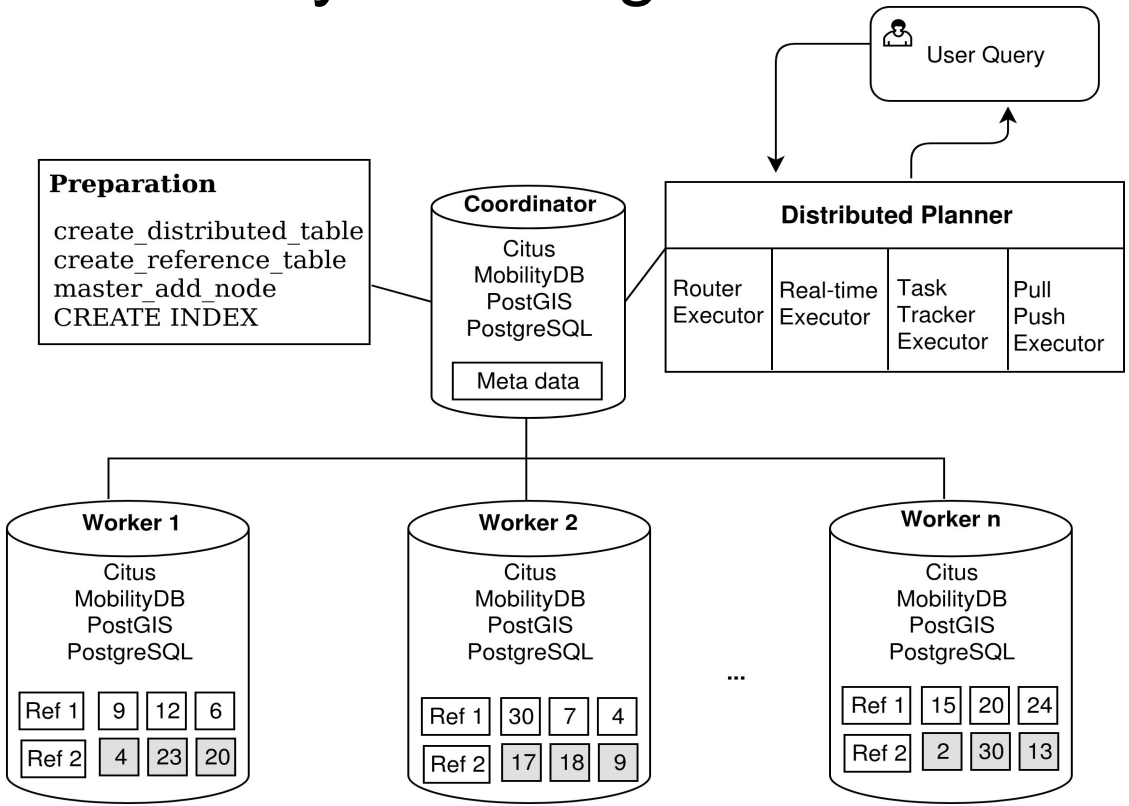


Vision & Directions

MobilityDB as Mainstream Database for Mobility

- Distributed Database
- Cloud Integration
- Visualization
- ...

Distributed MobilityDB: Integration with Citus



* PhD project of Mohamed Bakli

Query Distribution Example

```
EXPLAIN
SELECT S.id , T.tripId
FROM trips T, ROI R
WHERE
  intersects(T.trip , R.geom)
```

* Trips table is partitioned by the tripid as hash key

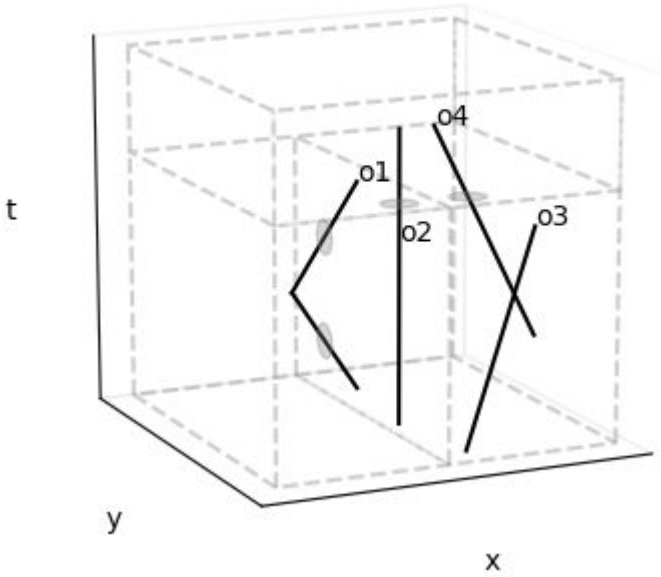
* ROI table is replicated on all nodes

```
Distributed MobilityDB Plan (Broadcast-Join Query):
 -> Scanning the global index: (Hierarchical):
  -> Total number of partitions: 36
 -> Remove Duplicates
  -> Number of Parallel Tasks: 36
   -> Task 1 (WorkerNode1):
     -> Local Query:
     -> SELECT S.id, T.tripId FROM trips T, ROI R
          WHERE intersects(T.trip,S.geom)
     -> Local Plan:
     -> Nested Loop
     -> Seq Scan on ROI R
     -> Index Scan using trips_shard_1_spgist_idx on
trips_shard_1
          -> Index Cond: (trip && S.geom)
          -> Filter: intersects(trip, S.geom)
```

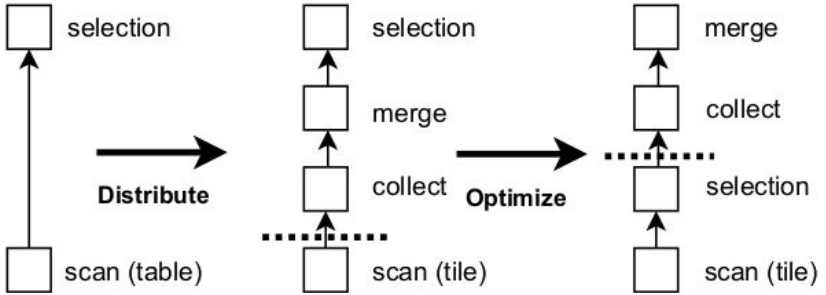
The Spatiotemporal Join Challenge

```
SELECT S.id , T.tripId
FROM trips T1, trips T2
WHERE
  intersects(T1.trip, T2.trip)
```

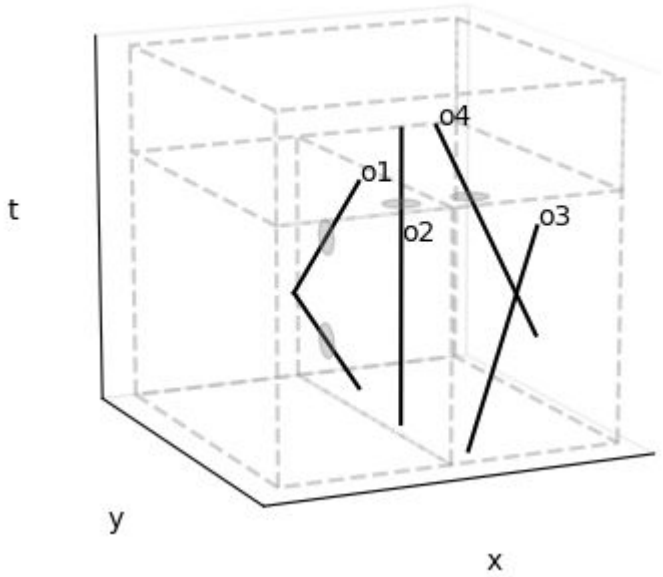
Multidimensional Tiling



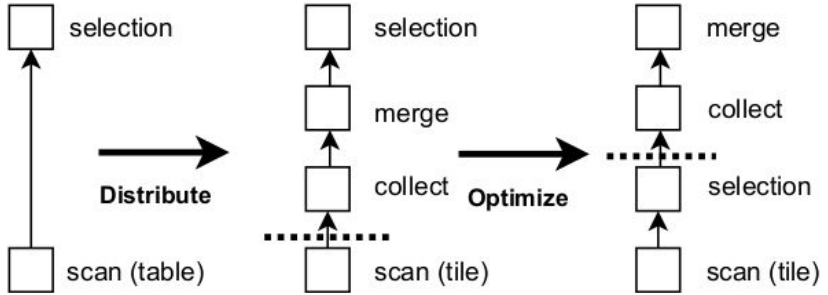

```
SELECT tripId FROM trips
WHERE intersects(trip ,'Polygon ((...) )')
```



Multidimensional Tiling

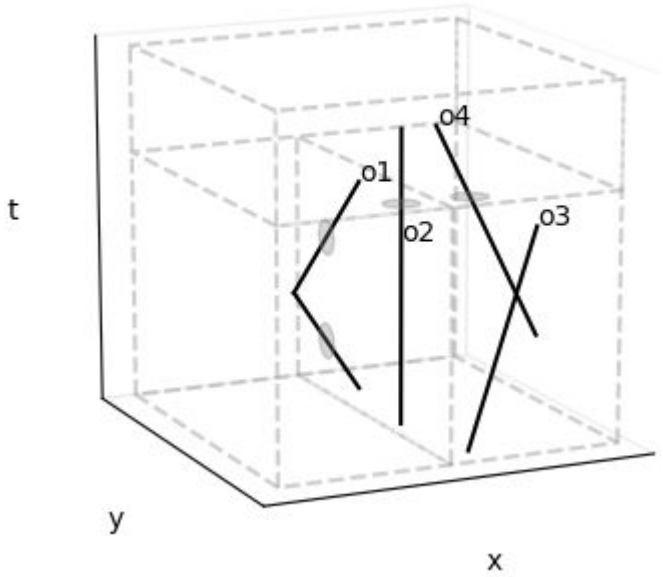



```
SELECT tripId FROM trips
WHERE intersects(trip ,'Polygon ((...) )')
```

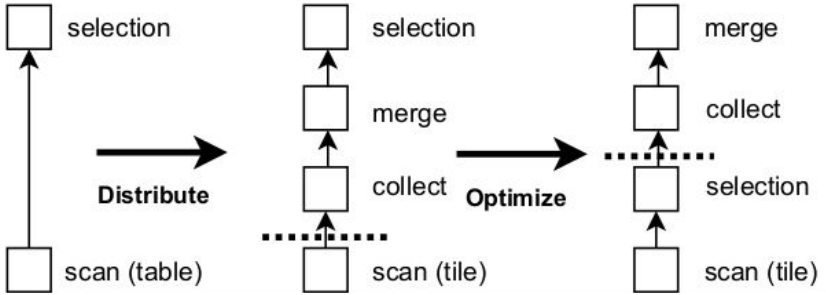



```
SELECT S.id , T.tripId
FROM trips T1, trips T2
WHERE
  intersects(T1.trip, T2.trip)
```

Multidimensional Tiling




```
SELECT tripId FROM trips
WHERE intersects(trip ,'Polygon ((...) )')
```




```
SELECT S.id , T.tripId
FROM trips T1, trips T2
WHERE
  intersects(T1.trip, T2.trip)
```

```
SELECT length(trip) FROM trips
```

MobilityDB on Azure*: Objectives

Automation 

- Exploiting the capabilities of Infrastructure-as-Code and Kubernetes to enable the automatic deployment and management of the MobilityDB cluster

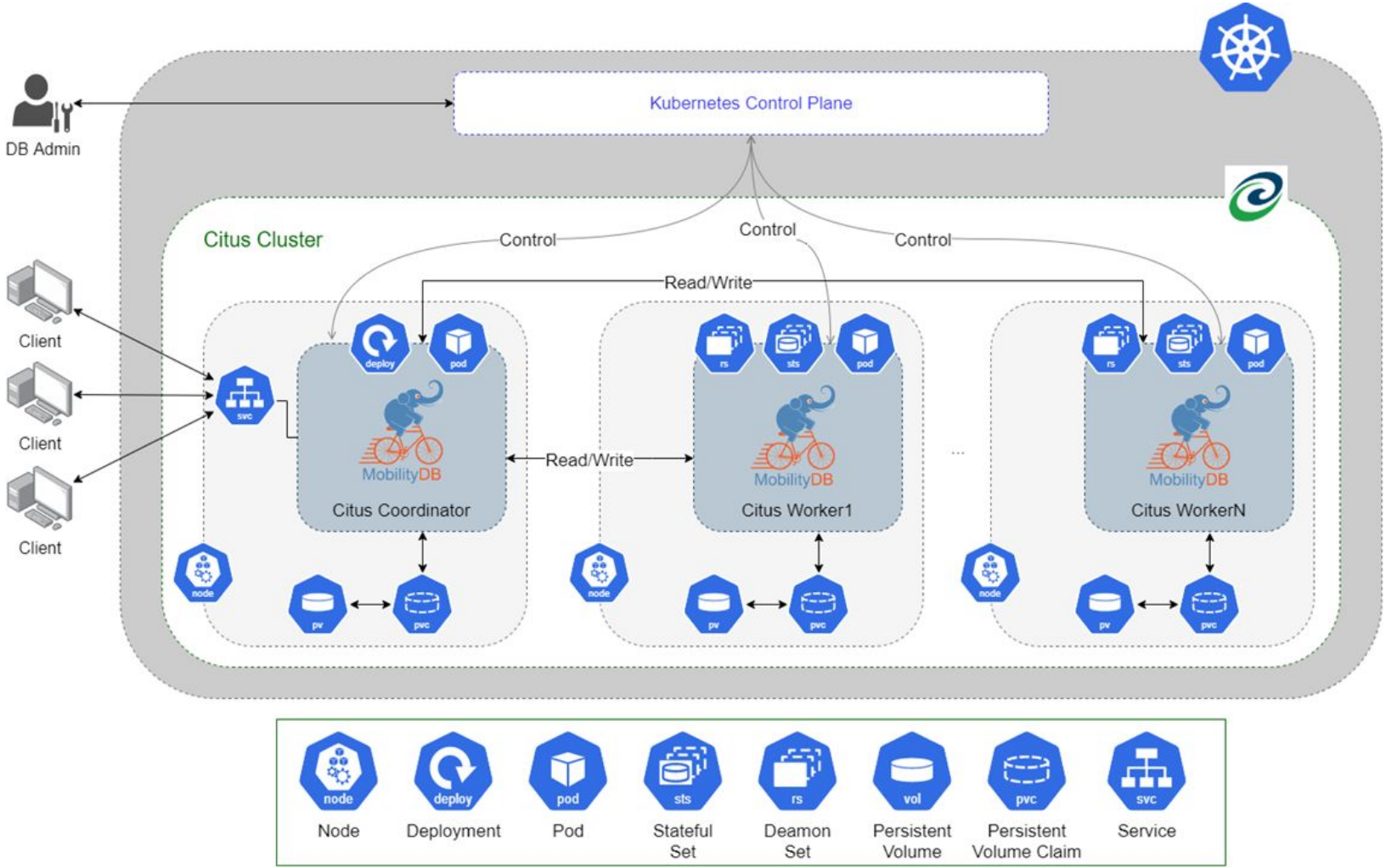
Scalability 

- Using Citus to move from an on-premises, single-node PostgreSQL Server to a cloud and fully distributed PostgreSQL cluster

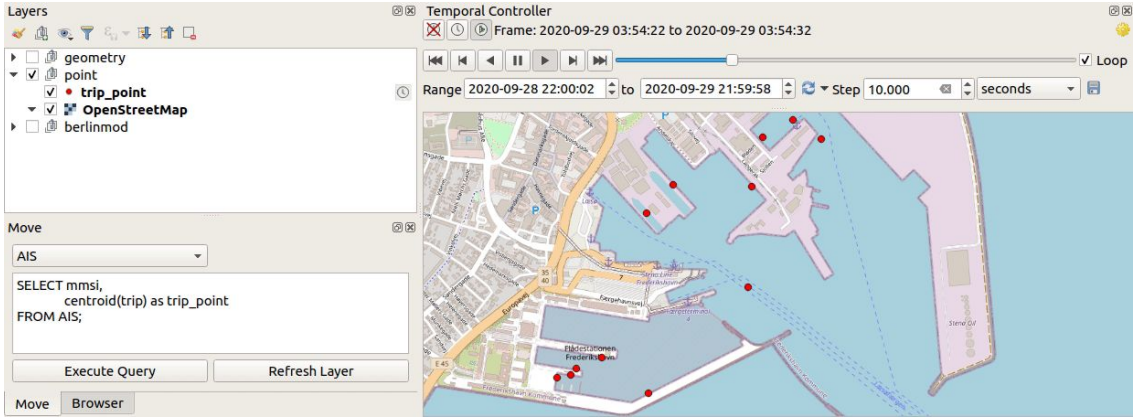
Elasticity 

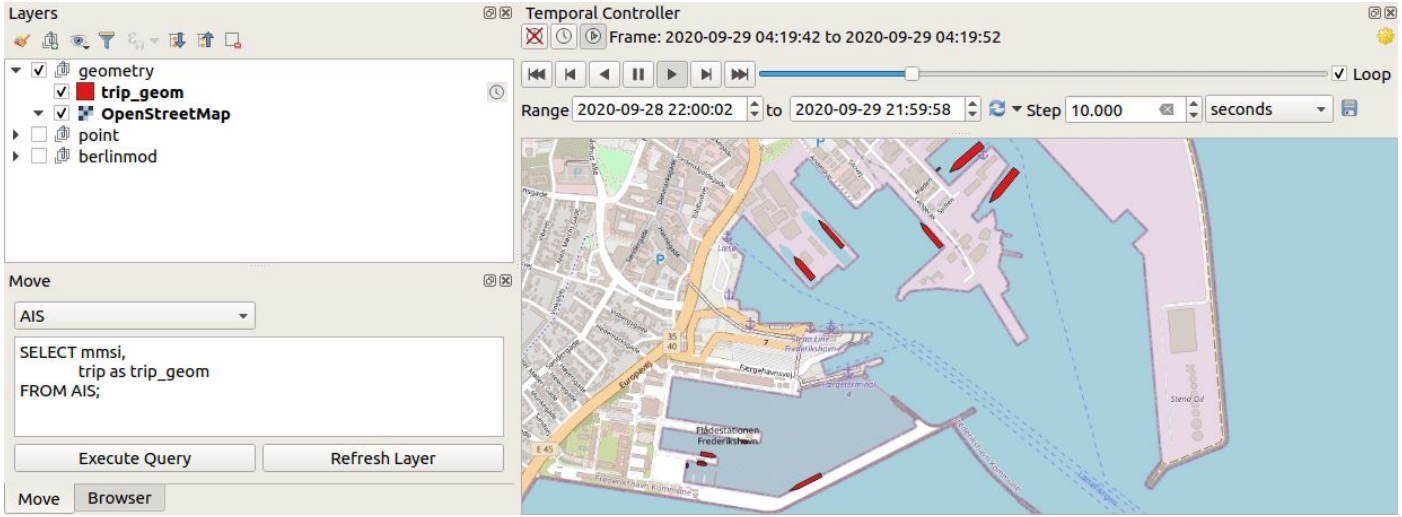
- Applying and experimenting with different auto-scaling techniques to perform cluster scale in/out operations, depending on predefined rules

* Work done by Dimitris Tsesmelis as part of his Master's Thesis


DB Admin
Kubernetes Control Plane
Citus Cluster
Control
Control
Control
Read/Write
Client
Client
Client
svc
deploy
pod
MobilityDB
Citus Coordinator
Read/Write
rs
sts
pod
MobilityDB
Citus Worker1
...
rs
sts
pod
MobilityDB
Citus WorkerN
node
pv
pvc
node
pv
pvc
node
pv
pvc
node
deploy
pod
sts
rs
vol
pvc
svc
Node
Deployment
Pod
Stateful Set
Deamon Set
Persistent Volume
Persistent Volume Claim
Service

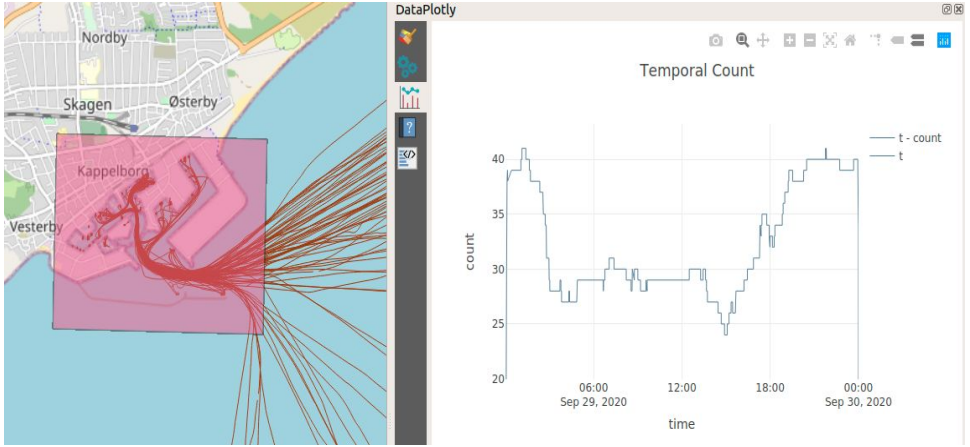
Scalable Trajectory Visualization

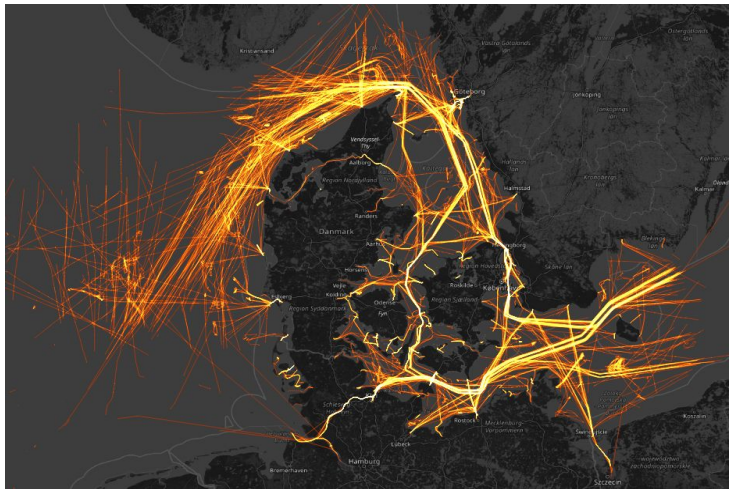




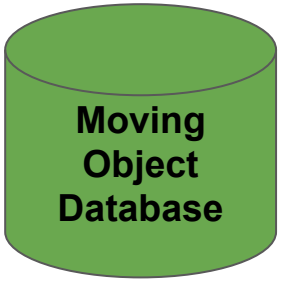
* Visualization done by Maxime Schoemans as part of his PhD Thesis

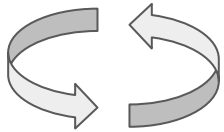
Complex Queries & Transformations





Database for Visualization





query-visualize

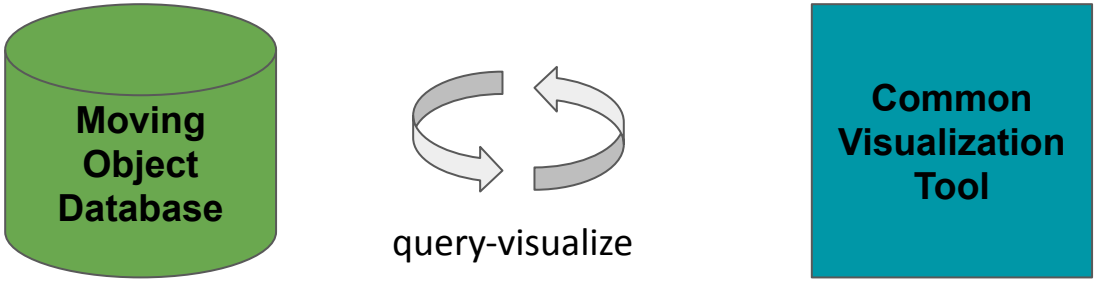
MobilityDB

Selection
Join
Aggregation
Transformation
Distributed Processing
Vector Tiles

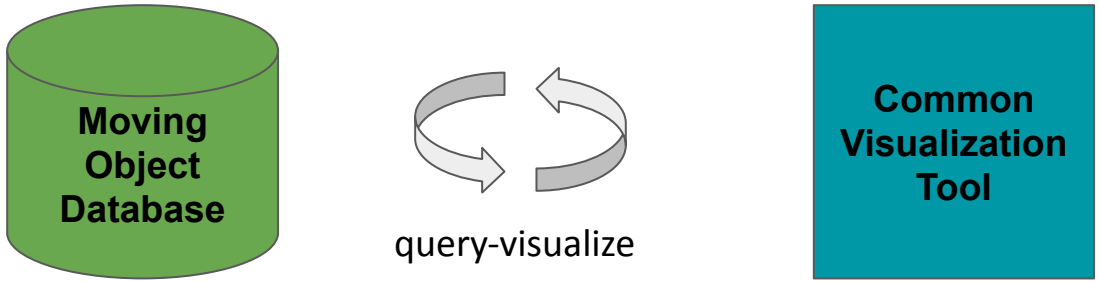
QGIS, Cesium, Keppler, GeoMesa, …

Static Geometry
Animation
Interaction
Symbology & labeling
Plugins

Database for Visualization



Database for Visualization



Future Work: Roadmap

- More functions
  - Data cleaning and preprocessing
  - Complex patterns
  - Ecological analysis
  - ML models
- Domain-specific implementations
  - Automotive
  - Public transit and multimodal mobility
  - AIS
- Real-time stream processing
- Periodic moving objects (e.g., GTFS, NeTEx, SIRI)
- Large-scale data visualization

Acknowledgements

MobilityDB team: Arthur Lesuisse, Mohamed Bakli, Maxime Schoemans, Dimitris Tsesmelis, Fabrício Ferreira, Robin Choquet, Xinyang Li, many Master's students

MobilityDB Steering Committee: Anita Graser (Moving Pandas), Darafei Praliaskouski (PostGIS), Krishna Chaitanya Bommakanti

PostgreSQL community: Oleg Bartunov (PosgresPro), Regina Obe (PostGIS), Vicky Vergara (pgRouting), Nyall Dawson (QGIS)

MobilityDB community: Nina Belyavskaya (Mosgortrans), Marco Slot et al (Microsoft), Alejandro Vaisman et al (IT Buenos Aires), Dimitris Zissis et al (MarineTraffic), Alessandra Raffaetà et al (Univ. Venezia), Yannis Theodoridis (Univ. Piraeus), Cyril Ray et al (French Naval Academy) and many others

MobilityDB

https://mobilitydb.com

Home | The Project | Documentation | Installation | Tutorials | Contact

MobilityDB

An open source geospatial trajectory data management & analysis platform.

View on Github

Location tracking devices, such as GPS, are nowadays widely used in smart phones, and in vehicles. As a result, geospatial trajectory data are currently being collected and used in many application domains. MobilityDB provides the necessary database support for storing and querying such geospatial trajectory data

MobilityDB is implemented as an extension to PostgreSQL and PostGIS. It implements persistent database types, and query operations for managing geospatial trajectories and their time-varying properties.

Learn More

Benefits

Compact geospatial data storage

Rich mobility analytics

Easy to use full SQL interface

Python adapters

Moving Features standards (OGC)

PostgreSQL ecosystem