Geographic Information Systems

Christian Grothoff

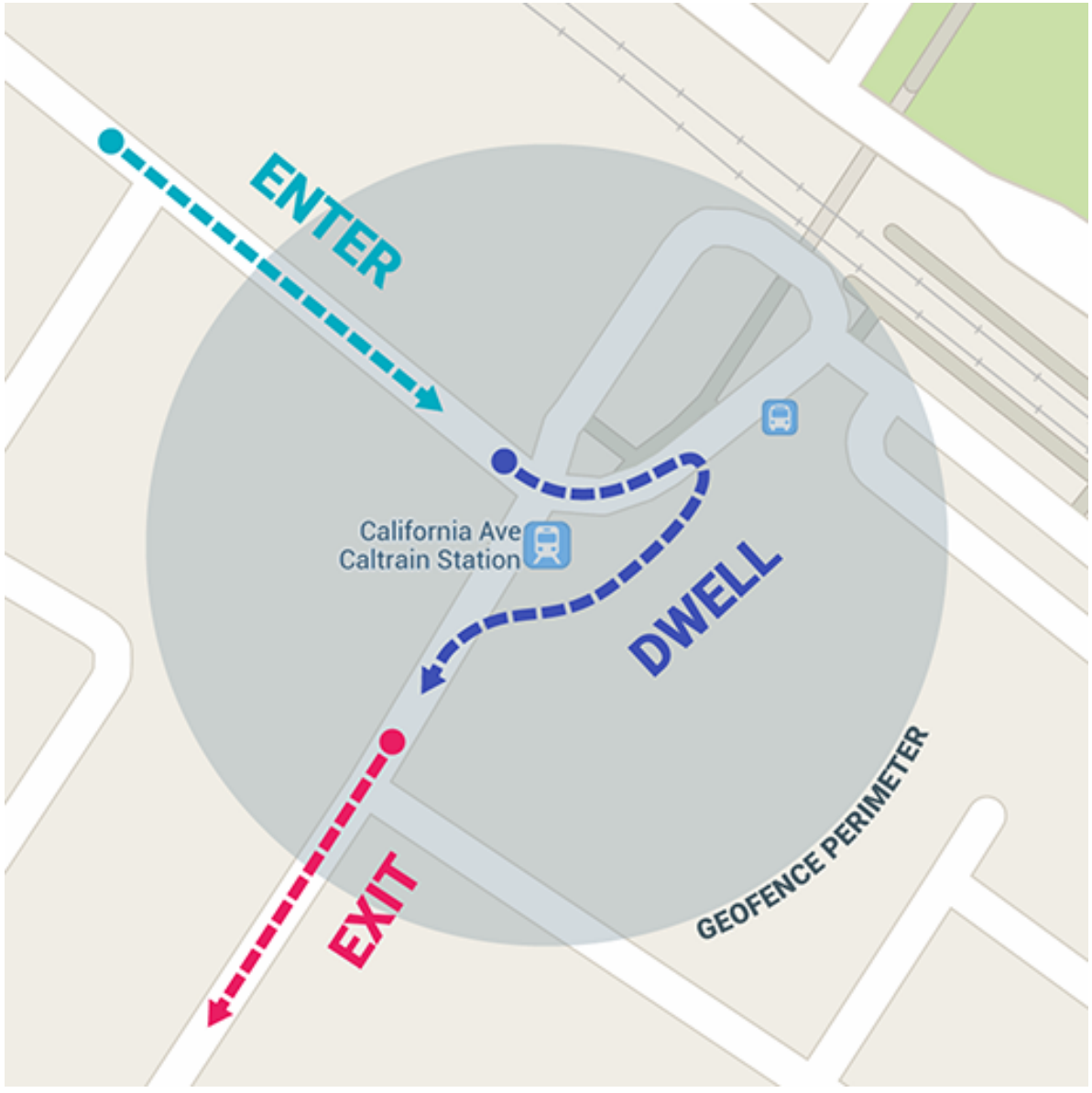
Berner Fachhochschule

April 5, 2019

Learning Objectives

- Android's GeoFence API
- What are geographic information systems
- How to use PostGIS:
  - Installation
  - Data import
  - Queries

Android's Geofence API



Limitations

- Up to 100 Geofences per app
- Region is circular
- Precision depends on circumstances (WiFi, GPS, etc.)

Main classes involved

IntentService Subclass implements functions called when GeoFence is triggered

GeoFence Specifies location and transition we care about, created via `GeoFence.Builder`

OnAddGeofenceResultListener Success adding `GeoFence`

ConnectionCallbacks Notified when connection to location services is up and we can add GeoFences

Permissions required

```
<uses-permission android:name=
  "android.permission.ACCESS_FINE_LOCATION"/>

<application
   android:allowBackup="true">
   ...
   <meta-data
    android:name="com.google.android.gms.version"
    android:value="4242000">
   <service android:name=
    ".GeofenceTransitionsIntentService"/>
<application/>
```

Check for Google Play

```
private boolean servicesConnected() {
  int resultCode
    = GooglePlayServicesUtil
      .isGooglePlayServicesAvailable(this);
  if (ConnectionResult.SUCCESS == resultCode) {
    // Continue with GeoFence
    return true;
  } else {
    // Handle the error
    return false;
  }
}
```

Creating a Geofencing client

```
private GeofencingClient mGeofencingClient;

// ...
mGeofencingClient
  = LocationServices.getGeofencingClient(this);
```

Create Geofence List

```
List<GeoFence> mGeofenceList = ...;
mGeofenceList.add(new Geofence.Builder()
   .setRequestId("my request name")
   .setCircularRegion(
           latitude,
           longitude,
           GEOFENCE_RADIUS_IN_METERS)
   .setExpirationDuration(GEOFENCE_EXPIRATION_IN_MILLISECONI
   .setTransitionTypes(Geofence.GEOFENCE_TRANSITION_ENTER |
           Geofence.GEOFENCE_TRANSITION_EXIT)
   .build());
```

Specify Geofence Request

```
private GeofencingRequest getGeofencingRequest() {
  GeofencingRequest.Builder builder
    = new GeofencingRequest.Builder();
  builder.setInitialTrigger(GeofencingRequest
    .INITIAL_TRIGGER_ENTER);
  builder.addGeofences(mGeofenceList);
  return builder.build();
}
```

Specify Intend to trigger

```
public class MainActivity extends AppCompatActivity {
  private PendingIntend mGeofencePendingIntent;
  private PendingIntent getGeofencePendingIntent() {
    if (mGeofencePendingIntent != null)
      return mGeofencePendingIntent;
    Intent intent
      = new Intent(this,
                   GeofenceTransitionsIntentService.class);
    mGeofencePendingIntent
      = PendingIntent.getService(this, 0, intent,
          PendingIntent.FLAG_UPDATE_CURRENT);
    return mGeofencePendingIntent;
  }
}
```

Handle event

In GeofrenceTransitionsIntentService extends IntentService:

```
@Override
protected void onHandleIntent(Intent intent) {
  if (LocationClient.hasError(intent)) {
    // Handle error
  } else {
    int transition = LocationClient.getGeofenceTransition
        (intent);
    switch (transition) {
    case Geofence.GEOFENCE_TRANSITION_ENTER:
    case Geofence.GEOFENCE_TRANSITION_EXIT:
      // action
    }
  }
}
```

Request Geofence activation

```
mGeofencingClient.addGeofences(getGeofencingRequest(),
                               getGeofencePendingIntent())
  .addOnSuccessListener(this,
                        new OnSuccessListener<Void>() {
     @Override
     public void onSuccess(Void aVoid) {
       // Geofences added
     }
  })
  .addOnFailureListener(this, new OnFailureListener() {
     @Override
     public void onFailure(@NonNull Exception e) {
       // Failed to add geofences
     }
  });
```

Deactivate Geofence

```
mGeofencingClient.removeGeofences(getGeofencePendingIntent(
  .addOnSuccessListener(this,
                        new OnSuccessListener<Void>() {
    @Override
    public void onSuccess(Void aVoid) {
      // Geofences removed
    }
  })
  .addOnFailureListener(this, new OnFailureListener() {
    @Override
    public void onFailure(@NonNull Exception e) {
      // Failed to remove geofences
    }
  });
```

Persistence

The app must re-register geofences if they're still needed after the following events:

- The device is rebooted.
- The app is uninstalled and re-installed.
- The app's data is cleared.
- Google Play services data is cleared.
- The app has received a `GEOFENCE_NOT_AVAILABLE` alert.[1]

[1] This typically happens after NLP (Android's Network Location Provider) is disabled.

Forground applications

An app is considered to be in the foreground if:

- It has a visible activity, whether the activity is started or paused.
- It has a foreground service.
- Another foreground app is connected to the app, either by binding to one of its services or by making use of one of its content providers.

If none of those conditions is true, the app is considered to be in the background.

Background limitations

As of Android 8.0, background applications are limited to “a few times each hour” in terms of how often they can obtain the user’s location!
This affects:

- Fused Location Provider
- Location Manager
- GeoFence API
- GNSS measurements
- Wi-Fi manager

Expected latencies for GeoFence

- 2–3 minutes if device is moving
- 6 minutes if device was stationary
- Forever if there is no WiFi/data service and/or GPS
- Since Android 4.3, WiFi "off" may not matter[2]

[2] Device my be configured to do "Wi-Fi scan only" which suffices for location.

Project (suggestion)

Create a background service that:

- Detects when user successfully (!) enables WLAN
- Creates a Geofence for the area
- Disables data service whenever WLAN-area is entered

Why not automatically enable WLAN as well?

Geographic Information Systems

Are systems designed to:

- capture
- store
- manipulate
- analyze
- manage and
- present

spatial or geographic data.

Databases

- store
- manipulate
- analyze
- manage

PostGIS

Postgres *extension* to:

- store
- manipulate
- analyze
- manage

spatial or geographic data.

Applications

- Public safety
- Asset inventory, land management
- Planning (water management, transportation, engineering)
- Network management

Spatial queries

"Show me the region with the highest number of households where the average closest distance to any pizza parlor with a star ranking below 5 is greater than 10 kilometers." –PostGIS in Action

"Give me the number of houses within a two kilometer radius of the coastline." –PostGIS in Action

Spatial data types

Geometry
: the world is two dimensional

Geography
: lines and polygons are drawn on the earth's curved surface

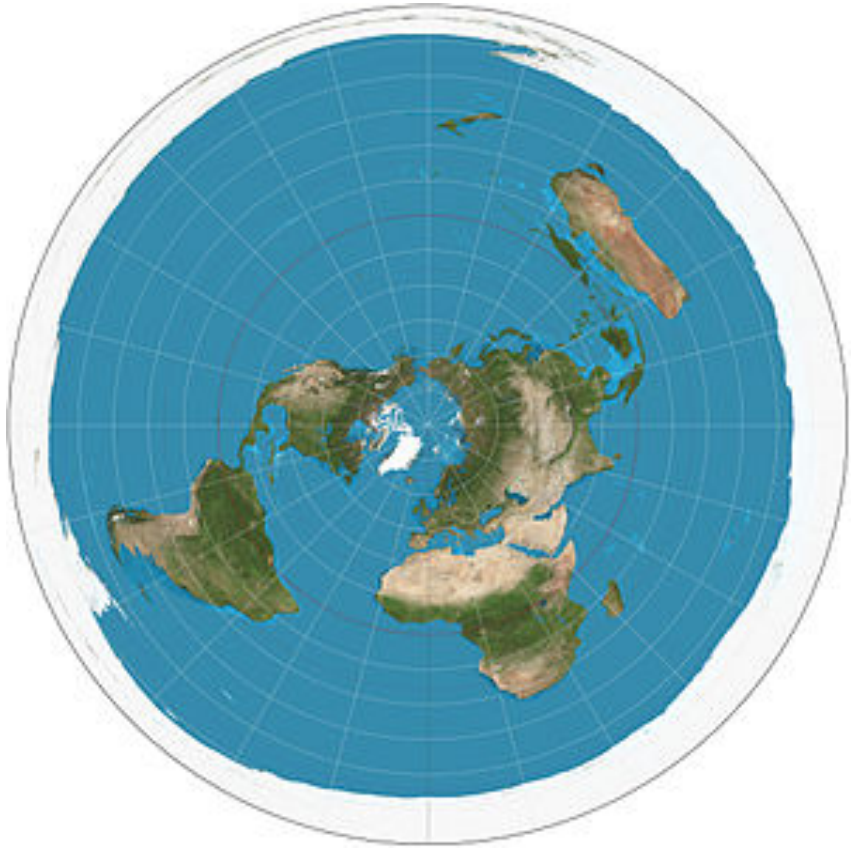
Raster
: models spaces as a grid of rectangular cells with associated data values

Topology
: models world as a network of connected nodes

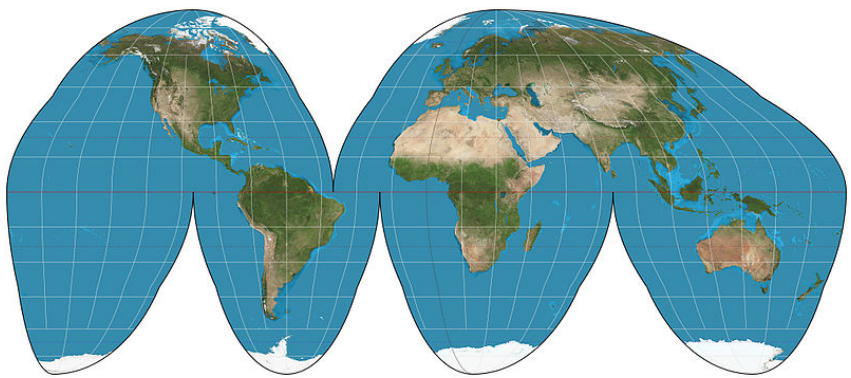
Geometry types

- Linestrings ("coastline")
- Polygons ("region")
- Points ("houses")

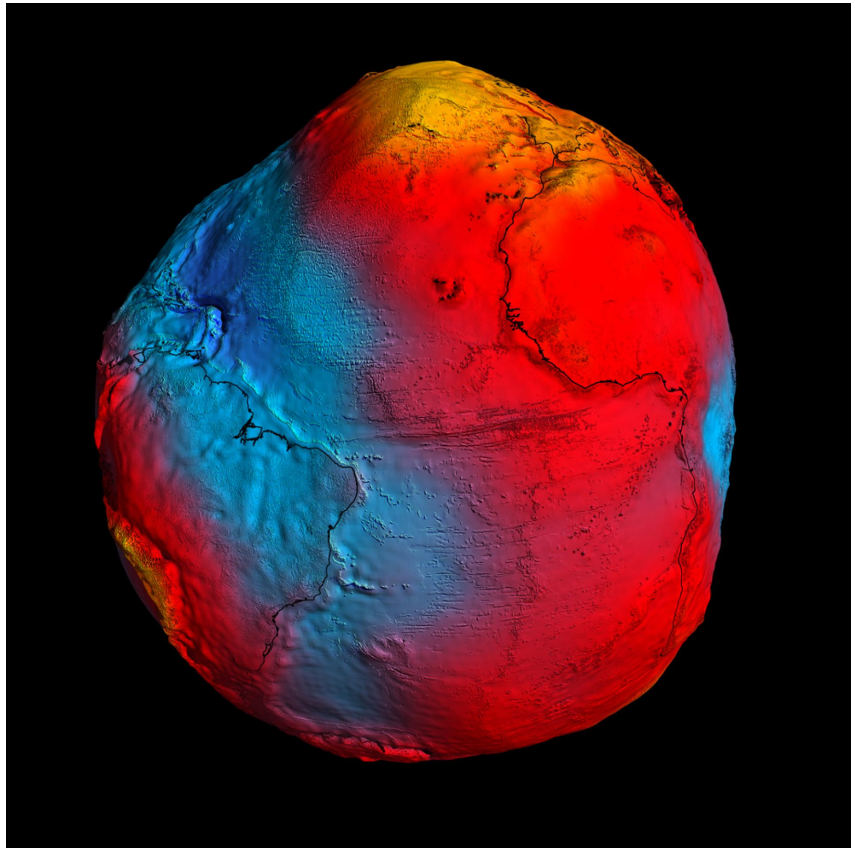
Azimuthal equidistant projection



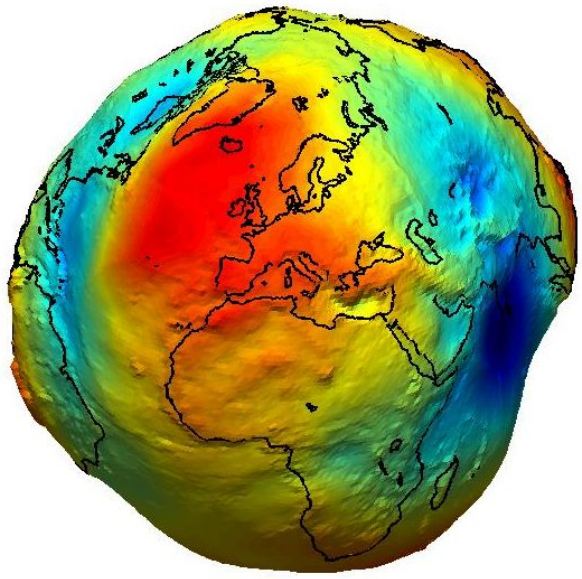
Goode homolosine projection



Gravity

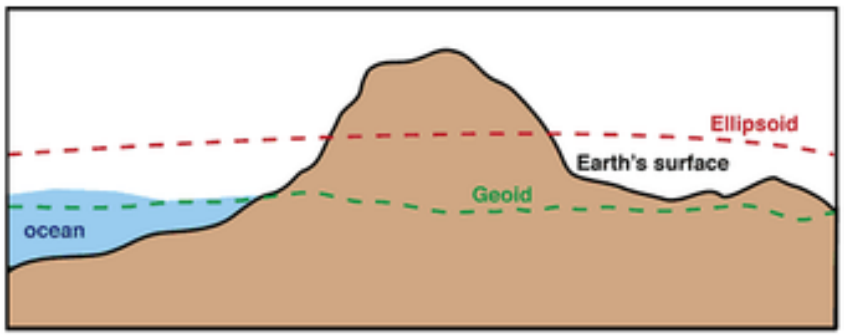


Shape



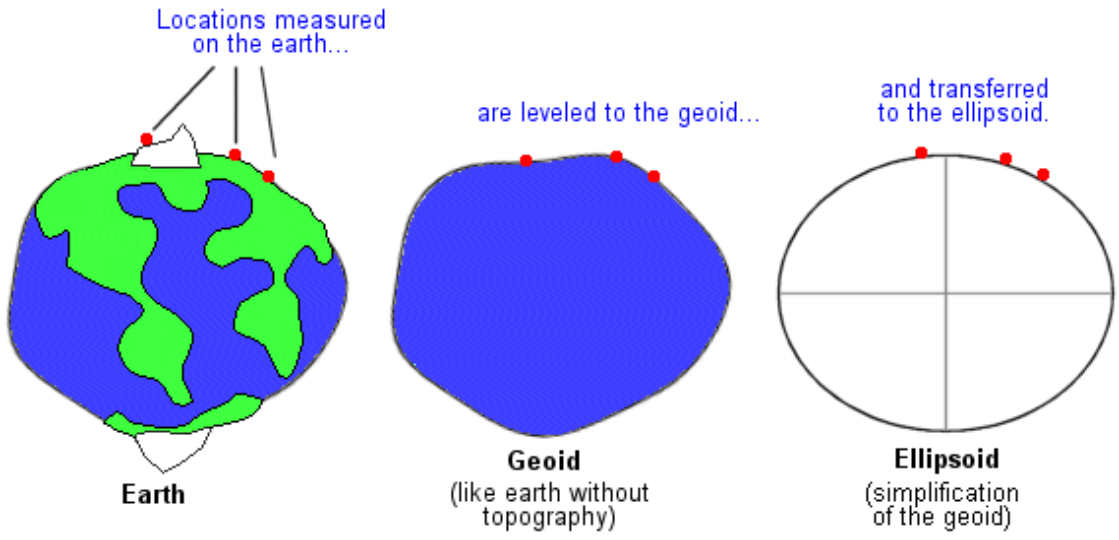
The Geoid

The geoid is an equipotential, or level, surface of the earth's gravity field. Imagine the oceans are allowed to settle under the influence of gravity (and rotation) only and are not subject to tidal or atmospheric forces. Tunnels are also used to connect the oceans so that the water can move freely.



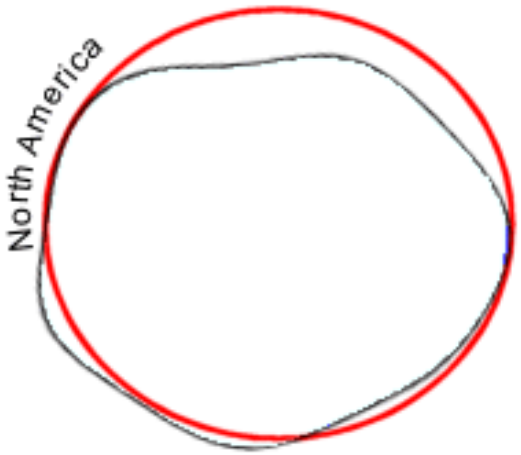
The geoid is approximately equal to mean sea level.

Fitting

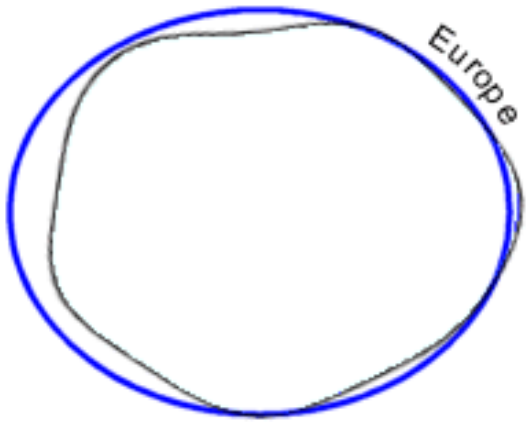

Locations measured
on the earth...
Earth
are leveled to the geoid...
Geoid
(like earth without
topography)
and transferred
to the ellipsoid.
Ellipsoid
(simplification
of the geoid)

Localization

A geographic coordinate system needs a way to align the spheroid being used to the surface of the earth for the region being studied. For this purpose, a geographic coordinate system uses a *datum*. A datum specifies which spheroid you are using as your earth model and at which exact location (a single point) you are aligning that spheroid to the earth's surface.



The red ellipsoid fits the geoid well in North America.



The blue ellipsoid fits the geoid well in Europe.

Installation

```
# apt-get install postgis
# su - postgres
postgres@localhost: $ createuser -s -O $USER
postgres@localhost: $ createdb -O $USER gis
postgres@localhost: $ psql gis
gis=> CREATE EXTENSION postgis;

$ psql gis # From here on everything as $USER
gis=> SELECT postgis_full_version();
```

Create schema for locations

```
CREATE SCHEMA bfh;
CREATE TABLE bfh.locations (
  id SERIAL PRIMARY KEY,
  name VARCHAR (300) NOT NULL,
  geom GEOMETRY(POINT,4326)
);
CREATE INDEX idx_loc_geo
  ON bfh.locations
  USING gist(geom);
```

SRID 4326 corresponds to World Geodetic System (WGS) 1984 data.

Create schema for roads

```
CREATE TABLE bfh.roads (
  id SERIAL PRIMARY KEY,
  label VARCHAR(120) NOT NULL,
  geom GEOMETRY(multilinestring,4326)
);
CREATE INDEX idx_roads_geo
  ON bfh.roads
  USING gist(geom);
```

Import data

```
$ shp2pgsql points.shp | head -n 10
SET CLIENT_ENCODING TO UTF8;
SET STANDARD_CONFORMING_STRINGS TO ON;
BEGIN;
CREATE TABLE "points" (gid serial,
"osm_id" float8,
"name" varchar(48),
"type" varchar(16));
ALTER TABLE "points" ADD PRIMARY KEY (gid);
SELECT AddGeometryColumn('','points','geom','0','POINT',2);
INSERT INTO "points" ("osm_id","name","type",geom) VALUES
  ('280587','Figuiers','bus_stop',
   '01010000003F8BA548BE721A4089CD6CFC2E424740');
```

Import data (for real)

```
$ shp2pgsql -s 4326 points.shp | psql gis 2> /dev/null
```

Convert data

```
INSERT INTO bfh.locations (name,geom)
SELECT
  name,
  geom
FROM points
WHERE name IS NOT NULL;
DROP TABLE points;
SELECT COUNT(*) FROM bfh.locations;
```

The result should be 16700.

Import road data

```
$ shp2pgsql -s 4326 roads.shp | psql gis
```

(may take a few minutes).

Convert road data

```
INSERT INTO bfh.roads (label,geom)
SELECT
  name AS label,
  geom
FROM roads
WHERE name IS NOT NULL;
DROP TABLE roads;
SELECT COUNT(*) FROM bfh.roads;
```

The result should be 78006.

Query data

```
# Determine road location(s)
SELECT label,ST_AsText(geom)
  FROM bfh.roads LIMIT 3;

# Determine total length of roads in meters
SELECT SUM(ST_Length(ST_Transform(geom,21781)))
  FROM bfh.roads;
```

EPSG[3] 21781 is die Schweizer Projektion.

[3] European Petroleum Survey Group

Query data

```
# Find location of interest furthest from any road
SELECT name, ST_AsText(loc.geom)
  FROM
    bfh.locations AS points, bfh.roads AS lines
  ORDER BY ST_Distance(points.geom,lines.geom) DESC
  LIMIT 1; # (30 minutes runtime)
# Find all locations of interest within 1 m of any road
SELECT loc.name, ST_AsText(loc.geom)
  FROM
    bfh.locations AS loc
  INNER JOIN
    bfh.roads AS road
  ON ST_DWithin(ST_Transform(road.geom,21781),
                ST_Transform(loc.geom,21781),
                1); # (2h runtime, 1861 rows)
```

Do not forget to get a coffee!

Project suggestions

- Implement application to collect your own movement data
- Store location (GPS + CID) and time data first as CSV
- Export result from mobile, import into PostGIS
- Overlay movement data with maps
- Determine mode of transport (public transit schedule, movement speed)
- Compute CO_2 emissions, calories burnt, ...

Acknowlegements

This presentation used material from:

- `https://developer.android.com/training/location/geofencing.html`
- `https://github.com/marimiyachi/geofence_example`