Creating a Simplified Large Scale 3D Model in PostGIS for Noise-calculations,

Based on Lidar and Cadastral Data

tom.van.tilburg@geodan.nl
WHY?
X 10 million
Software
- PDAL
- PGPOINTCLOUD
- SFCGAL
- POSTGIS
- GDAL
- GRASS

Postgres extensions:
- PLV8
- PLpython

in the python extension:
- numpy
- sklearn
- do it yourself algorithms
CREATE OR REPLACE FUNCTION noisemodel.dbscan3d(points text, eps double precision, minpoints integer)
RETURNS SETOF float[] AS
$$
from sklearn.cluster import DBSCAN
import numpy as np
def dbscan3d(points, eps=0.5, min_samples=5, metric='manhattan',
             algorithm='auto', leaf_size=30, p=None, n_jobs=1):
    geom = points[:,:3]
    db = DBSCAN(eps, min_samples, metric, algorithm,
                 leaf_size, p, n_jobs)
    db.fit(geom)
    labels = np.zeros((len(points), 5))
    labels[:, :4] = points
    labels[:, 4] = db.labels_
    return labels

arrpoints = np.array(eval(points))
return dbscan3d(arrpoints, eps, minpoints)
$$
LANGUAGE plpythonu;
Where are we?
LOD1.3 for NetherLands

Run time:
5 minutes / km2

Amsterdam = 20 hours
Where do we go?
LOD2.2

- Gable roofs
- Dormers
- Porches
Conclusion

A geospatial database is suitable for large scale 3D processing