Free and Open Source Software for Geospatial (FOSS4G) Conference Proceedings

Volume 16 *Bonn, Germany* Article 16

2016

How to cross a square?

Jakob Miksch
Department of Geography, Heidelberg University, Germany

Stefan Hahmann
Department of Geography, Heidelberg University, Germany

How to cross a square?

Jakob Miksch, Stefan Hahmann

Department of Geography, Heidelberg University, Germany – jakob.miksch@stud.sbg.ac.at, stefan.hahmann@geog.uni-heidelberg.de

ABSTRACT:

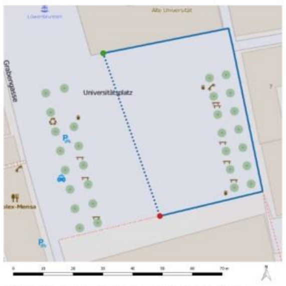
The poster compares algorithms how squares can be integrated into a routing engine. A case study showed that most algorithms increase the processing time and the created count of edges significantly.

How to Cross a Square?

Miksch, J. [1,2], Hahmann, S. [1]
[1] Chair of GIScience, Department of Geography, Heidelberg University, Germany
[2] Z_GIS – Department of Geoinformatics, Salzburg University, Austria

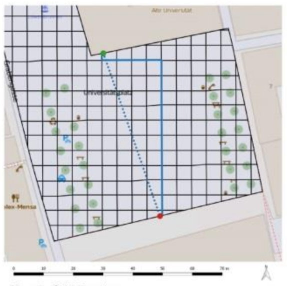
1 Introduction

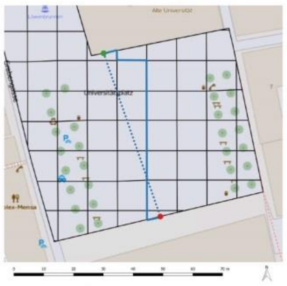
Routing engines have problems to find the shortest path for pedestrians when a square is on the way. The calculated route is often longer than necessary (Map 1). This has a simple reason: Just the boundaries of the squares are integrated into the street network. The interior area of the square is not taken into account.

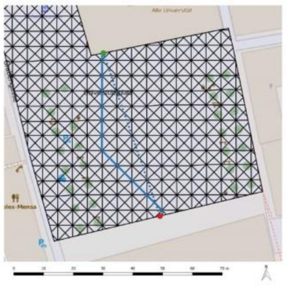

Map 1: shortest path vs. suggested path by OpenRouteService [1]

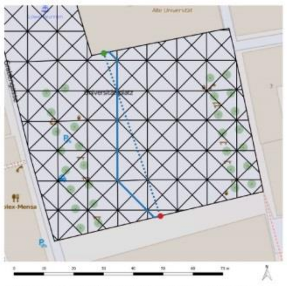
2 Method

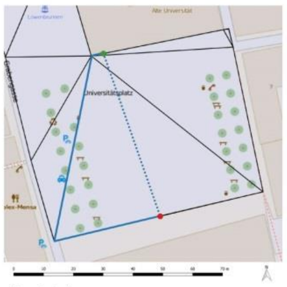
Creating virtual ways on the square makes it possible to integrate the square into the street network. There are many algorithms for creating these virtual ways. Map 2 until 8 show some examples. The location is the University Square in Heidelberg, Germany. The algorithms have been created with JTS [2]. The blue line shows the new suggested path. The dotted line shows the shortest path.

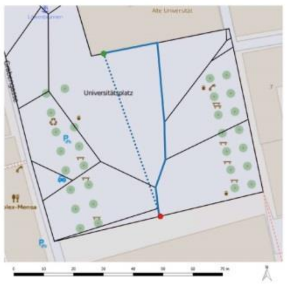

Map 2: Grid 5 meter


Map 3: Grid 10 meter


Map 4: Spider grid with diagonals - 5 meter


Map 5: Spider grid with diagonals - 10 meter


Map 6: Delaunay


Map 7: Voronoi

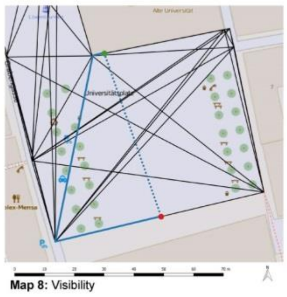
3 Comparison of Algorithms

Map 2 and 3 show a grid layer with 5m / 10m grid size. Map 4 and 5 show an extension where two diagonals are added. The algorithms of map 6,7 and 8 take the connection points to the remaining street network into account.

The suggested path of the grids and the "Voronoi" is shorter than in map 1, but has still an unnatural shape. The suggested ways of Delaunay and Visibility are quite long. The reason for it is that there are no field connection points around the red destination node.

In order to compare these algorithms on a larger scale, we calculated them for all accessible squares in Baden-Württemberg using OpenStreetMap [3] data. The processing time and the count of additional edges were compared with a street network created by GraphHopper [4].

The diagram below shows that both parameters become increased significantly by most of the algorithms. So far "Grid 10" seems to be the best compromise. It delivers both an acceptable routing result and does not increase the processing time and the network size to much.


Map 8: Visibility

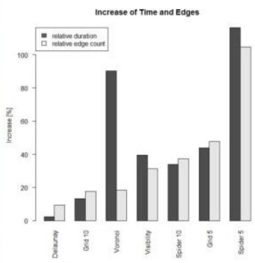


Diagram: Comparison

4 Further Work

This work has done a pre-assessment. In the next steps, the algorithms will be improved in order to make them faster. Moreover some features could be added: The grid could be oriented along the main axis of the square. "Visibility", "Delaunay" and "Voronoi" could be improved by adding more connection points. The final goal is to implement one or many algorithms into a routing engine.

References
[1] www.openrouteservice.org
[2] www.vividsolutions.com/jts/JTSHome.htm
[3] www.openstreetmap.org
[4] www.graphhopper.com
The maps use OpenStreetMap mapnik style as background.

Contact
jakob.miksch@stud.sbg.ac.at.
hahmann@uni-heidelberg.de

A part of the research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 612096 (CAP4Access)