UNIVERSITÄT TRIER

Jan Weymeirsch
Hanna Dieckmann
Ralf Münnich

Construction of a Georeferenced House Data Set for the City of Trier within the MikroSim Project

Research Papers in Economics
No. 9/24

Construction of a Georeferenced House Data Set for the City of Trier within the MikroSim Project

Jan Weymeirsch, Hanna Dieckmann, and Ralf Münnich

30th September 2024

Abstract

Spatial dynamic microsimulations have distinct potential to operate and project populations on finely detailed geographic level. For this, a detailed building and dwellings data set is typically needed, such that inner-region migration flows and local population developments may be modelled accurately. There exists, however no comprehensive building register for Germany, especially none that is openly available to the research community and which accurately distinguishes buildings in regard to their usage as accommodation or as potential workplaces. The following work serves as a pilot study to create such a data set by combining several publicly available data sources for a single region in Germany. Its quality is evaluated by comparing it to both, aggregate information from German official statistics, and register data from the city of Trier.

1 Introduction and Motivation

The *Multi-sectoral Regional Microsimulation Model* (MikroSim) project aims at developing a dynamic microsimulation model of the German population at the level of persons and households. It is distinguished by the possibility to conduct regionally differentiated analyses on district and municipality level. While this suffices for some applications, more detailed geographical levels are required for example for transportation planning, the evaluation of housing policies or the provision of health services. Thus, a buildings data set that is projected alongside the georeferenced population is required.

This paper describes the creation of a building data set with location reference for 2021 and thus serves as a basis for georeferencing the base population in the MikroSim model. The house data set reflects the year 2021 (i.e. the year the data was gathered), as it is mainly based on data from sources which are continuously updated, such as OpenStreetMap (OSM). It is therefore a good fit to data sources that reflect the reference year for the upcoming German census 2022. The creation of the georeferenced house data set for the city of Trier is intended as a feasibility study.

This paper is structured as follows. Section 2 introduces the different data sources. The OSM information constitutes the core for the creation of the house data set. Furthermore, data from Rhineland-Palatinate's Web Mapping Service (WMS) portal as well as Light Detection and Ranging (Lidar) information from the area around the city of Trier and an appropriate Digital Elevation Model (DEM) are used for categorization of building information as well as approximations of building heights. Register data provided by the city of Trier is used as a validation step for the building classification. Supplemental information, such as school districts, are added for application examples. Section 3 describes the approach undergone to combine the different data sources introduced in the previous section and modelling dwelling space. The creation of a house data set within a microsimulation model as a basis for georeferencing the base population comprises two steps: The creation of a house data set as such, and the approximation of living space and dwellings per residential building. Section 5 discusses the possibility to automate these steps and points out data limitations.

2 Data

Data sets that contain information on the building floor plans and the building locations are suitable for the creation of a house data set. In this paper, we consider OSM data as the main basis. These data sets are supplemented by Lidar information as well as a DEM for the Trier area. In order to categorize buildings as residential, city plans of houses and buildings in Trier via the administration's WMS are used and added to the data set. Thus, it is possible to obtain a house register for relatively recent time spans that includes information about the floor area, building volume, and building type. Based on information from the Urban Atlas, the area of mixed-use buildings utilized for residential purposes is approximated and buildings that are located in residential areas but do not serve residential purposes are determined.

2.1 OpenStreetMap data

The OSM project aims at providing a free, editable map of the world. Anyone, including individuals, organizations, as well as businesses, can become a member of the OSM-Community and contribute to the project using a collaborative wiki editing software (OpenStreetMap Wiki contributors, 2019a). Thus, the coverage and quality of the data depends on the activity of the OSM-Community in the respective region as well as their experience (OpenStreetMap Wiki contributors, 2020c). OSM records more than six million registered users and 3.5 million map changes per day as of February 2020 (OpenStreetMap Wiki contributors, 2020b). A tagging guideline is provided in order to ensure consistency and thus facilitate further processing of the data. However, the members of the OSM-Community are free to use any attribute names (OpenStreetMap Wiki contributors, 2019c), often times resulting in inconsistent tags or partially missing information.

Legend: ■ OpenStreetMap buildings in the data set.

Figure 1: OSM buildings on satellite imagery by Esri (2021)

Among other aspects, the OSM data contain information about streets, water bodies, forest, and buildings (OpenStreetMap Wiki contributors, 2019c). The building key depicts the floor plan of a building (see Figure 1). According to OpenStreetMap Wiki contributors (2020a), the ground plan is recorded either by aerial photography, explorations from the street by making sketches and simple measurements, or by walking along the outer walls and recording the GPS. The minimum tagging requirement for a building is an indication, whether the object is a building or not. OSM contributors also have the option to specify the building type (e.g. hotel, flat) as well as the building height and the number of floors (OpenStreetMap Wiki contributors, 2019b, 2020a). However, most building objects in OSM appear to be tagged with the minimal amount of necessary information, such that around 80% of buildings in OSM are only tagged as buildings (`building = yes`) without further information on its usage (Topf and Topf, 2021). The OSM data content is licensed according to the Open Database Licence (ODbL) 1.0, granting extensive right to use, change, and distribute the data as long as contributors (such as the OpenStreetMap community and others) are mentioned. The output data therefore has to be licensed under the same terms (OpenStreetMap Wiki contributors, 2019a). OSM is supported by fund-raising from the OpenStreetMap Foundation (OSMF), which primarily manages the server infrastructure that hosts the OSM project services and online resources (OpenStreetMap Wiki contributors, 2019a).

To generate a solid basis of the building data set, the OSM information is retrieved directly through the project's Application Programming Interface (API) via a custom function for the `GNU R` statistics programming language (Weymeirsch, 2021a). Both, OSM Nominatim and OSM Overpass services, are used to retrieve different types of information. The Nominatim service, on the one hand, can be used to retrieve geocoding information on the full, worldwide OSM data set, including searches for area names, coordinate boundaries, and (reverse) geocoding of given objects and locations (OpenStreetMap Wiki contributors, 2021a). Overpass,

on the other hand, may be used to retrieve data and Points of Interest (POIs) on any objects in the database, given a set of input criteria, such as a bounding box of coordinates, object type, including `ways`, `nodes`, and `relations` (OpenStreetMap Wiki contributors, 2021b).

2.2 City Plan Information

A detailed plan of cities and buildings in Rhineland-Palatinate is provided by the Office for Geodesy and Geoinformation Rhineland-Palatinate (2021b) in the form of an online map viewer. The viewer accesses information on the administration's public data server, using the open WMS specification. A custom framework for `GNU R` is used to gather information from the LVermGeo's WMS server (Weymeirsch, 2021b). The data is provided as image files with additional geographic information in the GeoTIFF format as specified by the Open Geospatial Consortium (2019).

For the entire area of Trier, 26 high resolution (5000×5000 `px`) image files, each holding 72 megabytes of lossless raster image information, are downloaded, spanning over a distance of 0.07° from west to east and north to south each. The images depict a high contrast area plan with information on buildings' positions as well as building type (see Figure 2). Residential buildings are colour coded[1] dark grey, industrial buildings as light grey, and administrative buildings as red.

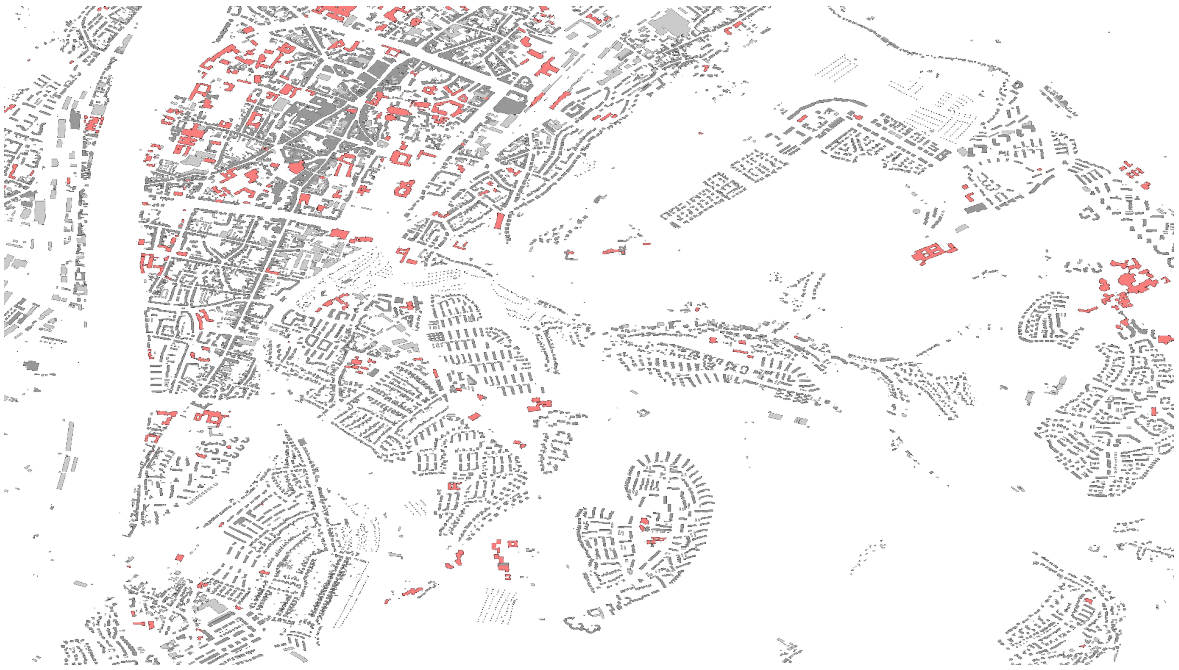


Figure 2: Property register data from the LVermGeo WMS server.

[1] RGB code for residential: `#989898`, industrial: `#c5c5c5`, administrative buildings: `#fd807e`.

2.3 Land usage information

OSM and the city plan data serve as a basis for the creation of the house data set. However, the city plan identifies individual buildings as either residential or industrial, whereas in reality buildings could be both. This is particularly true for the inner city area of Trier, where many buildings have shops on the ground floor and flats on the upper floors. Additional information about the dominant usage category of a certain area can be used to specify assumptions about the buildings' utilization.

The European Urban Atlas 2018 from the Copernicus Programme is used as supplementary data as it offers a high-resolution land map of urban areas for a fairly recent time span (Copernicus Programme, 2020a). The data set was chosen despite the reference year 2018, as we are not aware of any alternative freely accessible, high-resolution data set for the required area and information. Figure 3 shows the different area categories for the city of Trier, where data from the Urban Atlas is used. The Urban Atlas is a joint project of the Commission Directorate-General for Regional and Urban Policy and the Directorate-General for Enterprise and Industry within the frame of the EU Copernicus programme, with the support of the European Space Agency and the European Environment Agency. The Urban Atlas is based on multispectral SPOT 5 & 6 and Formsat-2 pan-sharpened imagery with a 2 to 2.5 meter spatial resolution. Information on the degree of sealing is derived from the high resolution layer imperviousness and functional information is extracted from ancillary data sources (Copernicus Programme, 2020a).

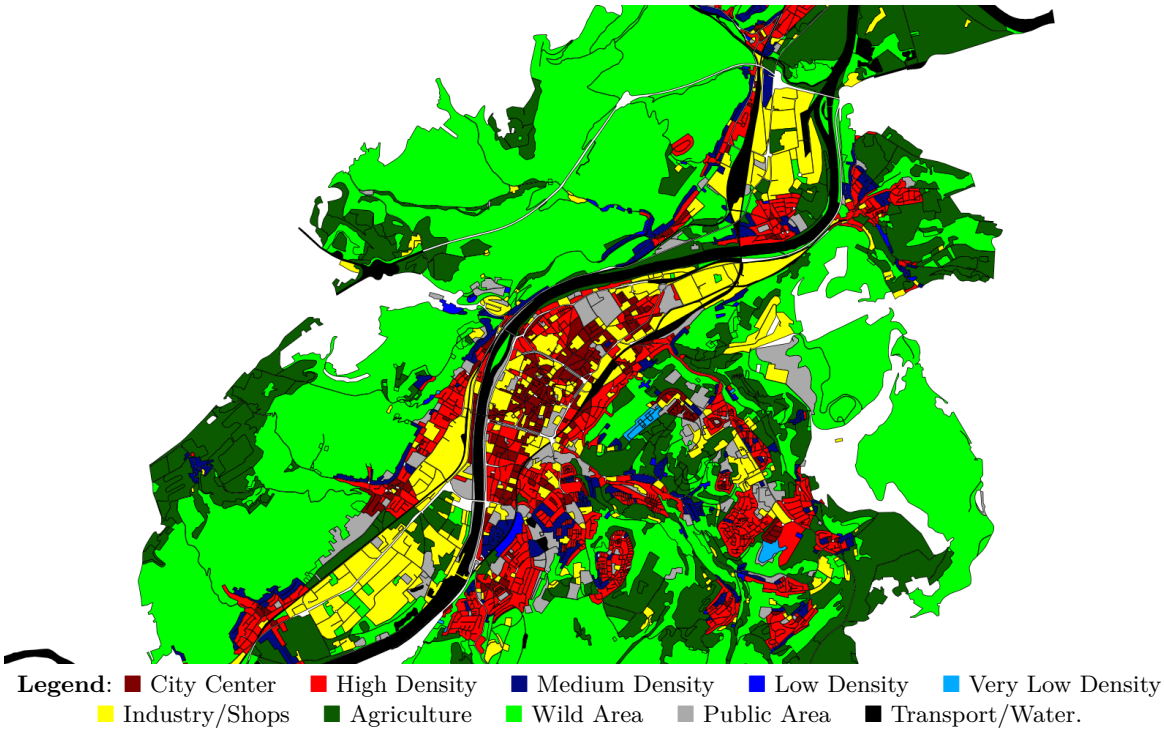


Figure 3: Building and Area Categories in Trier from European Union (2016)

The Urban Atlas 2018 includes 785 Functional Urban Areas, which were defined by Eurostat in cooperation with the respective countries (Copernicus Programme, 2020b), and covers 39 countries of the European Economic Area. The land use map classifies the surfaces into 27 different categories (Copernicus Programme, 2020a). The category *urban fabric* (i.e. continuous and discontinuous urban fabric as well as isolated structures) is defined as urban areas with dominant residential use or inner-city areas with central business district and residential use (European Union, 2016, p. 11). The minimum mapping width is 10 meters (Copernicus Programme, 2020a).

2.4 Elevation and Lidar Information

Neither the data from LVermGeo nor the OSM objects contain comprehensive and reliable information about the building height. While there is some information for concrete building heights or number of floors in the OSM data, such identifiers are rarely added to the objects (for the city of Trier, around 5% of buildings have the `building:height` or `building:levels` variables). As this information is crucial for approximation, a combination of a DEM and Lidar cloud point data from Office for Geodesy and Geoinformation Rhineland-Palatinate (2021a, 2017) are used to estimate the elevation of buildings. Based on the volume, the building type, as well as the living area, the number of dwellings per building can be approximated.

Lidar is an optical remote-sensing technique. It uses laser light to measure the distance between the target and the Lidar sensor and thus produces highly accurate `x, y, z` measurements, as illustrated in Figure 4 (Esri, 2019c). In order to estimate the building height, typically, the Lidar point collection is converted into a raster elevation surface (Esri, 2019b). The elevation model thus includes tree canopies and buildings (Esri, 2019a). The raster cell size must be small enough to clearly define what is part of the building and what lies outside the building. By sampling at random locations, the height of the building at the respective location are determined. In order to estimate a single height for each building, this information is summarized by taking the mean of the elevation at the random locations (Esri, 2019b). However, contrary to the typical *raster approach*, for the underlying OSM base data with precise polygons of the outer border for each building, it makes sense to use the raw point measures instead of creating a less detailed elevation surface grid.

As Lidar information provides the absolute height of a certain area or point including the terrain underneath an object or building, an appropriate reference is required. For this, a DEM in the form of an elevation grid for the Trier area is used. The difference between measured heights in the laser data and the elevation model can be used as approximation for the respective building's height. The DEM is published by the LVermGeo as OpenData grid-cells with 25 meter side length.

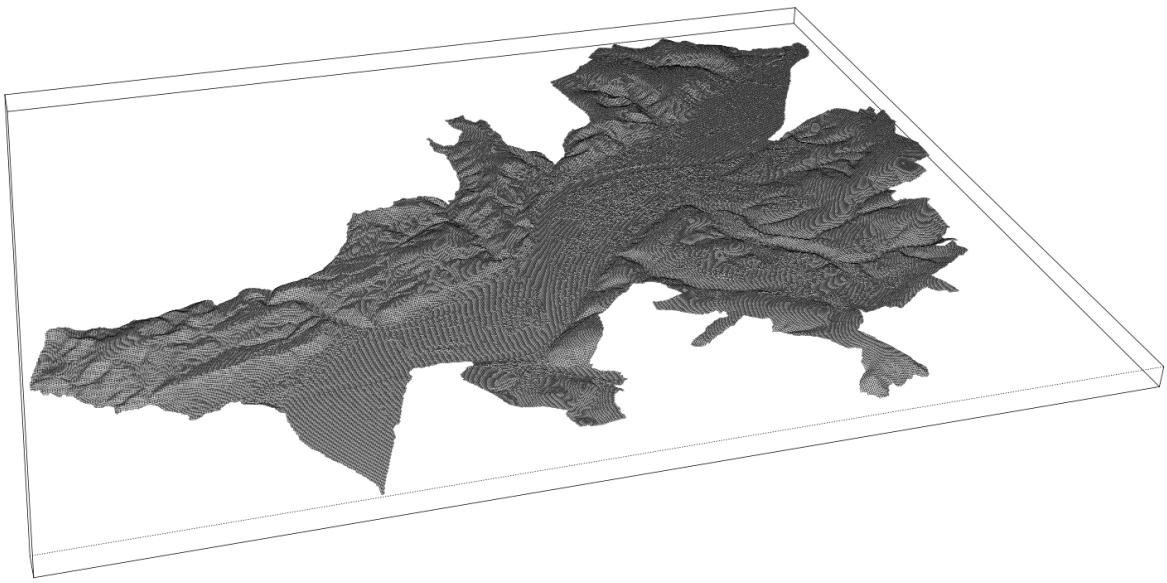


Figure 4: Digital Surface Model of Trier using Lidar data.

2.5 Supplemental Data

In order to use the data for further analyses and applications, additional information is added to the buildings corpus. This type of information is not necessarily required for the evaluation of the data set itself, but instead can show some possible application fields, such as within-city migration analyses, city planning or ecological disaster prevention.

Using OSM Nominatim, polygons from districts and villages belonging to Trier are gathered and every building in the data set is assigned to the respective region (OpenStreetMap Wiki contributors, 2021a; Weymeirsch, 2021a). Similarly to the addition of the city plan information explained in Section 2.2, polygons of school districts in the city of Trier are gathered via the city's WMS server (Office for Land Management and Geoinformation Trier, 2021; Weymeirsch, 2021b).

Furthermore, the city of Trier offers an online viewer for flood maps with different water levels of the river Mosel between 9.00 meters and 12.70 meters, indicating which parts of the city are potentially threatened by floodwater as well as rising levels of groundwater (Office for the Environment Rhineland-Palatinate & City of Trier, 2019). The data is not accessible as shape files and can only be exported manually in the GeoTIFF format, discussed in Section 2.2 already. As can be observed in Figure 5, the colour legend is non-linear, such that the deeper water levels have greater steps between the water heights ($> 750 - 100$ cm) than the shallow water levels ($100 - 10$ cm). Additional data is available on locations where groundwater rises substantially, to a degree where sub-terrain constructions might be flooded. According groundwater map images only contain binary colour codes.

Since no API is available for the online map viewer, the data is gathered automatically using a combination of a simple script written in `GNU R`, an ordinary open source web browser as

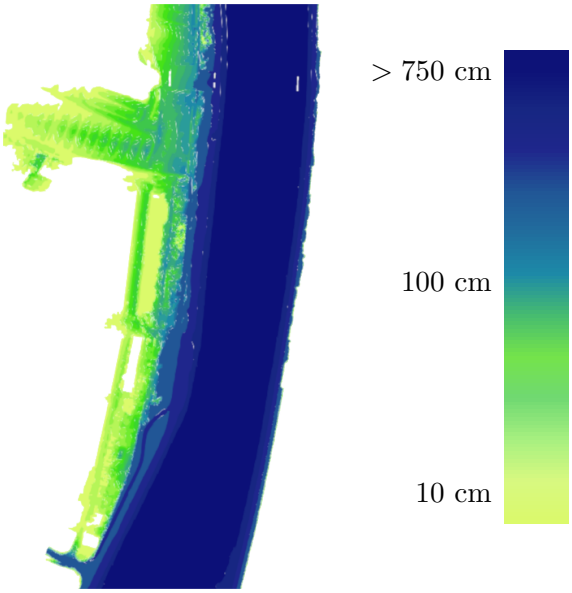


Figure 5: Exemplary section of flood map provided by Office for the Environment Rhineland-Palatinate & City of Trier (2019)

well as a few lines of Javascript (JS) code. In detail, the URLs with according parameters like the coordinates of a single section on the map, the zoom-level as well as the layer (i.e. the water level scenario of the Mosel) and its layer opacity are generated by the R-script. Using the map's coordinates, Trier is divided into 270 square sections, which need to be captured to gather information for the entire area. Five different water level scenarios are used both for the water depth and the groundwater areas, resulting in ten map-layers and 2,700 single map viewer URLs in total.

The R-script then successively calls the URLs via the system's web browser, which is configured to execute JS code when the city's geoportal is loaded. After waiting for a brief period to let the map viewer load completely, the code simulates mouse clicks on the layer-widget to choose the option without any map-background, such that the output images do not contain any unnecessary information. It then opens the 'Export Image' window of the map viewer in order to download the current viewport as GeoTIFF and closes the browser tab just before the R-script iterates onto the next URL. The procedure takes 30 seconds for each image, resulting in almost 23 hours for all sections and map layers. A more technical explanation can be found in Appendix A.

2.6 Benchmark Data

In order to evaluate the quality of the categorisation procedure, register data provided by the city of Trier are used. In particular, address information of all households within the city for the years 2011 until 2022 is available. For the comparison, only unique address strings (`street name, house number`) across all years were used, as they do not contain any information deemed sensitive. This list of unique addresses only contains the information,

that at least one household lived at a specific address during the 12-year time period. Neither information on how many households and persons lived at an address, nor during which timespan were used in this application.

The use of information that reaches back up until 2011 introduces the drawback, that the usage type could have been changed in the meantime (e.g. dwelling space up until a certain year and strictly commercial ever since then). Therefore, it is likely that the benchmark slightly over-states the number of residential buildings in 2021. Contrary to this, the advantage of using several rather than a single year lies within the potential to account for vacant residential buildings, which can be substantial for many areas in Germany. The census 2022 results report 3,540 vacant dwellings in Trier (6% of all dwellings), of which almost half are vacant for more than twelve months (Statistical Offices of the Federation and the Länder, 2024). If benchmark information of only a single year had been used, many buildings that were vacant during the relevant year incorrectly could have been deemed non-residential in the evaluation phase. could/would have incorrectly been deemed

3 Methodological Approach

The creation of a house data set as a basis for georeferencing the base population in MikroSim consists of two steps: First, a data set for the year 2021 is compiled. Second, the living space is estimated and dwellings are assigned to the residential buildings. Optionally, the data can then be updated with additional information and be released afterwards.

3.1 Updating OSM with the city plan raster images

Legend: Residential Industrial Administrative/Other

Figure 6: OSM buildings with city plan information on satellite imagery by Esri (2021)

Within the first step, all data sources have to be harmonised and compiled to a single, coherent data set. In order to categorize the different building types into `residential`, `industrial`, and `administrative` buildings, the previously downloaded city plan image files are processed by rastering them and *translating* their colour information into a data matrix of 5000 columns and 5000 rows (i.e. one cell for each pixel in the image). Each pixel's RGB colour-code is used as value in their according matrix cell. Since the GeoTIFF image files do not provide precise spatial information on each pixel's corresponding coordinates, but rather on locations of the images' outer border, a precise mapping of pixels to real-world locations is non-trivial.

Due to the earth's curvature and the fact that same-sided image files are downloaded, the images are slightly distorted. This results in the distance between two neighbouring pixels of any image represented to be about $0.000014°$ ($0.07°/5000px$), regardless of which location on the globe the pixels correspond to. Therefore, distances between pixels may only be specified here in terms of degrees, not in meters[2]. While the distortion could be problematic in larger scale applications, it should be minimal – if at all noticeable – within the area of Trier. The colour-matrix is therefore scaled according to the bounding latitudes and longitudes, setting the first column as lower latitude and the last column as upper latitude. Similarly, the first row is set as upper longitude and the last row as lower longitude, such that each pixel can be mapped directly to a specific coordinate (or more so, to a box with side-length $0.000014°$).

Next, for each building in the OSM data set defined by a vector or polygon of its outer border, the city plan's matrix values that fall within this polygon are registered and the most occurring one is used as building type (i.e. `residential`, `industrial`, `administrative`). This procedure ensures that robust results are obtained even in cases where the OSM object polygons do not coincide with the city plan's buildings, or where the pixel-mapping is imperfect. A comparison between Figure 6 and Figure 1 indicates which buildings are likely to be inhabited by Trier's population.

3.2 Approximating building volumes

As mentioned in Section 2.4, the approximation of building heights is two-fold and involves three different data sources. In a first step, the absolute heights of each building are computed by taking the median value of all Lidar points within the OSM building polygons. The median is used to circumvent imperfect fits between the Lidar information and the OSM object-polygons. The result is a single absolute height value for each building.

In a second step, the DEM25 elevation data are used as a reference for the height of the terrain. Here, the grid-cell's value in which the centroid (point of gravity) of each building is located is used. The terrain height of each building is subtracted from its absolute height, resulting in the buildings' approximate height. Figure 7 shows the approximate building

[2] Either way, this roughly translates to 1.5×1.5 square meters throughout Trier.

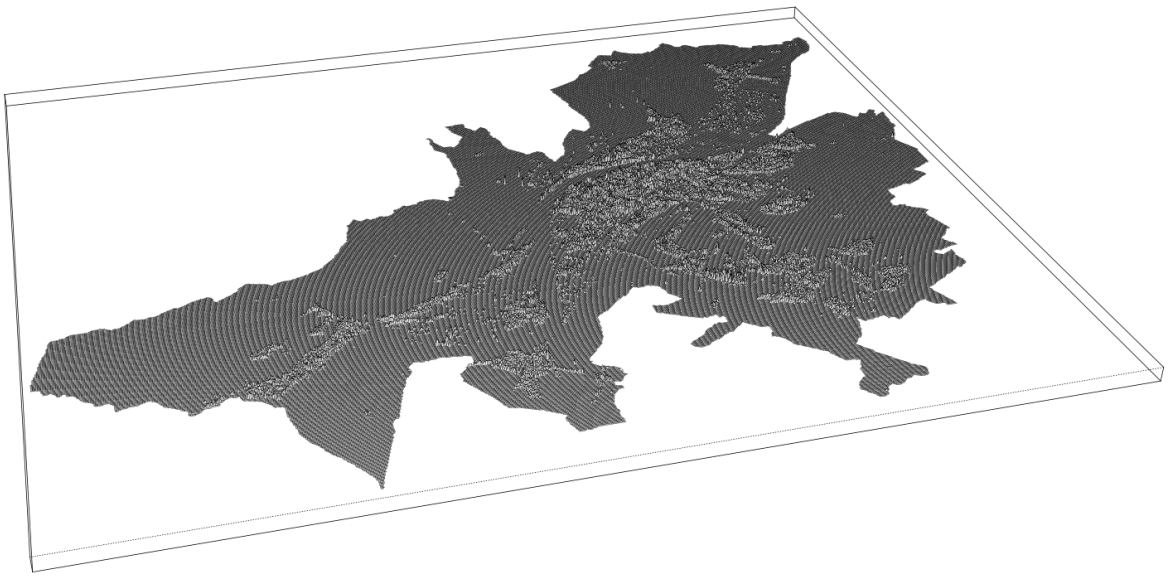


Figure 7: Differences between Digital Surface and Elevation Models.

heights, i.e. the difference between the absolute heights illustrated in Figure 4 and the DEM25 data.

The building volume and living space are crucial for georeferencing a population. In order to estimate the buildings' volumes, the lengths of their outer walls have to be determined first. Since the OSM base data set provides polygons of the floor plans for each building, the length can be easily translated to meters with the help of the inverse haversine, as described by Panigrahi (2014, pp. 212ff.):

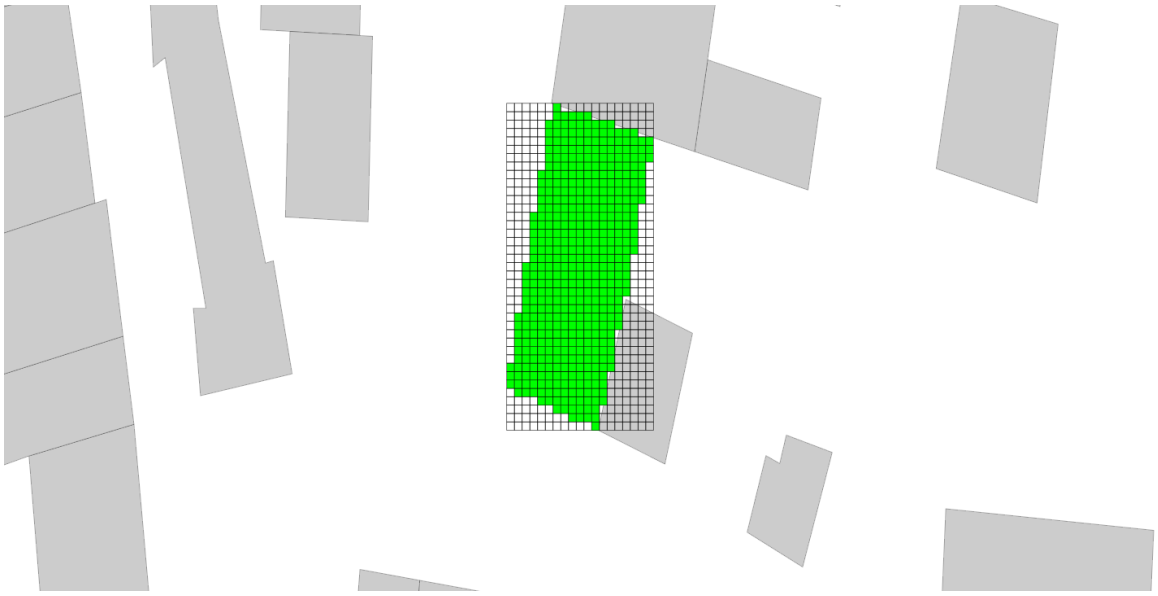


Figure 8: Simplified Approximation of Floor Area of a Building

$$d = 2r \arcsin \sqrt{\sin^2(\frac{\varphi_2 - \varphi_1}{2}) + \cos(\varphi_1) \times \cos(\varphi_2) \times \sin^2(\frac{\lambda_2 - \lambda_1}{2})} \quad (1)$$

with d being the distance in meters, φ_1 and φ_2 the latitudes and λ_1 and λ_2 the longitudes marking the respective starting and end points of a certain building's wall. Lastly, r is the earth's radius in meters (here approximated by $r := 6371000.8$).

Since most buildings are not perfectly square, calculating the exact floor area is computationally extensive and non-trivial to implement in code. Instead, the buildings' floor area is approximated by 10×10 cm^2 blocks, which is both, reasonably accurate and fast to compute. Figure 8 illustrates the procedure in simpler terms by means of an example building, where the floor area is approximated by a grid of small cells with standardized size.

Based on the buildings' approximated floor areas as well as their approximated median heights, volumes of the buildings can be calculated. For simplicity, buildings are assumed to have a uniform height, meaning that Level of Detail 1 objects are used as building models (U.S. General Services Administration, 2019). This is particularly relevant, as the building height determined by Lidar data takes into account the roofs of the buildings. However, rooftops are rarely part of a dwelling's living space. Under such considerations, living space within each building is assumed to be close to 85% of the buildings' total volume (including outer walls), as proposed by Loga and Diefenbach (2013) and Bricongne et al. (2019). The height of each storey is assumed to be between 2.8 and 3.2 meters, whereas Loga and Diefenbach (2013) use 2.5 meters as *ventilation reference room height*. It is important to note, that this value may only be used as approximation. Moreover, this value does not consider the thickness of the floor/ceiling of the building, which however is included in the case of this data and therefore needs to be accounted for. Based on the volume, the number of dwellings in a residential building can be estimated. The total number of dwellings at a 100 meter grid cell level is used as benchmark (Statistical offices of the Federation and the Länder, 2020).

3.3 Mixed use buildings and supplemental information

Neither the OSM base data set nor the LVermGeo's city plan are reliable sources for mixed-use buildings which have multiple purposes, like buildings with shops in the ground floor and flats in the upper floor(s). In order to model such buildings, the Urban Atlas is used. This highly detailed data set gives a rough idea about the dominant building categories of a certain area and thus helps to identify mixed-use buildings.

The following assumptions are used to model mixed-use buildings:

- Buildings marked as industrial within city centre areas are assumed to be non-residential only in the ground and first floor (if applicable).

- Buildings marked as residential within industry/shops areas are assumed to be non-residential in the ground and first floor (if applicable).

Finally, the additional information described in Section 2.5 such as the corresponding city- and school-districts are added to the building data set by assigning all buildings to the respective area, whose centroids are located within such regions.

Similarly, buildings are marked as endangered by floodings depending on their location relative to the flooding areas defined by the floodplain image data at different water levels. Each image file is rastered and loaded as matrices holding the corresponding colour values of each pixel, whereas the column and row indices are scaled according to the coordinates given by the metadata of the GeoTIFF image files. The buildings are then categorized by analysing the colour values of the pixels that fall within their region.

This procedure corresponds directly to the method used to update the OSM base data with the city plan raster images, described in Section 3.1. A four-level categorization is used here, marking buildings as `very endangered` (water levels > 100 cm), `endangered` (water levels < 100 cm & > 0 cm), `affected by groundwater` or `not endangered` for the five different water level scenarios.

4 Evaluation

Register data from the city of Trier for the years 2011 until 2022 are used to evaluate the building corpus and categorization quality (City of Trier, 2023). The highly confidential register data includes the addresses of individuals living in Trier, i.e. the street name and

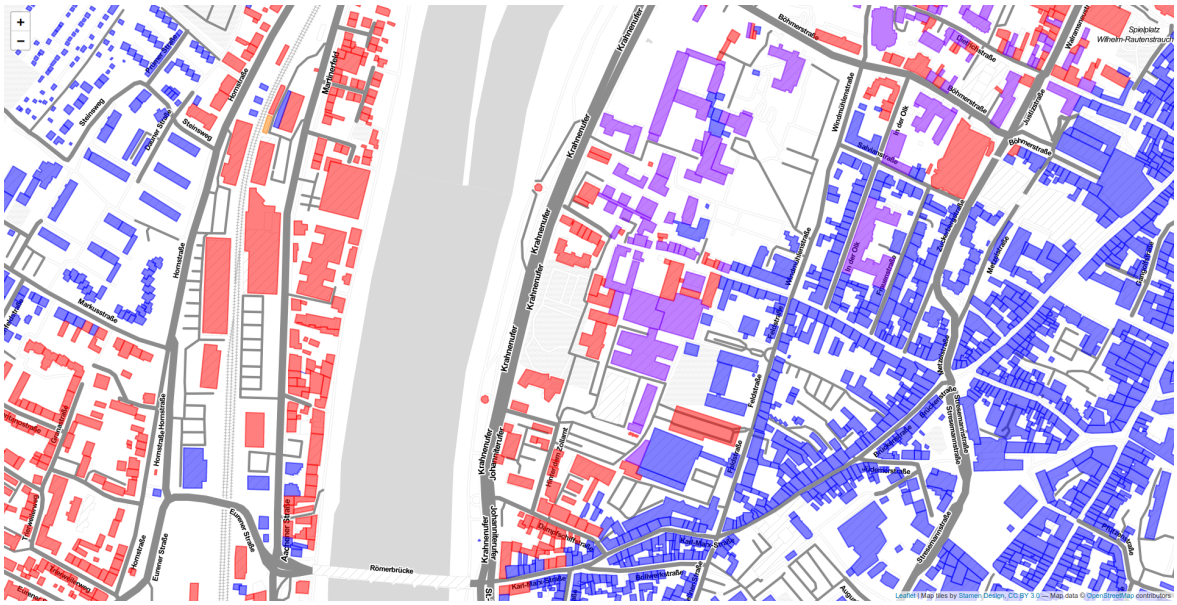


Figure 9: Flooded Areas with 12 Meters Water Height

house number. From this, unique addresses of individuals with their primary residency in Trier across all twelve years are exported.

In a next step, these 21,673 addresses are mapped to latitude-longitude coordinates via a local installation of OSM Nominatim (OpenStreetMap Wiki contributors, 2021a; Weymeirsch, 2021a). Using this mechanism, more than 20,000 addresses could be mapped, while 1,038 could either not be found at all or were ambiguous. For those remaining addresses, a Google Maps' address query is integrated using scraping techniques through the `selenium` package for `R` which is described in more detail in the Appendix B (Thorpe, 2023; Google LLC, 2024). After geocoding, 933 addresses resolved to the exact same coordinates, despite differing addresses. This is likely related due to differing spelling of street names, non-existent or otherwise erroneous extra-letters for the house number, missing buildings in the buildings corpus or even errors within the register. These coordinates were removed from the benchmark, such that 20,740 coordinate pairs (addresses) remain.

Finally, the coordinate pairs may be joined with the buildings data set. Because the coordinates of households (addresses) are not exactly equal to the centroids of buildings or any of their polygons' coordinates in all cases, two joining-strategies are applied sequentially to map addresses from the registers to the buildings from Open Street Map.

In the first matching approach, addresses are joined with buildings if the mapped coordinates are located within the building polygon (`point-in-polygon` strategy[3]). This process successfully mapped 12,923 coordinate pairs to building objects within roughly 23 seconds[4].

Additional 5,986 coordinate pairs are joined via the second procedure within 401 seconds, where coordinate-pairs are assigned to the closest building's centroid, with a gradually rising maximum distance of 1m to 50m (`centroid-nearby` strategy[5]). This only concerns those buildings, which have not yet been mapped to a previous address. The remaining 1,831 coordinate pairs are discarded, assuming that, potentially, the address or mapping is improper or because the building has been dismantled in the meantime.

This creates a list of buildings, of which we can be reasonably sure that they are residential or at least mixed-use. The benchmark data therefore consists of 21,108 addresses within 19,602 buildings. This is around 5.6% lower than the number of buildings with residential space reported in the census 2011 and 8.6% lower than in 2022, as shown in Table 1 (Statistical office of Rhineland Palatinate, 2015; Statistical offices of the Federation and the Länder, 2024). The number of empty residential *buildings* itself is not reported in the census 2022, however the share of empty *dwellings* amounts to only 5.7% in 2022 (Statistical Offices of the Federation and the Länder, 2024).

[3] Further explanation is available in Appendix C.1.

[4] Using a single-core process on an 11th Gen Intel® Core™ i7-1165G7 CPU and the statistical programming language `R 4.2.2`.

[5] Further explanation is available in Appendix C.2.

Year	2011	2022
Buildings with Residential Space	20,704	21,286
consisting of		
Residential Buildings	19,579	20,706
Residential Accommodations	53	52
Mixed use & other	1,072	528

Sources: Statistical offices of the Federation and the Länder (2024); Statistical office of Rhineland Palatinate (2015, T4).

Table 1: Buildings with Residential Space in Trier according to Census Results

Even if we assumed that each empty dwelling corresponds to an uninhabited residential building, a substantial gap between the benchmark data and the census aggregates remains. This misalignment may stem from slightly outdated or erroneous information in the registers or – more likely – from missing or misspecified objects in the buildings corpus and resulting joining issues.

5 Discussion and Conclusion

This paper showcases a pilot study for the creation of a house data set for the city of Trier for the year 2021 based on publicly available Open Data releases. The OSM data was chosen as base data due to its free availability, reasonably high quality, and timeliness of the information. The base data set is supplemented with various information from official sources such as the Office for Geodesy and Geoinformation Rhineland-Palatinate and the Office for Land Management and Geoinformation Trier, using mostly open standard APIs, such as WMS or scraped Open Data image sources. Furthermore, these data are supplemented by high resolution Lidar information in the form of a DSM and a DEM as reference point for the heights. In order to create a buildings data set for the whole of Germany, the described steps are to be automatized, which poses non-trivial due to the availability, complexity and non-standardization of some data releases.

Therefore, the approach presented in this paper does not seem feasible as a standalone dwelling corpus for all of Germany. For one, the presented approach is relatively complex and heavily relies on specific data sources for a given area, which could be in very different data formats throughout the different administrative areas, if they are available at all. Where little auxiliary information is available, the quality of OSM data is even more important. Particularly, when trying to *filter* different types of buildings, where the OSM tags are of little help, this may result in serious quality issues. Additionally, suitable benchmark information[6] can be used to meet certain criteria, such as number, types, and size of dwellings, or the height of buildings. However, such information is not available in all cases, either.

[6] Some of this data may be gathered from previous Census and Microcensus data sets, but regional or local alternatives could be of use as well.

Although the gathered information on the building objects in this pilot study seem reasonable in most cases individually, the evaluation of the buildings corpus with register data for the city of Trier reveals substantial issues and misclassifications on an aggregate level. There is a multitude of potential reasons for this and evaluation is non-trivial, as indicated in Section 4. The resulting buildings data set, appears appropriate for superficial analyses or coarse usage of residential living space. However, the quality is lacking for applications on finely granulated geographical areas such as city districts or neighbourhoods, let alone inner-city dynamic microsimulations.

While we conclude that this approach is unsuitable for a standalone dwelling corpus, the described methodology and data sources could be used to supplement or complement a buildings data set with a low information profile or which is not spatially complete, such as the early releases of the ALKIS data sets for Germany (Office for Geodesy and Geoinformation Rhineland-Palatinate, 2024).

Acknowledgements

This research was conducted within the research group FOR 2559 *Sektorenübergreifendes kleinräumiges Mikrosimulationsmodell (MikroSim)*, which is funded by the German Research Foundation.

References

Bivand, R. S., Pebesma, E., and Gomez-Rubio, V. (2013). *Applied spatial data analysis with R, Second edition*. Springer, NY.

Bricongne, J.-C., Turrini, A., and Pontuch, P. (2019). Assessing house prices - insights from "houselev", a dataset of price level estimates. *European Economy*, 1(101).

City of Trier (2023). Population register for the city of Trier. Retrieved September 0, 2023.

Copernicus Programme (2020a). Urban Atlas 2012: Metatdata. `https://land.copernicus.eu/local/urban-atlas/urban-atlas-2012?tab=metadata`. Retrieved June 8, 2020.

Copernicus Programme (2020b). Urban Atlas: Note on the evolution of FUA boundaries. `https://land.copernicus.eu/user-corner/technical-library/note-on-the-evolution-of-fua-boundaries`. Retrieved June 8, 2020.

Esri (2019a). Creating raster DEMs and DSMs from large lidar point collections. `https://desktop.arcgis.com/en/arcmap/latest/manage-data/las-dataset/lidar-solutions-creating-raster-dems-and-dsms-from-large-lidar-point-collections.htm`. Retrieved June 8, 2020.

Esri (2019b). Obtaining evaluation information for building footprints. `https://desktop.arcgis.com/en/arcmap/latest/extensions/3d-analyst/3d-buildings-obtaining-elevation-information-for-building-footprints.htm`. Retrieved June 8, 2020.

Esri (2019c). What is lidar data? `https://desktop.arcgis.com/en/arcmap/10.3/manage-data/las-dataset/what-is-lidar-data-.htm#ESRI_SECTION1_C3F0DD5E4ED24CE185C0B533A8D3A441`. Retrieved June 8, 2020.

Esri (2021). Imagery and Remote Sensing Solutions. `https://www.arcgis.com/home/item.html?id=10df2279f9684e4a9f6a7f08febac2a9`. Retrieved September 22, 2021.

European Union (2016). Mapping Guide for a European Urban Atlas. Mapping Guide v4.7, European Commission, Copernicus Programme. `https://land.copernicus.eu/user-corner/technical-library/urban-atlas-2012-mapping-guide-new`. Retrieved June 8, 2020.

Google LLC (2024). Google maps. Retrieved August 20, 2024.

Loga, T. and Diefenbach, N. (2013). Tabula calculation method - energy use for heating and domestic hot water. reference calculation and adaptation to the typical level of measured consumption.

Office for Geodesy and Geoinformation Rhineland-Palatinate (2017). Geobasisinformation der Vermessung- und Katasterverwaltung. Based on measures from 2017.

Office for Geodesy and Geoinformation Rhineland-Palatinate (2021a). Digitales Geländemodell Gitterweite 25m (DGM25). `https://lvermgeo.rlp.de/de/produkte/geotopografi`

e/3d-geodaten/digitale-gelaendemodelle-dgm/. OpenData Version, Retrieved June 30, 2021.

Office for Geodesy and Geoinformation Rhineland-Palatinate (2021b). Kartendaten zu Liegenschaften Rheinland-Pfalz. https://www.geoportal.rlp.de/map?WMC=11721. Retrieved June 30, 2021.

Office for Geodesy and Geoinformation Rhineland-Palatinate (2024). Amtliches Liegenschaftskataster Informationssystem (ALKIS). https://lvermgeo.rlp.de/ueber-uns/unsere-aufgaben/liegenschaftskataster/alkisr. OpenData Version, Retrieved November 15, 2024.

Office for Land Management and Geoinformation Trier (2021). Grenzen der Schulbezirke. https://geoportal.trier.de/trier/mod_ogc/. Retrieved September 20, 2021.

Office for the Environment Rhineland-Palatinate & City of Trier (2019). Hochwassergefahrenkarte Trier. https://geoportal.trier.de/trier/index.php?service=bauen. Retrieved October 20, 2021, Data License: CC BY-NC 3.0 DE.

Open Geospatial Consortium (2019). OGC GeoTIFF Standard. https://www.ogc.org/standards/geotiff. Retrieved October 1, 2021.

OpenStreetMap Wiki contributors (2019a). About OpenStreetMap. https://wiki.openstreetmap.org/w/index.php?title=About_OpenStreetMap&oldid=1929744. Retrieved June 5, 2020.

OpenStreetMap Wiki contributors (2019b). DE:Gebäude. https://wiki.openstreetmap.org/w/index.php?title=DE:Geb%C3%A4ude&oldid=1898109. Retrieved June 8, 2020.

OpenStreetMap Wiki contributors (2019c). DE:Map Features. https://wiki.openstreetmap.org/w/index.php?title=DE:Map_Features&oldid=1910291. Retrieved June 5, 2020.

OpenStreetMap Wiki contributors (2020a). DE:Key:building. https://wiki.openstreetmap.org/w/index.php?title=DE:Key:building&oldid=1944877. Retrieved June 8, 2020.

OpenStreetMap Wiki contributors (2020b). Stats. https://wiki.openstreetmap.org/w/index.php?title=Stats&oldid=1990881. Retrieved June 5, 2020.

OpenStreetMap Wiki contributors (2020c). Why OpenStreetMap? https://wiki.openstreetmap.org/w/index.php?title=Why_OpenStreetMap%3F&oldid=1981275. Retrieved June 5, 2020.

OpenStreetMap Wiki contributors (2021a). Nominatim. https://wiki.openstreetmap.org/wiki/Nominatim. Retrieved September 22, 2021.

OpenStreetMap Wiki contributors (2021b). Overpass API. https://wiki.openstreetmap.org/wiki/Overpass_API. Retrieved September 22, 2021.

Panigrahi, N. (2014). *Computing in Geographic Information Systems*. CRC Press. ISBN: 1482223147, 9781482223149.

Statistical office of Rhineland Palatinate (2015). ZENSUS 2011 - Gemeindeergebnisse Gebäude und Wohnungen (Band 405).

Statistical offices of the Federation and the Länder (2020). Ergebnisse des Zensus 2011 zum Download - erweitert. https://www.zensus2011.de/DE/Home/Aktuelles/DemografischeGrunddaten.html?nn=3806618#Gitter. Retrieved: June 9, 2020.

Statistical offices of the Federation and the Länder (2024). Zensus 2022 - Gebäude: Art des Gebäudes, T. 3000G-1001. Retrieved November 22, 2024.

Statistical Offices of the Federation and the Länder (2024). Zensus 2022 - Wohnungskennzahlen: Leerstandsquote und Eigentumsquote, T. 4000W-0001. Retrieved November 22, 2024.

Thorpe, A. (2023). selenium: Low-Level Browser Automation Interface. https://cran.r-project.org/package=selenium. Version 0.1.3.

Topf, J. and Topf, C. (2021). OpenStreetMap Taginfo - Tag: building. https://taginfo.openstreetmap.org/keys/building#values. Retrieved September 22, 2021.

U.S. General Services Administration (2019). Level of Detail. https://www.gsa.gov/real-estate/design-construction/3d4d-building-information-modeling/guidelines-for-bim-software/document-guides/level-of-detail. Retrieved October 1, 2021.

Weymeirsch, J. (2021a). OSMtools - a toolbox of OSM functions for GNU R. https://gitlab.rlp.net/weymeirsch/osmtools. Version 0.0.8.

Weymeirsch, J. (2021b). WMStools - Simple interface to retrieve data from WMS in GNU R. https://gitlab.rlp.net/weymeirsch/wmstools. Version 0.0.1.

Appendix

A Downloading Flood Data

As described in Section 2.5, in order to download the flood map GeoTIFF imagery, a combination of `GNU R`, a specially prepared web browser and some simple JavaScript code was used. However, this may require further explanation, as the procedure is not as straight forward, as simply running some code.

Firstly, the `R` script creates a vector of the 2,700 URLs for the 270 sections of Trier, the five water level scenarios and the two map types (water heights and groundwater). Each of the URLs has the form of, e.g.:

`https://geoportal.trier.de/trier/index.php?service=bauen&zl=13&lo=1.0&bl=keine_hintergrundkarte&layers=hw1200dw&x=337458.372472304&y=5518997.94559161`

Whereas several parameters are pre-configured already, such as the zoom-level `zl` (here: 13) of the map, the layer opacity `lo` (here: 1.0, thus fully opaque), the background layer `bl` (here: `keine_hintergrundkarte`, eng: no background map), the relevant map layer `layers` (here: `hw1200dw`, meaning water level of 12 meters, showing only groundwater) as well as the latitude `x` and longitude `y`, indicating the section of the map that is to be shown, when opening the URL. The `R` script then cycles through the URLs one by one, executing an OS call to open them in the web browser (here: Mozilla Firefox 78 on GNU/Linux) with a 30 seconds delay between them:

```
for (i in 1:length(urls)) {

  # Some output for the user
  cat("Opening", urls[i], "\n\n")

  # Opening firefox (asynchronously)
  system2(command = "firefox",
          args = paste0("--new-tab '", urls[i], "' &"))

  # Wait for 30 seconds until the procedure ended
  Sys.sleep(30)

  # Rename the newly downloaded GeoTIF file
  newfile <- list.files("~/Downloads", pattern = "GeoTIFF_.*.tif",
                        full.names = TRUE)
  newname <- sub(x = urls[i], pattern = "",
                 pattern = "^.*keine_hintergrundkarte&") %>%
             gsub(pattern = "&", replacement = "_")
  file.rename(from = newfile[1],
              to = paste0("~/Downloads/", newname, ".tif"))
}
```

The web browser has been prepared beforehand, such that:

1. It directly downloads any file without a user prompt
   (in `about:config`, set `browser.download.useDownloadDir` to `true`).
2. It allows JavaScript code to close the current tab or window
   (in `about:config`, set `dom.allow_scripts_to_close_windows` to `true`).
3. A browser *add-on* that automatically executes a given JavaScript code, when opening a certain website or domain, such as `https://geoportal.trier.de` (e.g. the add-on *Javascript* by chee).

It is needless to say, that either of the settings or *add-ons* above could potentially be a threat to security and/or privacy if they were to be used in normal web browsing and should only be set for the duration of the downloading procedure.

The according JavaScript code, which is run by the browser when opening the Geoportal is given below. It consists of three simple functions:

- `SimulateClick()`: A function which simulates a click on a given HTML object
- `SelectLayer()`: A wrapper function around `SimulateClick()`, here used to work around a bug in the Geoportal, that seems to ignore the `bl=keine_hintergrundkarte` parameter, when exporting the image.
- `DownloadGeoTiff()`: A wrapper function around the Geoportal's own internal functionality to export the current view as GeoTIF file.

The functions are executed asynchronously after some short delays in order to let the site load and download the file completely. The JavaScript code is highly adapted to the Document Object Model (DOM) structure of the Geoportal at the time of writing. Any changes to the underlying software could potentially break the functionality to a point where it may not run any more without further adaptions.

```
/* ------------------------------------------------
   Simulate click of a mouse button
   ------------------------------------------------ */
function SimulateClick(element) {

  // Set default options of the mouse action
  var options = {
    pointerX: 0,
    pointerY: 0,
    button: 0,
    ctrlKey: false,
    altKey: false,
    shiftKey: false,
    metaKey: false,
    bubbles: true,
    cancelable: true
  }
  var oEvent, eventType = null;

  // Create the mouse click event
  oEvent = document.createEvent("MouseEvents");
  oEvent.initMouseEvent("click", options.bubbles,
                        options.cancelable, document.defaultView,
                        options.button, options.pointerX,
                        options.pointerY, options.pointerX,
                        options.pointerY, options.ctrlKey,
                        options.altKey, options.shiftKey,
                        options.metaKey, options.button,
                        element);

  // Execute action (click)
  element.dispatchEvent(oEvent);
  return element;
}

/* ------------------------------------------------
   Select map layer background via the dropdown menu
   ------------------------------------------------ */
function SelectLayer() {
  // Select Dropdown Menu
  SimulateClick(document.querySelector("#baselayer_select >" +
    "span:nth-child(1) > span:nth-child(1) > span:nth-child(1)"));
  // Select 9th Layer ("no background")
  SimulateClick(document.querySelector("#baselayer_check_listbox >" +
    "li:nth-child(9)"));
}

/* ------------------------------------------------
```

```
49    Export imagery of current view
50    ------------------------------------------------- */
51 function DownloadGeoTiff() {
52   // Set export format (internal function of geoportal)
53   changeImgexportformat("image/tiff");
54   // export current view (internal function of geoportal)
55   submitPrintpdf();
56 }
57
58 /* -------------------------------------------------
59    Execute functions asynchronously
60    ------------------------------------------------- */
61
62 // After 5s: Select background layer
63 setTimeout(() => SelectLayer(), 5000);
64
65 // After 7s: Open Export Dialog (internal function of geoportal)
66 setTimeout(() => openPrintdialog("a4_landscape"), 7000);
67
68 // After 10s: Export GeoTIFF imagery
69 setTimeout(() => DownloadGeoTiff(), 10000);
70
71 // After 20s; Close Tab
72 setTimeout(() => close(), 20000);
73
74 /* EOF */
```

Note: The gathering process of the flood data was conducted in 2021 already, when the `selenium` *R-package, described on page 24 did not exist yet. Today, this process could be more streamlined and without the need to prepare a web browser or other third party tools such as browser add-ons, through the use of* `selenium` *(Thorpe, 2023).*

B Scraping Google Maps

The scraping approach utilises the mechanism of the Google Maps web application to show and encode the requested data. Strictly speaking, the mechanism used here does not *(web) scrape* Google Maps, it merely runs the web viewer as any normal user would and gathers the required data from the URL the web viewer redirects the user to after a search is issued. The output within of the web application the browser window itself is in fact irrelevant for the process.

In simple terms, whenever a search is started in Google Maps either through the web interface or by calling the search URL used in the R-code further below, the address dissolver redirects the browser to another URL after a brief period. The final URL contains a lot more information about the results, which is however encoded and not human-readable. For example, querying Google Maps for the address of Trier University via the query URL used within in script:

- By URL-encoding the search-string (*Universitätsring 15, 54296 Trier, Germany*), the query-URL results as: `https://www.google.com/maps/search/?api=1&query=Universit%C3%A4tsring%2015,%2054296%20Trier,%20Germany`
- Initially, Google Maps redirects to `https://www.google.com/maps/search/universit%C3%A4tsring+15+54296+trier/@49.7569147,6.6425227,14z`, whereas the last part of the URL, right of the `@`-symbol, denote configurations for the webviewer. In this case the longitude and latitude of the upper-left corner of the map that is shown and `14z` being the zoom-level
- After a short period in which Google's address dissolver is running, the web application shows the map object that it deems to be a match for the given search term. At the same time it redirects to the final URL, which includes encoded information about the location of the marker and much more. It is this URL the mechanism described here is seeking – nothing else is required: `https://www.google.com/maps/place/Universit%C3%A4t+Trier+%2F+DM,+Universit%C3%A4tsring+15,+54296+Trier/@49.7443274,6.6822537,17z/data=!4m6!3m5!1s0x47957c0a08c3fe57:0x52a010b0d9bfb5b1!8m2!3d49.7443274!4d6.6848286!16s%2Fg%2F12ltz3j7r`
- Of particular interest are two parts of the resulting URL:
    1. The string between the `/place/`-portion and the `@`-symbol of the URL denotes the address of map object that was matched to the input search term. In this case it matched the DM-building at Trier University. This string could give hints on mismatches, in case where completely different addresses are shown.
    2. The encoded string at the end of the URL after `/data=` gives information about the location of the map-marker shown in the web application, indicating the location that was matched. In particular, the marker's coordinates can be retrieved between the `!3d` and `!4d` portions (longitude), respectively the `!4d` and `!16s`

portions (latitude). Using Regular Expressions, these coordinates can easily be extracted from the URL.

This mechanism is not *future-proof*, as it heavily relies on Google Maps' current behaviour, which could change at any time when Google makes big or small changes to its web application. Below are a few lines of code that were used to retrieve the remaining addresses' coordinates via Google Maps, that could not be retrieved via OSM Nominatim:

```
# cycle through addresses that have not (yet) coordinates
for (i in which(is.na(addr$lon))) {

  # if no selenium / browser session is running, re-start it
  if (!selenium_server_available()) {
    server <- selenium::selenium_server(interactive = FALSE)
    session <- SeleniumSession$new()
    Sys.sleep(10) # give browser time to start
  }

  # Construct URL from address string
  query <- utils::URLencode(paste0(
    "https://www.google.com/maps/search/?api=1",
    "&query=",
    addr$street[i], " ", addr$number[i], " ", addr$extra[i],
    ", Trier Germany"
  ))

  # navigate URL to constructed query
  session$navigate(query)

  # Fetch URL of the browser after waiting for re-direct
  Sys.sleep(10)
  addr$url[i] <- session$current_url()

}

# Parse URL to retrieve Coordinates
addr$lon[is.na(addr$lon) & !is.na(addr$url)] <- gsub(
  x = addr$url[is.na(addr$lon) & !is.na(addr$url)],
  pattern = "^.*!3d([0-9\\.]+!4d.*$",
  replacement = "\\1",
  perl = TRUE
)
addr$lat[is.na(addr$lat) & !is.na(addr$url)] <- gsub(
  x = addr$url[is.na(addr$lon) & !is.na(addr$url)],
  pattern = "^.*!4d([0-9\\.]+!16s.*$",
  replacement = "\\1",
  perl = TRUE
)
```

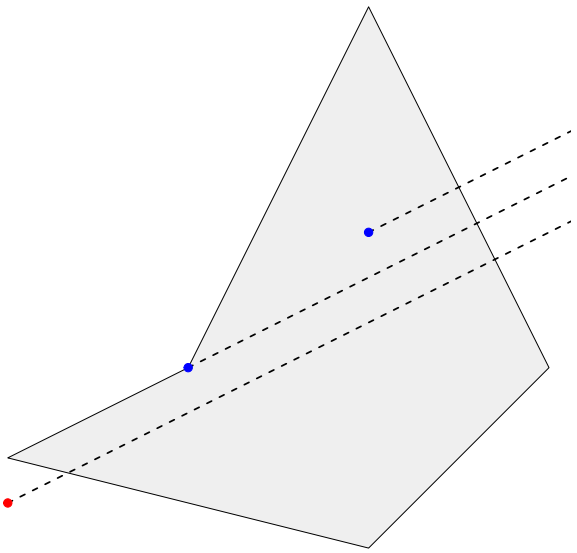
According to the Google Maps / Google Earth Terms of Service from July 2022, retrieved on August 22, 2024, this approach is well within the terms of service. None of the behaviours listed under section 2 *Impermissible behaviour* were violated.

C Matching association of coordinate points with polygons

C.1 Point-in-Polygon

Allocation of point-data (here: addresses) to coordinate polygons (here: building plans) is a straightforward process as long as the points are located within the polygon. In this case, the matching procedure may follow the `point-in-polygon` technique, as implemented in the R-function `sp::point.in.polygon()` and illustrated in Figure 10.

This procedure follows the *ray casting algorithm*, whereas from each point, a linear ray (dashed lines in Figure 10) is cast in a random direction and the number of intersections with the polygon boundaries is counted. If the amount is even, the point lies outside the polygon, if the amount of intersections is odd, it lies within. If there are no intersections at all, a different direction of the ray has to be chosen (Bivand et al., 2013).



Legend: Coordinate pairs located within ■ and outside ■ of a polygon.

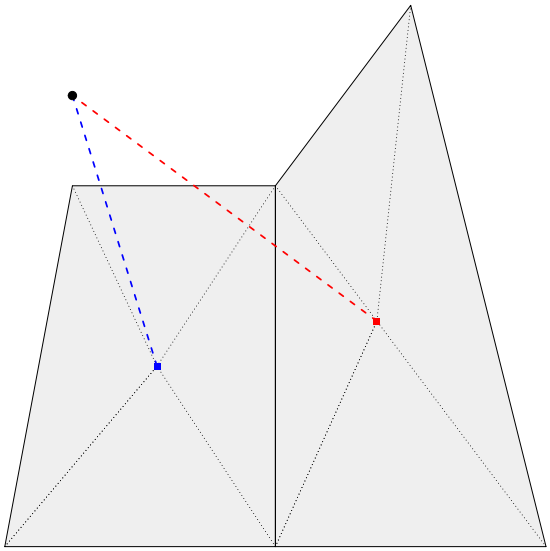
Figure 10: Illustration of the Point-in-Polygon concept

C.2 Centroid-Nearby

In the application within this work, the coordinate pair of a geocoded address string often times lies *very close*, yet *outside* of the according building polygon. This has to do

with the way that addresses are sometimes associated with building objects or locations in OpenStreetMap.

For those cases, the closest building polygon to some coordinate pair was chosen as the match, using maximum distance constraints in order to prevent linking of unlikely matches. This is illustrated in Figure 11. As a reference point for each building, the polygons' centroids are used and their distance to the point calculated (dashed lines), using the inverse haversine function from Equation 1 on page 12.



Legend: Centroid of polygon with the shortest ■ and longest ■ distance to the coordinate point ■.

Figure 11: Illustration of the Centroid-Nearby concept

A polygon's centroid is essentially the weighted centre point between all corner coordinates (dotted lines in Figure 11) that define its area. Because the building polygons are relatively small and the distances we want to calculate are ideally just a few meters, we may ignore the spherical form of the earth in this application and approximate the buildings' centroid with the arithmetic means of the polygon edges' latitude and longitude coordinate pairs.