MAP GENERATION FOR RETRO GAMING ENGINES

H. Tauscher[1]*, V. T. Le[1]

[1] HTW Dresden, University of Applied Sciences, Dresden, Germany - (helga.tauscher, viettien.le)@htw-dresden.de

KEY WORDS: Gamification, Map generation, Tile sets, IFC, CityGML, OSM.

ABSTRACT:

Current attempts to integrate gaming, geospatial and AEC technologies are driven by increased hardware capabilities and target virtual worlds as close to reality as possible. In this paper we pursue a converse destination and populate low-tech virtual worlds from geospatial data, building and city models, specifically for retro gaming engines in spatial chat tools. We consider an island, a building and a city district scenario and populate these scenarios from OSM, IFC, and CityGML data sources. The derived worlds are targeting use cases in business, educational and recreational settings. Based on a prototypical implementation, we study the feasibility and limitations of good quality retro gaming map generation with automatic means from existing data sets, but also the potential of such worlds for Green IT, accessibility and inspiration of new user groups. We describe the algorithms and processes to generate the maps and outline the concept for a user survey. Beyond that, by discussing map generation techniques from classical gaming in context of and comparison with geospatial and AEC practices, this paper contributes a retrospective of early gaming techniques that might be both entertaining and informative for current development practice.

1. INTRODUCTION

Currently, we see a convergence of virtual worlds with the real world inspiring a mutual rapprochement of the gaming and geospatial industry (OGC and USGIF, 2021), subsequently also involving the architecture, engineering and construction (AEC) industry with a longer history of VR/AR and gaming affirmation (Bille et al., 2014). This convergence and surrounding technologies are discussed under the term "digital twin" or, when taken to the global scope, referencing the idea of the "metaverse", a termed coined by Neal Stephenson in his novel "Snow Crash" (Stephenson, 1992). Increasing hardware capabilities put this vision within reach, achievable through massive gathering of data about the natural and built environment as well as improved data processing and transmission, texturing and rendering methods to achieve real-time hyperrealistic replicas and high-resolution simulations. As such, these topics constitute fruitful and abundant areas for current research and development. With this work however, we are taking a step back and look at converging real and virtual worlds on the geospatial and building level through methods that may seem outdated, obsolete and deprecated: those of retro or classical games.

Retro gaming has regained some popularity through applications such as the game Minecraft[1] and its open source sister Minetest or ad-hoc video conferencing systems such as Remo, Gather or WorkAdventure. In an ad-hoc video conferencing system (sometimes also called spatial chat tool), participants move freely in a virtual world and establish temporary conferences by moving their avatars near to each other. The increased popularity is mainly driven by aesthetic and nostalgic purposes, but has potential benefit beyond that. The methods developed decades ago were targeting systems with low resources. If revised today, they could help achieve Green IT goals, e.g. less material usage and electronic waste through longer lifetime of devices (Rudolf et al., 2020) or reduced energy consumption through moderate utilization of system resources. Reduced system requirements could also broaden the number of potential users and increase accessibility of software and services by including people in regions with poor internet connections or with insufficient spending capacity to purchase the latest high-tech devices. In WorkAdventure, for example, one of the declared motivations to resort to 2D instead of 3D, was to lower bandwidth and device specs (Négrier, 2021) to increase accessibility in day-to-day casual use during the Covid-19 pandemic when working in remote areas. Even though the effect of these savings can be used up by the overhead of modern runtime environments, we still believe that introspection of development practices and recall of early game development techniques is valuable as an educational and inspirational exercise. From an end user point of view, there is certainly a practical benefit in easy customization of engaging worlds for casual, educational and business meetings. Finally, the recreational and fun touch of retro games may also foster engagement of the public and younger audiences for geospatial and building information problems and solutions.

The objective of our study is three-fold: We first review classical gaming techniques (Section 2), we then identify use cases for their application in geospatial and built environment context (Section 3 and finally demonstrate prototypical implementations that populate retro gaming worlds from geospatial data and digital building models (Section 4).

2. RELATED WORK

Visualization and animation in classical games are based on indexed tile maps, tile sets and sprites. Tiles and sprites make up the building blocks of a game's graphics for background, foreground, objects and avatars. They consists of crafted raster images, usually of square size, e.g. 32x32 pixels, and are bundled in tile or sprite sets transmitted as images containing multiple tiles or sprites arranged in a grid. With knowledge of the

* Corresponding author

[1] Two examples show the popularity through adoption in the geospatial and AEC industries: The city of Frankfurt has issued a call for participative city planning with minecraft (Frankfurt 2099, `https://ffm2099.thejocraft.de/`. FME has added support for conversion to Minecraft worlds (FME, 2020) and Norway is available for Minecraft download.

tile size and an indexing scheme (e.g. top–down, left–right), the tiles or sprites in such a set can be addressed with a one- or two-dimensional index. Both tiles and sprites are arranged in layers and may appear and disappear on screen throughout a run of the game. While tiles are supposed to appear aligned to an overall grid, sprites can be animated on top of the tiles without being tied to positions on grid cells. In addition, the whole composition with the grid can be shifted across the screen to produce a movement of the view section in the larger world, in classical games often constraint to particular directions (side and vertical scrolling).

This structure of tiles and sprites makes perfect use of the then-current graphics hardware capabilities by staying close to GPU operations, that is copying and shifting ranges of pixels via blitting or hardware sprites. The speed and number of bitmap storage and copying operations are still relevant measures for 2D graphics, for example in GPUs for embedded systems such as the BT815 used in Gameduino[2] (Bowman, 2013, Chapter 3, still valid for Gameduino 3). In addition to fostering efficient GPU operations, the overall size of a map is reduced for storage and transfer as well as RAM usage, since the indexes or ranges of indexes into the tile and sprite sets are all the game logic needs to know and handle with regard to the visual representation of the game elements.

Similar techniques of arranging tiles in a grid can be found in cartography when it comes to map and tile servers, just that here, the context and provenance of the tiled map and the motivation and objective to slice a raster graphic into bite-sized pieces is slightly different. First, while still aiming at good performance and efficient memory usage, the notion of efficiency has shifted with hardware evolution. Tiles are usually larger — a tile size of 256x256 has become the de-facto standard, which can be attributed to larger display resolutions in conjunction with more powerful GPUs and larger RAM. Second, every rendered tile differs from others, maps are larger and feature different scale levels. Further, the tiles are rendered from vector data instead of hand-crafted as raster images. The rendering and caching of rendered tiles is done automatically instead of manually and optionally on-the-fly. This task is done by a server in order to relieve client from the rendering task. Thus the need to slice a map into tiles is rather driven by limitations of data transfer between server and client (selectively on request) rather than by display requirements. By raising the bar of hardware capacities and requirements, clients became even powerful enough to handle rendering of vector data themselves, which has led to the evolution of raster tiles towards vector tiles and even 3D tiles.

If we are aiming to recreate and leverage the frugal methods of the past to visualize geospatial and construction models with lower system requirements, we are particularly interested in ways to efficiently generate, provision and deliver 2D visualization from 2.5D and 3D data. The fundamental challenge is to create tile arrangements based on different parallel projections such as orthogonal top-down, cavalier and isometric views, where the number of tiles in the tile set is substantially lower than the number of grid cells in the map.

This can be rephrased as classification or generalization problem and thus could motivate new application areas and a fresh view on cartographic generalization methods, for example by applying constraint-based generalization (Sayidov et al., 2020)

[2] The Gameduino 3X Dazzle was released in 10/2020: `http://gameduino.com`, BT815: `https://brtchip.com/bt81x/`

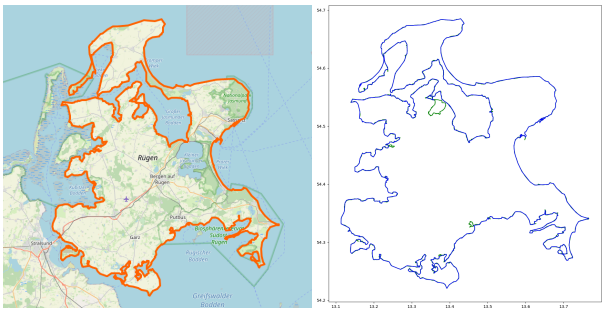


Figure 1. Island scenario input: An OSM polygon, ©OpenStreetMap contributors [3].

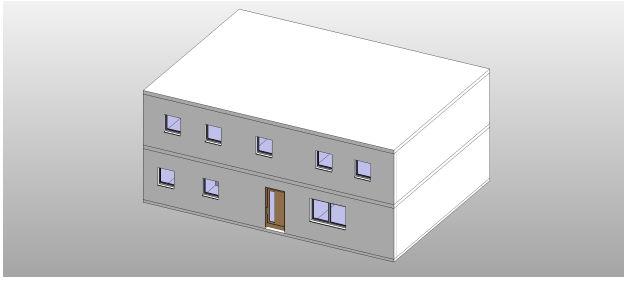


Figure 2. Building scenario input: A Revit 3D building model.

to prepare input data for the tiling process. Another area of traditional knowledge to tie in with are developments in the AEC area, such as the derivation of architectural drawings from 3D data and more recent reinterpretations for interactive media (Gotlib et al., 2020) or applications for indoor maps via building floor plans in GIS context (Konde et al., 2018, for example).

3. USE CASES AND SCENARIOS

In respect of the use cases, we are considering professional work, educational and recreational settings. Work settings have given the name to the WorkAdventure platform. According to its originator, WorkAdventure has evolved during the Covid-19 pandemic in an attempt to facilitate informal workplace communication. Colleagues could gather around a co-worker's desks or in the coffee lounge and simulate a casual get-together despite working remotely from their home offices and beyond the realm of official and formal business meetings via traditional video-conferencing software.

With regard to the geospatial and digital building content, we are studying three scenarios on different scales: an island (Section 4.2), a building (Section 4.3), and a city district (Section 4.4) scenario. Islands and buildings are classic gaming environments because they constitute closed worlds with natural borders. City or village scenarios do exist as well as game environments, but they need some artificial boundary as long as they are not situated in medieval Europe and feature surrounding walls.

The worlds for these scenarios are populated from three different types of data sources: OpenStreetMap (OSM), Industry Foundation Classes (IFC) and CityGML (LOD 2). These data sources are based on open standards and as such our conversion procedures can be reused to produce maps from other input than the one used in our study. OSM data (for example Figure 1) is freely available, though with varied completeness

[3] OSM copyright and license: `https://www.openstreetmap.org/copyright`.

and quality. CityGML is available from governmental agencies and cadastral or surveying offices, distributed under different open or commercial licences. IFC is usually not available as open data, except for some demo or sample datasets. However, as building information modelling becomes widely accepted, more and more building owners and operators accumulate or have access to respective data (for example Figure 2).

We see the use cases and scenarios as independent aspects, although some use cases and scenarios work better together than others. For example, maps for workplace settings for WorkAdventure usually feature indoor building scenarios and replicate the office layouts of a company. However, work meetings could also happen in other work related environments. For educational scenarios we assume either similar setups with school or university buildings replicated with seminar rooms and lecture halls. But an educational setting could also benefit from virtual environments derived from the subjects taught, e.g. in geography and can then fit well with the geospatial or city district scenario. For recreational settings we anticipate interest in locations where the meeting participants share common experience in or want to tell their friends and family about, e.g. travel destinations, places of birth or current residence.

4. PROTOTYPICAL IMPLEMENTATION

To date, we have created prototypical implementations for the island and building scenario. In the following, we first describe the target environment for the implementation and then show and discuss map setups, methods and algorithms to extract and process the required information from the sources for these two scenarios. The implementations for the scenarios were conducted independently by the authors without sharing code. Both the island and building scenarios are written in Python, but currently make use of different libraries. The approach and algorithms do naturally share some commonalities, though.

4.1 Target environment

With regard to the target environments, the prototypical implementation aims at the ad-hoc video-conferencing system WorkAdventure and the underlying 2D game engine Phaser. Phaser is capable of consuming a JSON format equivalent to TMX (Tile Map XML) with embedded tile sets. TMX and TSX (Tile Set XML, for external tile sets) are the custom XML-based formats of the widespread 2D tile editor "Tiled". These formats have evolved into de-facto standards and are used by many game frameworks and engines besides Phaser and subsequently by applications that build upon a game engine like to WorkAdventure, for example Gather. Further there are engine-independent implementations to write, parse and load TMX and TSX data in many programming languages. As the "Tiled" JSON is semantically equivalent to TMX/TSX but differs in field names, not all of these implementations may support the JSON version. For our implementations so far we did without such libraries and wrote the tile map JSON by example and according to the documentation.

Apart from the tiles that form the background for the avatars to be rendered on, a map can also contain further content such as objects, multiple layers and arbitrary attributes attached to tiles in the map or in the set as key-value pairs. Interpretation of these attributes is up to the consuming application, semantics such as whether a tile causes collision is not specified by TMX. Thus apart from the tile map and set format, also the documentation of the consuming application, in this case WorkAdventure has to be consulted, for these application-specific semantics and also to determine the extent to which TMX features are supported. Object layers, for example, are currently not supported. But WorkAdventure does support the interconnection of multiple maps into worlds and navigation via tagged entry and exit tiles. Recently, further functionality has been added, e.g. scripting capabilities to create more interactive maps and worlds. The WorkAdventure web application can load maps and worlds from external servers, such that the hosting of the ad-hoc videoconferencing application and the hosting of the maps are decoupled and any simple web server is sufficient to publish maps and worlds for WorkAdventure.

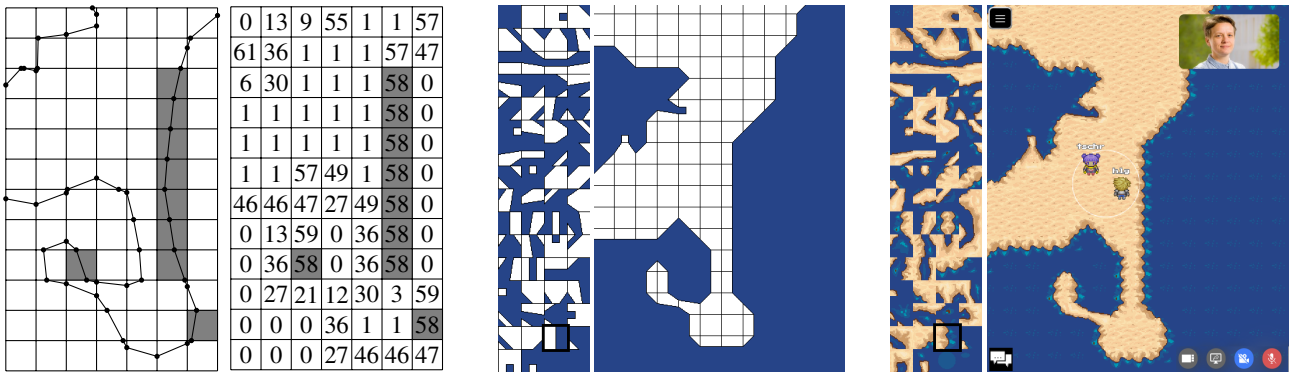
4.2 Island scenario

The conversion from OSM polygons to tile map JSON uses standard Python libraries such as *json* or *urllib*. In addition the Python Imaging Library (PIL) has been used to render the maps for testing purposes outside of Phaser or WorkAdventure and to create the "abstract" tile set.

The island polygons from OpenStreetMap are first run through a preprocessing service provided by the French OSM community[4] (see Figure 1). The service is applicable to any feature with polygonal geometry, not just islands, and requires the feature to be modelled as a relation, not just a simple way in OSM. The polygon pieces are collected and assembled, then cached in a PostGIS database and can be retrieved in different formats after optionally applying PostGIS operations to simplify and smoothen the geometry. The applicable operations are Douglas-Peucker algorithm via *ST_SimplifyPreserveTopology*, grid reduction via *ST_SnapToGrid* and polygon offset towards the exterior or interior via *ST_Buffer*. We don't apply any offset, but have empirically established a value for the grid size and simplification tolerance based on the maximum extension of the respective polygon. Thus we first download the original and subsequently a simplified version in GeoJSON format for further processing. After preprocessing, the geospatial polygon coordinates undergo map projection and scaling. We apply logarithmic scaling such that for large islands the maximum recommended map size is not exceeded, small islands provide a minimum number of tiles to walk around and at the same time differences in island size can still be experienced.

These coordinates are then further processed into a data structure designed to derive the appropriate tile and its index in a tile set for a particular cell. To this end we intersect the polygon with a grid according to the tiling and keep only the topology of the resulting intersection. That is, we throw away the information about how the polyline runs inside the cell and any potential vertices lying in the cell. We only keep the information of how many segments cross a particular cell and which of the four sides of a cell each segments enters and exits. For corner cases we also need the information in which order segment entries and exits appear on a particular border of a cell. Taken together, this information makes up a "signature" to identify a tile in the tile set. Hence the mapping between this signature and the tile set is bijective or, in other words, the "signature" can be used as an index into the tile set with the exception of the zero-segments case which can be a land or water tile and has to be discriminated separately. Figure 3a shows the intersection of grid and polygon in the left part and resulting indexes in the right part. The tile with index 58 is highlighted on both sides

[4] Polygon preprocessing: `https://polygons.openstreetmap.fr`.



0	13	9	55	1	1	57
61	36	1	1	1	57	47
6	30	1	1	1	58	0
1	1	1	1	1	58	0
1	1	1	1	1	58	0
1	1	57	49	1	58	0
46	46	47	27	49	58	0
0	13	59	0	36	58	0
0	36	58	0	36	58	0
0	27	21	12	30	3	59
0	0	0	36	1	1	58
0	0	0	27	46	46	47

(a) Creation of the tile map by indexing into the tile set based on cell topology.

(b) Tile map rendered with the ”abstract” tile set

(c) Tile map used in WorkAdventure with the ”sand” tile set.

Figure 3. Creation of the indexed tile map and use with different tile sets. The tile with index 58 is highlighted in the map and tile set.

and also in the tile sets shown in Figures 3b and 3c. This tile corresponds to a cell with exactly one segment running from the bottom (south) border to the top (north) border of the cell (with land to the left and water to the right). The tile with index 36, for another example, has one segment as well, running the opposite direction from top to bottom of the cell.

Figures 3b and 3c show the tile map rendered with different tile sets. The ”abstract” tile set (Figure 3b) was created semi-automatically to cover all cases where a maximum of two polygon segments cross a tile. Segments entering and exiting via the same border of a cell are only considered if they are the only segment in that cell’s signature. This decision was made in order to keep the number of tiles in the tile set reasonable. Any such segment in a cell with other segments in addition is eliminated by replacing the two connected segments in the neighbouring cell by a single segment. This way, most of the corner cases with more than two segments in a cell could be reduced to cells with a maximum of two segments. This can be seen as a simple way of agent-based generalization. The positioning of the entry and exit points on the borders in the synthetic tile set is such that any cell border with one crossing is crossed in the middle and any border with two crossings is split in three. The ”sand” tile set (Figure 3c) is an extended version of a tile set from OpenGameArt [5]. While the original tile set contains only tiles according to signatures with zero or one segment, the extended tile set covers the same signatures as the ”abstract” tile set and additional tiles have been crafted by traditional means of pixel artistry.

Finally, the tile map is enriched with application-specific semantics for WorkAdventure. In the tile set, the tile with index 0 (water tile) is marked as ”colliding” with a property of type Boolean, such that these tiles on a map cannot be entered. Coastal tiles in the tile map are collected on a dedicated entry layer, such that initial placement of avatars is not on water without any options to move and neither in the middle of a larger island without any orientation.

The algorithms have been successfully applied to an assortment of 39 islands of varying size and the scripts as well as results have been published on Github[6]. Islands that were not modelled as relations but as ways at the time of inspection in OSM, have been excluded upfront. Five more islands were initially selected and have been dropped due to various issues in the early phase of development, but should be re-evaluated with the latest version of the scripts.

4.3 Building scenario

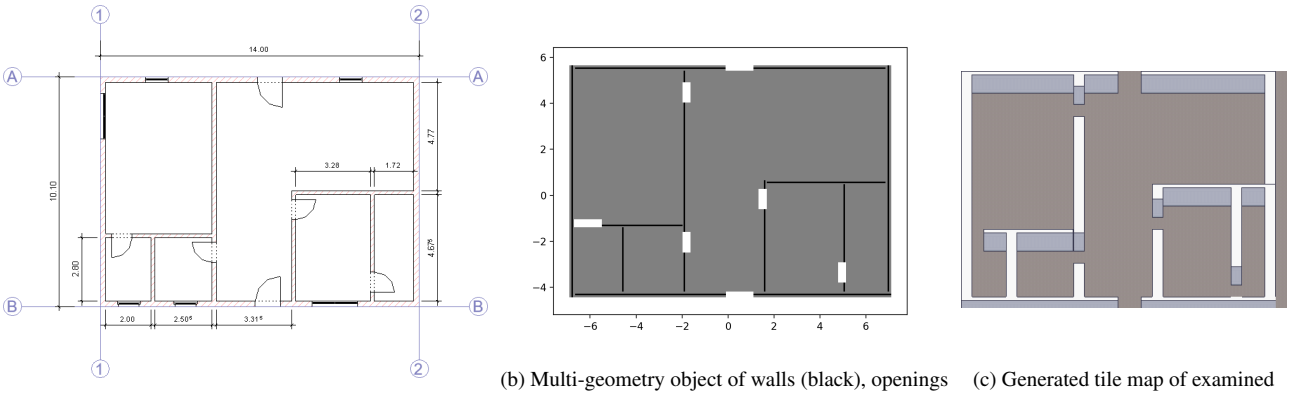
The data source for this scenario conforms to the Industry Foundation Classes (IFC), a standard for open Building Information Modelling (BIM) data exchange. With IFC, buildings and their components can be described in detail. As this data model follows the object-orientated approach, it is possible to represent both semantic structure and geometry of a building model. On the one hand a building is broken down semantically into its building components, on the other into its spaces. Along the semantic classification of objects, detailing relationships between objects is also a key quality. These relationships are not formed by direct associations, but instead use objectified relationships (Borrmann et al., 2018).

The goal of the implementation is to create tile maps in style of a retro game world based on IFC as input data. To achieve this, the open source libraries *IfcOpenShell*, *Shapely* and *GeoPandas* are used in this implementation. First step is to break down the building model into semantic units. As the intended output is a 2D tile map and the input data is a 3D building model (Figure 2), only one floor, as displayed on Figure 4a, can be displayed per map. Therefore, this implementation examines the elements of a single floor within the given building model. Because the desired tile map is an abstracted replica of the actual model, not every semantic component is relevant. The relevant object types are walls (*IfcWall*), floor slabs (*IfcSlab*), doors (*IfcDoor*) and their containing openings (*IfcOpeningElement*). These are the minimum required elements to rebuild a floor as a tile map. After all relevant building elements are determined, their geometric representations need to be extracted. For the sake of efficiency, the implementation uses the 2D-representations from IFC if they exist. Otherwise, a 2D-representation is generated from the existing 3D-representation of the respective element. The details of each element’s representation are used to create one multi-geometry object (see Figure 4b), which is finally used to derive the tile map. The multi-geometry object consists of a polygon for the slab footprint, polylines for wall axes, and rectangles for the location of openings.

For the building scenario, we first developed a set of predefined tiles for wall elements. Those tiles now need to be arranged according to the floor layout. Therefore, geometric operations

[5] Original sand and rock tile set: William.Thompsonj and Daniel Eddeland 2014. OpenGameArt, `https://opengameart.org/content/lpc-sandrock-alt-colors`.

[6] Assortment of resulting island maps: `https://hlg.github.io/wamap`, Python scripts: `https://hlg.github.io/tiler`



(a) Ground plan of examined floor in Revit.

(b) Multi-geometry object of walls (black), openings (white) and floor slab (grey).

(c) Generated tile map of examined floor.

Figure 4. Conversion of an IFC floorplan (a) to WorkAdventure tile map (c) via an intermediate data structure (b).

are applied incrementally to the multi-geometry object overlaid with a grid. For each cell of the grid, the index of the corresponding tile is determined based on the numbers and direction of intersections between the multi-geometry object's elements and the grid lines. The indices are collected in a list, which, after a full run, contains all tile indices, and can be used as a data layer for the tile map. Before integrating visual and logical layers into the tile map file, algorithms for visual improvements are run through the data layer to enhance the view to 2.5D. Finally, the tile set and JSON files are uploaded to the dedicated server so that Work Adventure can retrieve the generated tile map (Figure 4c) and set it as the virtual world. We've tested this prototypical implementation on various building models with different scale. Besides the model shown in the Figures, the prototype has been applied to a model of an open plan office and to a complex model of an entire university building. The presented model and the open plan office model are custom-built. The model of the university building was created by civil engineering students. All building models were initially modelled in Revit, are available in Revit's native **.rvt* format and were exported to IFC4 (Reference View) using Revit's default export settings. As the conversion was applied to these models, it appears that the more complex the model, the more difficult it is for the prototype to reproduce the floor layout without deviations and omissions. This is due to fundamental limitations of this implementation.

The tiles created for this scenario are only representing vertical and horizontal wall sections, this means the prototype is limited to models of buildings with only orthogonal walls. Further limitation is the lack of furniture objects in the examined IFC models and therefore the created tile map doesn't contain any furniture. This leads to empty rooms in the replicated floor. Because furniture and decoration make spaces livelier, furniture and decoration could be added manually into the tile map by using a level editor. Alternatively, it may be required from the native BIM author that furniture objects are placed within the IFC model. The first prototypical implementation has shown that building models can be easily transferred into tile maps to replicate floors on Work Adventure for various purposes.

4.4 City district scenario

Similar to the previous scenarios we contemplate an orthographic projection for the city district scenario, but while the island scenario used a top-down (plan) projection for the lack of relevant height information (apart from the difference between land and sea level depicted uniquely on coastal tiles in the sand tile set) and the building scenario used a simplified axonometric projection from the front to reveal the sides and heights of all those walls with axes parallel to the projection plane, for the city district scenario we propose to use isometric projection revealing more sides and height information than the other two. Phaser is capable of handling isometric maps[7], but it is not yet fully adopted in WorkAdventure. Since this is a much requested feature, we assume that it will eventually be available and those we aim to tackle the generation of tile maps and appropriate tile sets to populate maps from CityGML LOD 2. This scenario will be more ambitious in terms of creating an adequate tile set, preprocessing the input data as well as tiling the map.

The use of 3D game engines with orthographic projection and dedicated textures could be an alternative approach to resemble the appearance of retro gaming worlds without trying to carry on the legacy of memory-efficient implementation.

5. DISCUSSION AND OUTLOOK

5.1 Prototype evaluation

Based on the findings from the prototypical implementation, we discuss and conclude whether good-quality retro gaming maps can be achieved in an automated manner or need to stay a handicraft as is the case with pixel artwork. We have shown that retro gaming maps can satisfactorily be generated from OSM and IFC input. For CityGML we still have to provide the proof of concept. The prototypical implementations are subject to limitations with regard to the semantic richness of input data and geometric complexity of the map projection. For the building scenario it may be beneficial to tackle the extraction and conversion to an intermediate format more suitable to represent indoor maps in a harmonized way separately upfront. For the island scenario, it may be useful to insert a tailored generalization procedure before the tiling. Tile set creation, however, will likely stay an artistic act. Although automatically generated tiles with more abstract and less nostalgic appearance are helpful for testing and development of algorithms and might even be able to breathe new life into retro gaming environments, we suppose user will find them not very lively and attractive.

5.2 Survey

We plan a survey among test users, which shall contribute to the evaluation of the prototypes in terms of the quality of gen-

[7] Phaser isometric example: `https://phaser.io/examples/v3/view/tilemap/isometric/isometric-test`

erated maps. In addition, the survey should answer the question of how recognizable real-world scenarios impact the user experience in spatial chat tools compared to pure fantasy worlds in various scenarios such as school, work and leisure.

The concept for conducting this survey is as following: Participants in the study are either recruited with known familiarity of respective locations or are asked for self-judgement before entering the map and filling in the survey. For example, for the island scenario in the recreational setting, they could state whether they have already been to an island for a day trip, a week-long or longer holiday stay or have lived there more than two months or whether they have read or watched some documentary. Thereafter, a location for their test run is selected such that the survey yields a balanced sample of evaluation for known and unknown locations. Optionally participants could be challenged with a task to complete and connected with other participants that have the same or different knowledge of the place to take the test run together. After a predefined period spent in the environment, users are taken to the survey. Finally users would be invited to freely use the maps, make appointments and share the links. Returning users could still be asked to fill a modified survey, but first-time user's evaluation would be most interesting.

5.3 Future work

Once the city district scenario is implemented, it will be interesting to look at integrated scenarios, where indoor and outdoor worlds are seamlessly connected. This type of world would lend itself to represent mobility topics and traffic simulation and thus broaden the scope of application of retro gaming maps generated from building and city models as well as other geospatial data. n a similar vein, navigation between maps of the same scenario would be a desirable feature, e.g. navigation between levels in the building scenario or "island-hopping" in the island scenario.

For both the islands and building further features await their implementation, such as settlements, streets, points of interest, furniture etc. From a technical perspective there are some possible improvements such as to extract a library for shared tiling algorithms across scenarios or switch to a Python TMX library for robust writing of the tiled map JSON. It would also be interesting to try and connect Phaser or WorkAdventure to a BIM server instance.

For the island scenario in particular, a more fundamental question to investigate is the to further evaluate the options of further tile set reduction and robust coverage of corner cases by using smaller tiles or applying constraint-based generalization methods. This would also involve testing with the full scale of available island polygons and treatment of archipelagos.

6. ACKNOWLEDGEMENTS

The authors like to thank Prof. Undine Kunze (HTW Dresden) for generously providing access to sample data of the university building for testing purposes and Martha Friedrich (Erklärfilmstudio) for pushing pixels to improve the "sand" tile set.

REFERENCES

Bille, R., Smith, S. P., Maund, K., Brewer, G., 2014. Extending building information models into game engines. *Proceedings of the 2014 Conference on Interactive Entertainment (IE2014)*, IE2014, Association for Computing Machinery, New York, NY, USA, 1–8.

Borrmann, A., König, M., Koch, C., Beetz, J. (eds), 2018. *Building Information Modeling: Technology Foundations and Industry Practice*. Springer, Cham.

Bowman, J., 2013. The Gameduino 2: Tutorial, reference and cookbook. `https://excamera.com/files/gd2book_v0.pdf#page=29`.

FME, 2020. How to make minecraft worlds. FME Desktop Knowledge Base, `https://community.safe.com/s/article/how-to-make-minecraft-worlds`, 29.07.2020.

Gotlib, D., Wyszomirski, M., Gnat, M., 2020. A Simplified Method of Cartographic Visualisation of Buildings' Interiors (2D+) for Navigation Applications. *ISPRS International Journal of Geo-Information*, 9(6), 407.

Konde, A., Tauscher, H., Biljecki, F., Crawford, J., 2018. Floor plans in CityGML. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences: Proc. of the 3D GeoInfo Conference*, IV-4/W6, Delft, Netherlands, 25–32.

Négrier, D., 2021. Energy consumption of web game engines. The Coding Machine blog, `https://thecodingmachine.io/energy-consumption-web-game-engine`, 20.04.2021.

OGC, USGIF, 2021. Advancing the interoperability of geospatial intelligence tradecraft with 3D modeling, simulation, and game engines (whitepaper). OGC Technical Papers, `https://www.ogc.org/docs/techpapers`, March 2021.

Rudolf, S., Blömeke, S., Sharma, P., Lawrenz, S., Scheller, C., Mennenga, M., Schmidt, K., Herrmann, C., Rausch, A., Spengler, T. S., 2020. Efficient use: An interdisciplinary framework towards the cascade use of electronics. *Proceedings of the International Congress "Electronics Goes Green 2020+". The Story of Daisy, Alexa and Greta*, Fraunhofer Verlag, Stuttgart, Germany, 460–467.

Sayidov, A., Aliakbarian, M., Weibel, R., 2020. Geological Map Generalization Driven by Size Constraints. *ISPRS International Journal of Geo-Information*, 9(4), 284.

Stephenson, N., 1992. *Snow Crash*. Bantam Books, New York.