TOWARDS EGRESS MODELLING IN VOXEL BUILDING MODELS

Ben Gorte[a], Mitko Aleksandrov[a], Sisi Zlatanova[a]

[a] GRID-UNSW, Sydney, Australia,
{b.gorte, mitko.aleksandrov, s.zlatanova}@unsw.edu.au

Commission IV / 5

KEY WORDS: 3D building models, indoor navigation, routing, egress modelling, voxel models.

ABSTRACT:
Egress (or Evacuation) modelling has the purpose to simulate how the people inside a building can move outside in a safe manner and as quick as possible in case of an emergency. The goal can be to assess the safety of an existing building or a building design, to establish evacuation routes and install signage, or even to give real-time guidance to the evacuating crowd during an emergency. In all cases, the availability of a three-dimensional building model allows to generate the necessary input for the egress modelling software. The algorithms for such software can be subdivided into two major categories, working with navigation grids or navigation networks, respectively. The purpose of this paper is to illustrate that voxel-based building models are particularly suited to provide input for both categories. We further illustrate how to process might continue using an algorithm based on (electrical) network analysis, which can be applied in both a gridded and a network-based simulation. We provide preliminary results for the gridded case.

1. INTRODUCTION

At the Geospatial Research, Innovation and Development (GRID) group at the University of New South Wales we are developing a comprehensive infrastructure for the representation of three-dimensional geographic information using voxels, next to the more common vector representation models. We strongly feel that voxel representations offer a wide range of possibilities for numerical 3d data analysis and information extraction that would be too complicated to perform on the basis of vector data. An example of such an analysis might be path finding, in the context of indoor navigation, as we recently described in (Gorte and Zlatanova, 2019, Koopman 2016). In the current paper, we want to carry this further into the application of evacuation modelling, addressing the problem of how a crowd of people egresses from a building in case of an emergency.

Evacuation (or Egress) modelling may generally be performed for two different purposes, which we will call simulation and guidance. Simulation is meant to predict how a crowd of a given number of people with a given spatial distribution over the building behaves in an emergency, while evacuating the building via regular and emergency exits. It is usually assumed that the situation in the building, for example concerning the available exit routes and the way in which these are indicated by signage, is static. Furthermore, a number of assumptions about the crowd's behaviour is being made, for example concerning: how quickly do people start moving after an alarm is given; is everybody moving precisely along the shortest route to the nearest exit or can they make detours; do they obey exit signs; do groups always stay together or will they split if they do not easily fit through some corridor and alternatives are available (Helbing and Molnar, 1995). Studies towards the validity of those assumptions are available in literature (Daamen et al 2017), although they are difficult to conduct, since (for example) the induction of a realistic level of 'panic' into an experimental setting is unethical.

The purpose of guidance is to optimize the process by giving instructions to the evacuating crowd (Spartalis e.a. 2014). Even when the crowd behaves 'well', each individual cannot predict to which extent doing 'the right thing' may lead to congestion later in the process. Such a prediction would require that everybody has full knowledge about everybody else's current position and future behaviour, and of the full consequences of planned and alternative behaviours. This cannot be expected from people, but in the presence of sensors that monitor the situation, software may be able to make such predictions and prevent sub-optimal situations by directing people away from them. An additional advantage is that the software can take the dynamics of the emergency into account, for example by excluding routes from the solution that are currently unavailable. The above description, however, already suggests that finding the optimal solution is challenging – it is an NP-hard problem indeed. Another challenge is how to communicate the results of the algorithm as directions to the crowd (by dynamic signage, announcements, smartphone apps etc.) and how to make people obey those.

Many approaches towards evacuation modelling have been studied (Kulagowski et al. 2010). Evacuation modelling on the basis of an available 3d model first requires the building model to be translated into a representation that is suitable for the selected algorithm. There are two major approaches, a gridded and a networked one. In the gridded approach, the floorspace of the building is subdivided into square tiles of (for example) 50cm x 50cm, in which people, modelled as 'agents' in certain software approaches, move from tile to tile (Wirth and Szabó, 2017). Also cellular automata approaches make use of tiling (Kirchner and Schadschneider, 2002).

Alternatively, a network of possible paths, i.e. a graph consisting of nodes and edges, is derived from the building model. People enter the graph at the nodes, according to the assumed distribution of the 'input crowd' around the network, and then move from node to node along the edges, as prescribed by the algorithm (in both the simulation and the guidance case) to finally reach an exit node. Edges can have capacities attached to them, depending on widths of stairs, doors and corridors in the 3d model. Algorithms are often based on physical (hydrodynamical) models and the notion of a crowd as a "thinking fluid" is coined by (Hughes, 2003).

The purpose of the current paper is mainly to show that both approaches (gridded and network) can have their inputs easily derived from a voxel building model. *En passant,* in order to

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-IV-4-W9-43-2019 | © Authors 2019. CC BY 4.0 License.

illustrate our findings, we introduce an evacuation modelling approach based on Kirchhoff's first law for electrical circuits, which can be applied to gridded and networks models alike.

We feel that the gridded approach (a) may be the more suitable one in general and (b) fits better to the voxel representation of our input model – this formulation is meant to leave room for discussion and is not intended as a conclusion or a verdict. We introduce both approaches but work out the gridded approach in greater detail.

2. OHM-KIRCHHOFF EGRESS MODELLING

Ohm's law relates the potential difference (voltage) U over a resistor with resistance R to the current I flowing through the resistor as $U=I*R$. It can be applied to compute the equivalent resistance of two serial resistors as R_1+R_2, or of two parallel resitors as $R_1R_2/(R_1+R_2)$. In complex configurations a more general approach is needed to compute the equivalent resistance of the network. Figure 1 shows a network with four nodes (circled numbers), connected by five edges containing resistors (rectangle with indicated resistances). If we set the potential at node 0 to be 0V, and the potentials at nodes 1, 2 and 3 as 8V, 12V and 20V, respectively, then by Ohm's law the currents through the edges can be easily computed (Fig. 1). According to Kirchhoff's first law, the sum of the (signed) currents at each node is 0. This is satisfied at nodes 1 and 2, but at node 3 there is a deficit of 12A, which has to come in from an additional edge (the arrow). Similarly, a surplus of 12A goes out of the network via an additional edge at node 0.

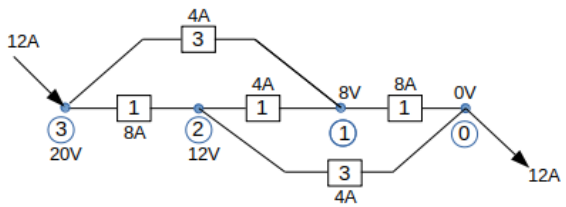

Figure 1. A simple resistor network, where nodes have potentials (measured in Volt) and edges have current (in Ampere) flowing through resistors with resistances in Ohm.

The relationship with egress modelling is that the network represents the available evacuation routes, where the lengths of the edges are represented as resistances. Currents express people's movements. Therefore, the shortest route between two nodes corresponds to the path with the smallest equivalent resistance in the network. However, as the example shows, currents do not necessarily follow shortest paths, but get distributed over the network in such as way that the total (equivalent) resistance of the entire network is minimal. Here, it is 5V / 3A = 5/3 Ohm, whereas along the shortest path it would have been 3 Ohm. In egress modelling terms this means that the capacity is maximized, which can be considered a 'good behaviour' of the currents. This behaviour implies that groups will split, e.g. when noticing that a 'preferred' corridor is congested while alternatives are available.

The remaining issue is that in an egress simulation only the 'incoming current' is known (and therefore also the outgoing current). The voltages are unknown, as are the currents along the edges. How to compute these? The algorithm treats the input/output currents at the nodes as observations and the potentials as unknowns. We set up a 'forward model' that can derive the observations from the unknowns if they were known: from the potentials and resistances over the edges we compute currents and add them up at each node, as above, to get an equation that yields an 'observation' at each node. With N nodes, we have N-1 potentials (the Nth one being 0) and N observation equations, which are solved by adjustment. They have a unique, exact solution when the sum of the incoming currents equals the sum of the outgoing ones – otherwise, we would get a least-squares approximation. Therefore, this does yield us the voltages that we weren't interested in, but meanwhile also the currents became known.

It gets even better in Fig. 2, where we have incoming currents at each node, representing people distributed all over the building, except for one node with an outgoing current at the building exit. Again, the figure shows an 'optimal' distribution of currents over the network. (The voltages are 25V, 33V and 40V at nodes 1, 2 and 3, respectively.)

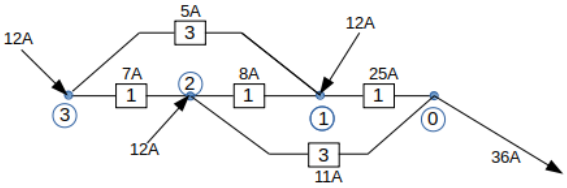

Figure 2. The network of Fig. 1 with input currents at multiple nodes (1, 2 and 3) and a single output at node 3. This is closer to an egress scenario.

2.1 Maximum-current extension

On a side note, it is possible to extend the above algorithm by putting a limit to the maximum current in each edge, in order to try to ensure that maximum capacities are not exceeded. The algorithm will then send more current through alternative routes. The way this works is by adding a variable amount to each resistance depending on the current in the edge (or, equivalently, on the potential difference over the edge): for small currents, the resistance is as usual, but when a current approaches the limit, the resistance increases exponentially. The least-squares inversion now becomes non-linear and needs to be solved iteratively. Fig. 3 shows the result when the limit for current is set to 20A in each edge.

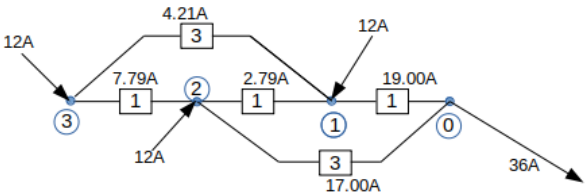

Figure 3. As Fig 2. but with a different distribution of currents, being maximized at 20A per edge.

2.2 Time Dependency

The results of the algorithm can be interpreted in two different ways. The first interpretation is that there is a certain batch of people to be evacuated; they move along the different edges of the network in an optimal way, and the algorithms correctly predicts how many people will have travelled along each segment when the evacuation is over. In the second interpretation, a given number of people *per unit of time* is moving towards the exit, in a continuous fashion that resembles water flowing through a network of pipes. Now the algorithm correctly tells the flow of

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-IV-4-W9-43-2019 | © Authors 2019. CC BY 4.0 License.

people travelling along each segment per unit of time. Both interpretations, however, do not correctly reflect an evacuation, where we are interested how *a single batch* of people moves through the network *as a function of time*. During the process the traffic in each edge fluctuates, and rather than knowing the *total charges* through edges we would like to know the *maximum* (over time) of the *currents*.

In order to learn about the time-dependent flow inside a segment we consider which are the nodes that will eventually contribute to it. All these nodes have different distances to (the start node of) the segment. By assuming that the flow has a constant (walking) speed, it can be predicted after how much time the contribution from each node will arrive at the segment. In other words, the histogram of distances of the input currents at the nodes contributing to a segment yields the time-dependent flow through that segment. Such methods are used in Hydrology to estimate the unit hydrograph of a river catchment during run-off modelling on the basis of a DEM (Ramirez, 2011).

In DEM run-off models the river network forms a tree in which the DEM cells are leaves: each cell follows a unique path towards the root of the tree. The same would apply if we modelled the evacuation along the shortest paths from each node to the exit. Dijkstra's shortest path algorithm, for example, transforms a (directed) graph into a tree. In our Ohm-Kirchhoff method, however, a node may contribute to several subgraphs. This leads to weights being taken into consideration while establishing the distance histograms of nodes for each edge. The algorithm is implemented recursively: starting at the exit node it works its way upstream through the network. Nodes may be visited several times; the first time the result is stored in a list of histograms, one per node, from which it is taken directly at subsequent visits.

3. ANALYZING A VOXEL BUILDING MODEL

In the next section we will discuss how to extract input for egress modelling from a voxel building model. In general, egress modelling requires input about the building and about the people inside it (at least their number and the distribution over the different rooms, perhaps even about other characteristics, such as age, disabilities, familiarity with the environment etc.). We would like to stay away from that, and assume people are evenly distributed over the building: the number of people in each room solely depends on the room size. As an advantage we could imagine that the generated result pertains to the building and is independent of varying building usage. It is a constant characteristic of the building, describing something like the "building egressability". One could also imagine that for a sports arena, a school or a cinema this characteristic should be "better" than for a residential building.

A 3d voxel model of the Arboretum Professional Centre building in Seattle, Washington, USA was created on the basis of publicly available floorplans, retrieved from the website of the building management, http://coho-apc.com/building-floor-plans. The model contains the floors, walls, stairs and roofs of the building; other building elements, including doors, were removed from the floor plans before going 3D. The voxel resolution of the model equals 4.55 cm (Fig. 4).

A 'navigable space' is created by constructing a 5-voxel (22.75cm) thick layer of voxels onto every horizontal surface using a dilation operation. This ensures that on a stairway the top of the layer on one step gets adjacent to the bottom of the layer on the next-higher step, and therefore the entire stairway gets covered by a single connected component in the navigable space.

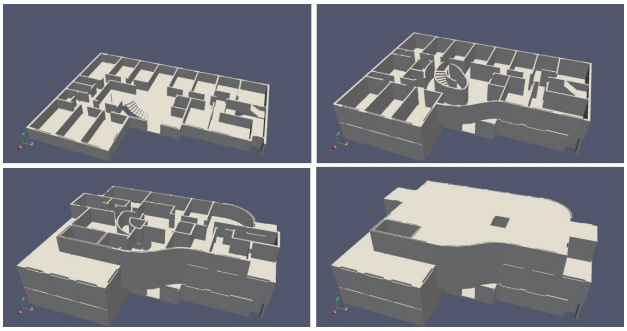

Figure 4. Sample three-storey building "Arboretum Professional Centre" at Seattle, WA (USA)

In fact, the entire set of layer voxels is submitted to a 3d connected component labelling operator. This yields one very large component that extends over all the floors in the entire building (Fig. 5), plus several smaller components (mostly located between a ceiling and the next floor) which we discarded.

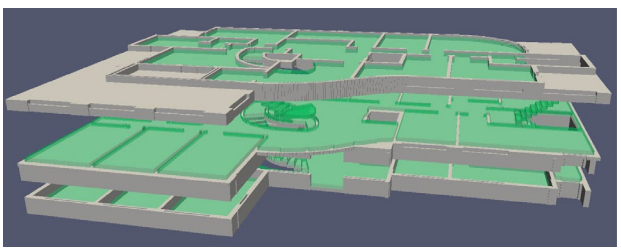

Figure 5. The (connected) navigable space in the sample building

Next we did four iterations of erosion on this largest component, taking away voxels from the top of the layer, if this does not break the connectivity with neighbouring voxels. As a result, we obtain an approximately one-voxel thick layer just above the building's floor voxels. A final clean-up was done using the surface skeleton algorithm of (Palagyi, 2002)

The result is a representation of the entire floor space of the building, extending over all the storeys, in single connected component, consisting of a minimal number of voxels (Fig. 6).

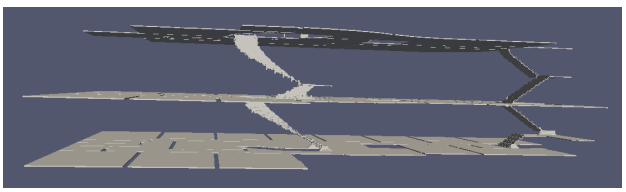

Figure 6. The surface skeleton of the navigable space

Based on this result, we can choose to continue in a network approach, or in a gridded approach.

4. SURFACE SKELETON TO NETWORK

We extract a line skeleton from the surface skeleton (Palagyi and Kuba, 1995) (Fig. 7). (We could have extracted both the line and the surface skeleton independently from the eroded navigable space, but then the line skeleton might not be fully inside side the surface skeleton).

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-IV-4-W9-43-2019 | © Authors 2019. CC BY 4.0 License.

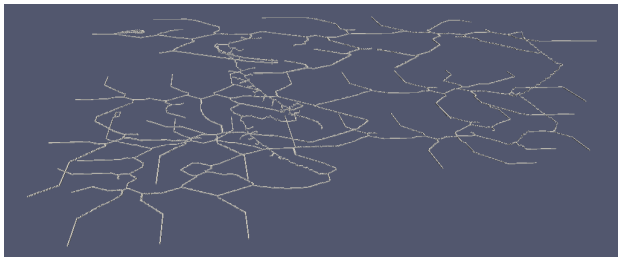

Figure 7. Line skeleton

The line skeleton represents the navigation network. In this network nodes and edges can easily be identified (Fig. 8). The values of the edges should represent their lengths, which can be obtained by counting the participating voxels. We consider that 4-connected voxel pairs contribute less to the length than 8-connected ones. Also, it is considered that vertical movement leads to larger resistance than horizontal, and therefore to a higher 'perceived' distance. We have not yet fully completed these kinds of calibration.

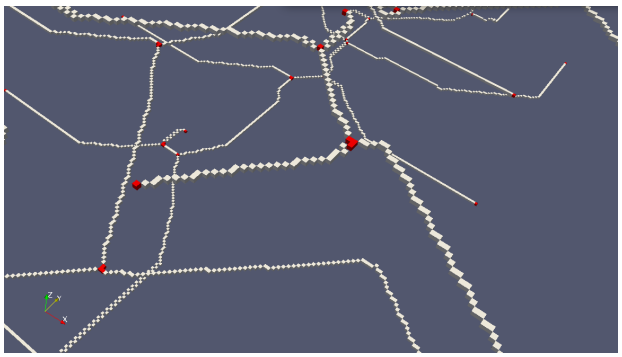

Figure 8. Detection of nodes and edges needed to derive the navigation network from the line skeleton

The *input current* at a node is supposed to represent the fraction of the crowd entering the network at that node. This is estimated from the area of the floor space to which that node is the closest node. We identified 250 nodes in the line skeleton, separated into leaf nodes (at the loose ends of edges) and connection nodes (where three or more edges meet). To determine the input current at each node, we subdivide the floor space (i.e. the surface skeleton) into 250 different parts by assigning to each surface skeleton voxel the value of the nearest node in the line skeleton and afterward counting those voxels into a histogram. The assignment is accomplished by a modified distance transform in 3D. A distance transform (Borgefors, …) assigns to each background pixel in a raster data set (an approximation of) the distance to the nearest object pixel, by applying a two-pass recursive 3x3x3 neighbourhood operation. We made two extensions to this algorithm. First, we restrict the distance propagation to take place through a subset of voxels (the surface skeleton), rather than through the entire space. Therefore, distances are measured along the surface skeleton, and the influence of a node pixel cannot 'jump' through the empty space to other parts of the surface. This makes the algorithms multiple-pass instead of two-pass. Secondly, we not only propagate distances (to nearest nodes), but also id-s (of nearest nodes), and assign these to another output data set, which can be considered a Voronoi diagram of the surface skeleton (Fig. 9), from which the above-mentioned histogram is derived.

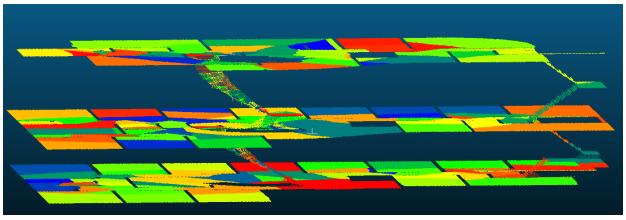

Figure 9. Subdivision of surface skeleton into a Voronoi diagram according to the nearest node in the line skeleton

It is easy to see how this approach would continue: by assigning some nodes as exits and submitting the network, having edge lengths and node input currents, to the algorithm of Section 2. Still missing, however, is an estimate of the capacity of each edge, which could be used either as a maximum current in the non-linear adjustment of Section 2.1, or as a criterion to be compared to the histogram-method for flow over time (Section 2.2) . We think such an estimate could be obtained by making another Voronoi diagram of the floor space, now on the basis of edges rather than nodes. Given that line skeletons share geometric characteristics with medial axes, we can assume that the distance from an edge to the outside of its Voronoi region relates to the width of the path represented by the edge. Those distances can be established in yet another distance transform, but this has to be further investigated.

5. SURFACE SKELETON TO NAVIGATION GRID

Our gridded approach also sets up a network. Each grid cell (voxel) of the surface skeleton is a node, and edges are links between adjacent skeleton voxels. A cell could in theory have 26 neighbours, but in a surface skeleton this is unlikely. Many voxels on the building floors have eight neighbours, however.

Current will flow along edges from node to node, i.e. from voxels to neighbouring voxels. The resolution of our voxel model being 4.55 cm, such a very detailed modelling of people's movements would be exaggerated and computationally demanding. Therefore, we sub-sample the surface skeleton to a courser resolution: in this case with a factor 9, leading to a 41cm voxel size. This needs to be done with care, in such a way that the relevant objects (floors and stairs) remain present, but also the holes and gaps in the surfaces (caused by walls in the 3d model) remain. Although slightly experimental, we obtained good results by sub-sampling floors and stairs separately after 9x9x1 'erosion' and 9x9x9 'closing' operators and combine the results (Fig. 10).

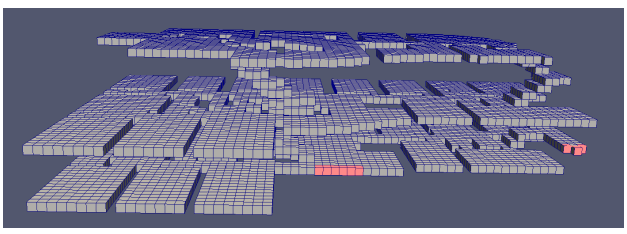

Figure 10. Surface skeleton, subsampled to 41cm resolution, to be used as a navigation grid. Red voxels indicate main and emergency exits.

We obtained 5121 voxels and numbered them as nodes 1 to 5121. Designated nodes, shown red in Fig. 10, were marked as exits and were linked to a 'virtual' exit node with number 0. All other nodes were connected to each other by edges according to their adjacencies. We assigned lengths (resistances) of 3, 4 or 5, depending on voxels being 4, 8 or 26-adjacent; the links between

the exit nodes and node 0 have a just-above-0 resistance. Next we assign an input current of 1 to each node (1 .. 5121) and an output current of 5122 to node 0, and run the algorithm of Section 2.

This causes currents being generated through all edges; the result of summing these up (unsigned!) per voxel is shown in Fig. 11.

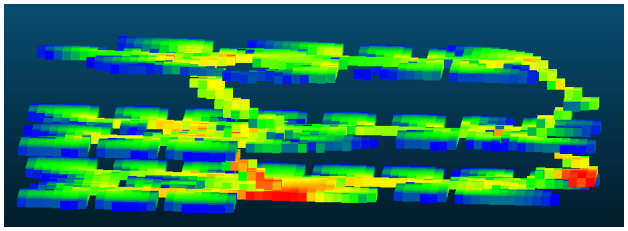

Figure 11. Current flowing through voxels during evacuation

Figure 11 shows how the evacuation proceeds according to the Ohm-Kirchhoff method, without taking the distribution of flow over time into account. To get this distribution we apply the algorithm of Section 2.2, computing distance histograms of nodes to all contributing 'upstream' nodes. For the busiest node (one of the red exit nodes in Fig. 11) this gives the distribution of Fig. 12.

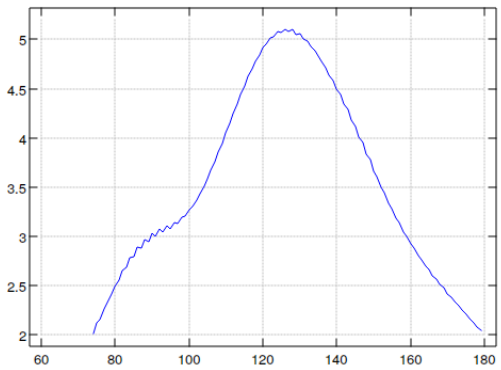

Figure 12. Current as function of time in the busiest voxel of Fig. 11, according to Ohm-Kirchhoff model

For comparison, we made the same computation for an evacuation strategy where movements are strictly according to the shortest paths to the exit (Fig. 13). Then, the average time of arrival is a bit earlier, but the total flow is much larger, whereas in Fig. 11 the traffic is spread out over a wider corridor of voxels.

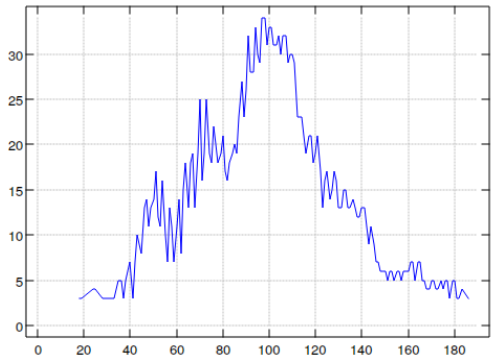

Figure 13 Current as function of time in busiest voxel in the building, according to shortest path evacuation model

It should be noted that all quantities shown are dimensionless numbers. They are expected to have (linear) relationships to relevant variables, such as crowd densities, speeds and durations, but the conversion factors have not been established yet.

6. CONCLUSION

We have demonstrated that a 3d voxel building model can be a good basis for studying various aspects of Egress modelling, using different approaches. The information required for egress modelling, either gridded or networked, can be readily extracted from a voxel model, using straightforward and well documented operation steps.

Furthermore, as an illustration, we introduced a new egress modelling approach, which may have potential in itself; this however needs further investigation. In principle, the approach works in both the networked and the gridded case. For networks we would have to refine the estimation of edge capacities, whereas a remaining issue in the gridded case is the size of the problem. The example shown required inverting a 5121 x 5122 matrix, which is feasible, but the sample building is not particularly large.

REFERENCES

Ramirez, J.A., 2005, Syllabus Engineering Hydrology, Ch 11, Prediction and modeling of flodd hydrology and hydraulics, Colorado State University

Palágyi, K., Kuba, A., 1998: A 3D 6–subiteration thinning algorithm for extracting medial lines. Pattern Recognition Letters 19, 613–627

Palágyi, K., 2002: A 3-subiteration 3D thinning algorithm for extracting medial surfaces. Pattern Recognition Letters 23, 663–675

Koopman M., 2016, 3D path-finding in a voxelized model of an indoor environment, 2016. MSc Thesis Geomatics TU Delft, Netherlands.

Kulagowski E.D., R.D. Peacock and B.L. Hoskins, 2010, A review of Building Evacuation Models, 2nd Edition, Technical Note 1680, NIST National Institute of Standards and Technology, U.S. Department of Commerce, 36p.

Hughes, Roger L., 2003. "The flow of human crowds". Annual Review of Fluid Mechanics. **35** (1): 169–182.

Helbing, D., Molnar, P., 1995 Social force model for pedestrian dynamics. Physical review E 1995, 51, 4282

Spartalis, E.; Georgoudas, I.G.; Sirakoulis, G.C., 2005, Crowd Modeling for a Retirement House Evacuation with Guidance. In Cellular Automata, Springer, pp. 481–491.

Kirchner, A, Schadschneider, A. (2002). Simulation of evacuation processes using a bionics-inspired cellular automaton model for pedestrian dynamics. Physica A: Statistical Mechanics and Its Applications. **312** (1–2): 260–276.

Wirth, E. Szabó, G. 2017. Overlap-avoiding Tickmodel: an Agent- and GIS-Based Method for Evacuation Simulations, Periodica Polytechnica Civil Engineering. **62** (1): 72–79.

Daamen W., C. Buisson, S. P. Hoogendoorn
Traffic Simulation and Data: Validation methods and applications, CRC Press 2017 262 Pages