Modelling and Simulation of Student Pedestrian Traffic at University Campus

Martin Bruckner[1,3 *], Shabnam Tauböck[1], Nikolas Popper[1,3], Stefan Emrich[2], Benjamin Rozsenich[1], Salah Alkilani[1]

[1] Inst. f. Analysis and Scientific Computing, [2] RED Real Estate Development, Vienna University of Technology
[3] dwh Simulation Services, Neustiftgasse 57-59, 1070 Vienna, Austria; **martin.bruckner@dwh.at*

Abstract. <more-space> is a software project of Vienna University of technology for supporting the planning phase of 'University2015' - a project to renovate all university buildings and to improve the existing infrastructure and the inherent processes. <more-space> determines and evaluates the spatial resources and time needs required and introduces a model for the room management that can simulate the usage of resources to optimize the planning of rooms and schedules. This contributions presents a part of the <more-space> project, the dynamic simulation for the student pedestrian traffic, which takes into account time requirements for reaching different lecture hall buildings and lecture halls on an individual basis, allowing also to incorporate needs for physical handicapped persons. Modelling and simulation of this dynamic individual student traffic is based on advanced cellular automata model, which are integrated into other <more-space> components (distributed DEVS model).

Introduction

The aim of the described simulation project as a part of the simulation system <more space> is to simulate the movement of students between lecture rooms, attending their regular curriculum to implement dynamic vacation times. Major outcome is the calculated time which they need to move from a starting point (for example an auditorium) to another location (arrival point).

The program is realized in the object-oriented programming language JAVA and connected to *Enterprise Dynamics*. Modeling approaches are Cellular Automata (CA), Agent Based Modelling (AB) and discrete simulation because the literature of this approach is widely spread, and after analysis of the project, the cost-benefit calculation for this modeling was the best. CA and AB approach can manage the dynamic behavior of the students finer and more efficient.

The dynamic model implemented in *Enterprise Dynamics* is the main model and simulation system including the data model, process descriptions and dynamic behaviour as using of resources depending on different system or environmental dependencies.

The described model for simulation of dynamic behaviour is implemented in the object-oriented pro gramming language Java and connected via TCP/IP with Enterprise Dynamics. To model the dynamic behavior of single individuals an agent-based system was chosen in which the individuals move on a discrete grid. The cell size is 0.125m x 0.125m, so that one m^2 consists of 64 cells. Each student takes 4 x 4 cells, or 0.5m x 0.5m. The forward movement of people in a building depends on several interrelated factors. Some of these are e.g. the density of people in a group, or the maximum speed which varies for each individual. Of course these factors are conditioned by the environment; for example moving into a room area or a staircase makes a noticeable difference. It is also of crucial importance, whether a person is facing multiple other individuals. All this is relevant for the speed and thus for the required time which the students need to switch from location A to another location B. One special aspect of this project was to integrate the possibility to simulate the path for physical handicapped persons.

1 Modelling

Basically, this simulation consists of two simulators. On the one hand Enterprise Dynamics, a commercial Software from INCONTROL based on the DEVS (Discrete Event System Specification) formalism in which the optimization of the room utilization is modeled. On the other hand a proprietary development in JAVA which provides the times who are needed from the students to changed the lecture rooms.

SNE Simulation Notes Europe – Print ISSN 2305-9974 | Online ISSN 2306-0271
SNE 22(2), 2012, 95-100 | doi: 10.11128/sne.22.tn.10131

This part shall ensure that the time between two teaching units, who are maybe in different buildings, is sufficiently dimensioned. In order to implement the task a cellular automats (CAs) model was used. The main components of this model is a discrete plane and the individual agents, or as in our case, the individual students who are moving on this grid, and whose decision on their further action depends on the surrounding agents. Because of the fact that the university area is too big to display to only one of such grids, the buildings are divided in logically coherent parts that are connected at several points to give the agent the possibilities to change these discrete planes. In order to allow the students to move through the huge number of planes in the shortest possible way a combination of graph theory and a kind of scalar field is used.

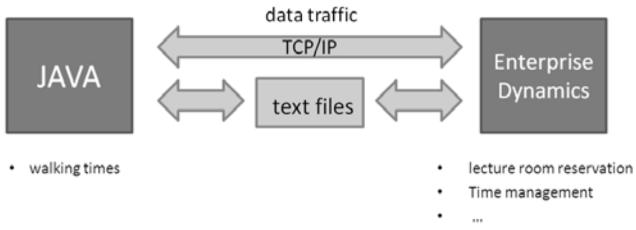


Figure 1. Data traffic.

2 Building Plans

The import of the building structures works via special edited planes. The original plans exist in the common used Autocad format .dwg. The editing is explained in the following shortly.

- Unnecessary information like dimensioning information, energy and water installations and many more are removed because it has no importance for the simulation and disturbs the clearness of the plans.
- The particular parts of the building will be colorized in one of the ten defined colors. (corridor, up- downstairs, wall, outdoor private or class room, no pass, window, text) This is necessary because the importer of the plans distinguishes the parts of the building only by their colors.
- In the next step the complete colored image will be exported to an image file in the Portable Network Graphics (PNG) format. The resolution of the graphics is chosen such that each pixel represents just 0,125m x 0,125m in reality, what exactly corresponds to the simulation of a cell of the CAs. For each pixel, the three primary colors of the RGB color space (red-green-blue) and in addition the alpha value are stored. The range of values for each of these four channels is 0-255, which is 8 bit consuming. This relative roughly resolution is also a reason why it is imported to remove the unnecessary and smallest details out of the plan as mentioned in the first point because this information degenerates after the export most to one dot.
- The last working step is to insert, with a special program, the so called Change Areas, Elevator Areas and Doors. This software tool is furthermore important because here the declaration of the buildings, CAs, rooms and many more will be performed and finally stored in the project file. This Comma-Separated Values (CSV) project file is in a way the “brain” of the data set.

The transformation of an original to an edit plan is depicted in the following pictures.

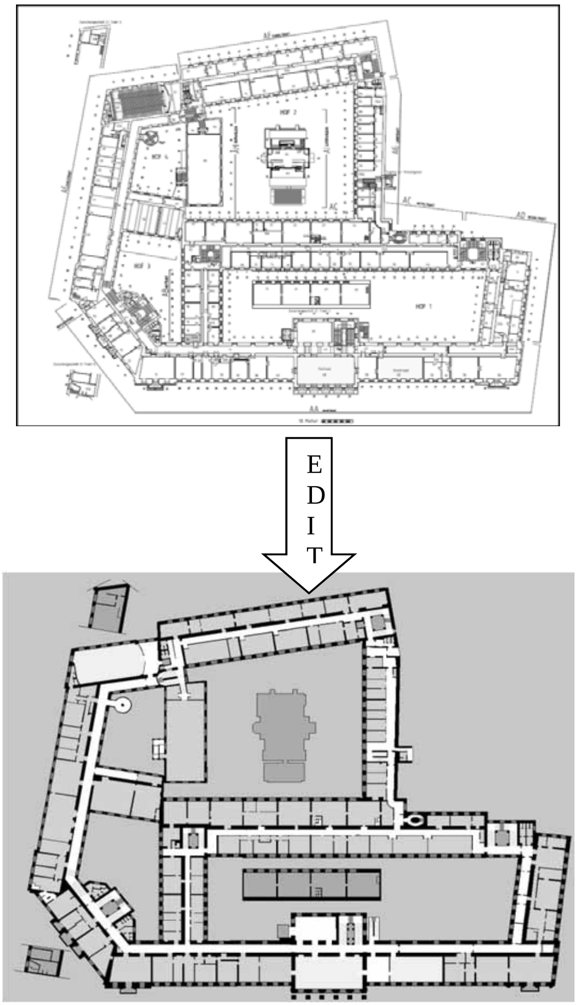


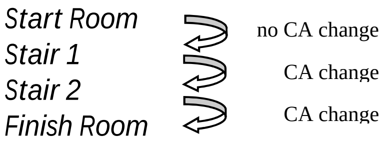
Figure 2. Example of a .png file (TU-Vienna, main building, first floor).

3 Routing

3.1 Global Routing

To simplify student´s life they need a plan how to get from point A to point B and if possible in the shortest way. To ensure that this happens, the graph theory was used. With aid of images and CSV files a connected undirected and weighted graph is generated.

- Node: Basically it exist only two types of nodes, doors and change-areas which includes also the elevator-areas. In case of doors we consider only the entrance of the lecture-rooms and building-exits. Doors, such as fire or corridor doors, are not in use. Only a small narrowing in corridor handicappes the persons. The change-areas give the individuals the ability to change the CAs, for example at the end of a stair or in case of an elevator to change the floors.
- Edges: In every CA each node is connected via edges with each other, whereby each edge has two weights, the distance in meter and the time which an average human needs for this way without obstructions.



Start Room no CA change
Stair 1 CA change
Stair 2 CA change
Finish Room

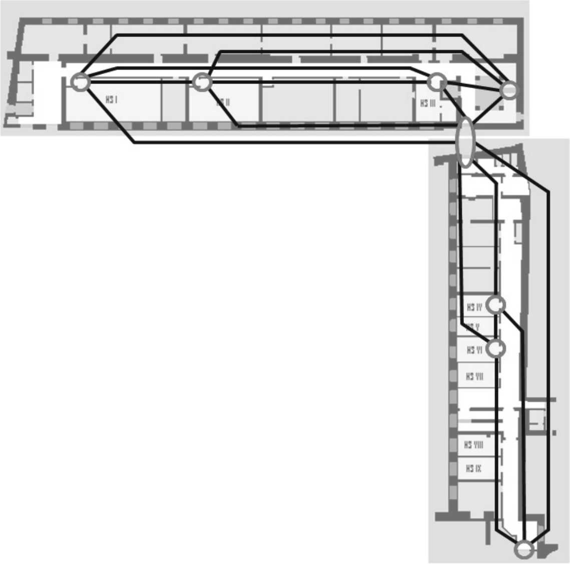


Figure 3. Example of a graph.

With the aid of this nodes and edges and a slightly modified Dijkstra's algorithm it is now possible to find a path, which has a good combination of shortest and fastest way, through the building.

Considering the shortest path is not always the best. For example when people have to wait for some time in front of an elevator instead of using the stairs. This algorithm provides now a list which looks like the following.

3.2 Local Routing

The aim of the local routing is to move the individuals in the simulation the shortest way through the current Cellular automata. To implement this, the simulator uses a Static Floor Field. As the name suggests, this is a field of static values, which contains the distances, to the various nodes, in cells. This field is embedded into the CAs. So it is possible to “ask” each cell: “How far is it to the door with name doorLectureRoom8?”

To determine now the shortest path, for each direction the associated cells will be scanned and afterwards they calculate the arithmetical average. So the direction with the smallest distance works in favor of the next step. To initialize this data structure, a modified variation of the Flood Fill algorithm is used. This is a simple procedure of Computer graphics; in which the bording areas are colored.

6	6	6	6	6	6	6	6
5	5	5	5	5	5	5	5
4	4	4	4	4	4	4	4
3	3	3	3	3	3	3	3
2	2	2	2	2	2	2	2
2	1	1	1	1	1	1	2
▨	▨	0	0	0	0	▨	▨
▨	▨	-1	-1	-1	-1	▨	▨

Figure 4. Static floor field.

Figure 4 shows an example from a static floor field and shall give a better view of the keynote. The yellow cells in the centre below illustrate a door bordered left and right from a part of a wall. A cell with the distance 0 indicates a starting point of the door. The remaining numbers show the distance of the associated cells to this starting points.

4 Deadlock

- React to other students: The first and perhaps most effective method is the timely reaction of contra flow or other barriers. To implement this, a searching area of approximately 1.5 meters wide and 8 meters in length will be scanned, while all found students are divided in three groups according to their relative walking direction; (1) traffic, (2) cross traffic, (3) oncoming traffic. This subdivision will be considered by the selection of the next step. For example, when overtaking a student, or make room for oncoming traffic.

- Jostle other persons: In order to allow a necessary change of direction even if another student who walks parallel and in the same direction and prevents jostling, each individual has the possibility to make a request to the neighbor to make room. The student can afterwards decide if he obeys this suggestion or not depending on his or her possibility.
- Right-hand walking: The next algorithm to avoid collision is the preferred passing of a contra flow on the right side. This behavior can be observed repeatedly in public and is also described in the literature by various experiments.
- Shrink Student size: Another very useful method is to imitate the behavior of people coming into the situation that the required space is too small for "normal" walk. We instinctively shrink our space, for example by rotating our shoulders. This shrink is also implemented in this model. Each student normally requires 0.25m2 or 4 x 4 cells in simulation. This space may, if the situation requires it, be reduced to 3 x 3 or 2 x 2 cells.
- Waiting in front of the auditorium: Further potential sources of blockages are the entrances of the lecture rooms. So arriving Students are waiting outside of the room until the room and therefore the doors are free.
- Second level: The last algorithm to resolve deadlocks is a second emergency level. About this second level blockade can be reduced and it will ensure that the students reach their specified destination.

5 Walking Speed

The walking speed of people is influenced by different factors. To simplify, the simulation acceleration and deceleration will be ignored.

5.1 Depending on density

A crucial factor of moving forward is the density of the traffic. The relation of density and speed is described with the so-called Kladek formula, which will be described in the following part.

v_h^0 Max. speed, depending on the underground.

ρ^{max} Max. density, 5.4 P/m^2

γ calibrate constant (empirical determined), 1.913.

$$v_h(\rho) = v_h^0(1 - exp^{\left(-\gamma\left(\frac{1}{\rho}-\frac{1}{\rho^{max}}\right)\right)})$$

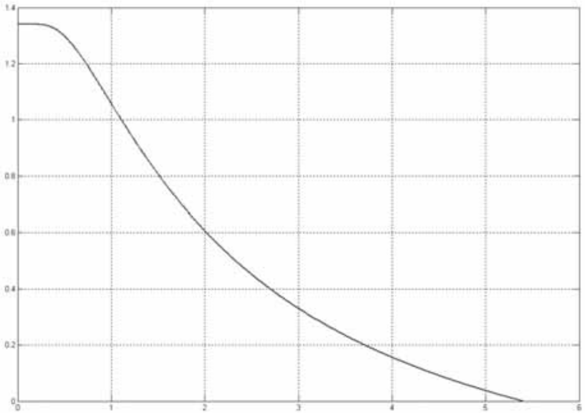


Figure 5. Relation of Kladek v_h^0 =1.34 m/s.

5.2 Depending on underground

The ground where the persons are moving is also an important factor. To simplify this model only three different types are defined. These values are the v_h^0 for the mentioned Kladek formula:

- Corridor 1.34 m/s
- Upstairs 0.61 m/s (only horizontal)
- Downstairs 0.71 m/s (only horizontal)

Because the size of the students varies, but it is in any case bigger than one cell, it is possible that they occupy cells with different speed-values, for example when entering a stair. To deal with this problem, the maximum speed v_h^0 is the arithmetic average of all cells.

5.3 Depending on random speed

In addition to this speed calculation each student has a random speed factor which is generated with a special probability distribution. This factor will be multiplied with the final speed value to bring the simulation closer to reality.

6 Calculation of the Next Step

Afterwards there exists a brief overview of the calculation of one time step. In the simulation one time step is 250ms. Please note that a time step is not the same as a cell shift. One time step has usually a few cell shifts.

- Data ascertainment: density, traffic, …
- Calculate next direction: with the aid of the collected data the next direction for the cell shift is chosen.

- Calculate speed and time: calculate the possible speed in this direction and the resultant time for this shift.
- Repeat 1-3: The first three points are repeated until the difference of the sum of the single shift time and the simulation time, in our case 250 ms, is minimal.

$$\left|\sum(shift\ times) - \ 250ms\right| = minimal$$

- Calculate discretization error: the difference of the above formula is the start value in the next iteration. This helps to minimize the discretization error.

The following example shows what happens without consideration of this discretization error.

Student speed = $1,34\ \frac{m}{s}$

Step time Δt = $0,25\ \mathrm{s}$

From this it follows a distance of

$$1,34\frac{m}{s} * 0,25\,s = \ 0.335\,m$$

Possible discrete distance are $\frac{2}{8}m = \ 0,250\,m$ or $\frac{3}{8}m = 0,375\,m$

To minimize the error we chose the value $0,375\,m$.

Herewith the resultant absolute error is

$0,375\ \mathrm{m} - \ 0,335\ \mathrm{m} = 0,04\ \mathrm{m}$ or a relative error from

$$\frac{0,04\,m}{0,335\,m} * 100 \approx \ 12\ \%$$

Moving: In the last step the precalculated shifts for each student are performed. This happens with the no crossing path method. This technique allows each student to move on its precalculated way until he has performed all shifts or crossed a trajectory of another individual who has already moved.

7 Elevators

An important part of each house, which has more than a few floors, are the Elevators. This element of the simulation is needed because without elevators the simulation is not really complete and indispensable for the simulation of persons with physical handicaps. To control the activity of the elevators an event-based approach was used. An elevator has seven distinct states (wait, move up, move down, open door, close door, person get in, person get out). Each of them serves exactly one purpose during operation (no superstates).

After an event has completed, the controller selects a new task depending on which car and floor buttons have been pressed. A flow diagram which shows all possible transitions from one state to another is displayed in the figure below.

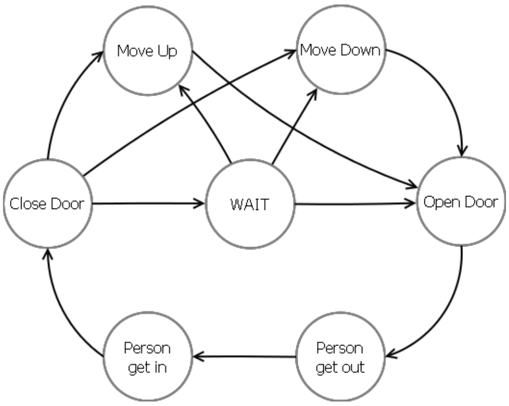


Figure 6. Possible elevator transitions.

In the course of implementation, we have found that the control of clustered elevators is not as easy as it seems at the beginning, since it is not always clear which elevator must be directed to which floor. For example, a floor call ordered an elevator in his floor. Now it possible that one elevator passes this floor anyway while another elevator is waiting at the ground-floor.

Now, shall the second elevator process the call or should the first elevator interrupt his ride to pick up the waiting passengers? This is just one question for the developer of the elevator controller. In order to keep this part of the simulation easy a relative simple algorithm of a controller is in use. He has only two major rules:

- If an elevator is waiting on a floor, send him to the waiting passengers.
- If no waiting elevator is available stop the next cabin that passes the floor.

With those two points the control of the elevator works surely not in the best an efficient way but the algorithm is easy enough to implement and adequate for our needs.

The next big problem witch the elevators indirectly creates was that each person in the simulation strives to take the shortest and supposedly the best path through the building. We have defined the length of paths between floors when using elevators as zero. As a matter of fact, the shortest path through the building is always by the way of the elevators, but this does not have to be the fastest way. To handle this problem a combination of shortest and fastest path is used. To calculate the time via the elevators each person who used one informs the controller after his ride about the waiting time. With the aid of this data that is stored in files for following simulation runs an estimation can be calculated to give the persons a pointer for the approximate waiting time.

A special potential hazard for collisions between people and following deadlocks are the entrances of the elevators. To disarm this each person, who wants to take the elevator, registers him- or herself at the end (Remark: Persons with physical handicap at the beginning) of a waiting list, while situating themselves near the elevator area. They don´t enter this area which is represented in the picture with the dark blue color.

When the cabin arrives the floor the first step is to exit all passengers, which will be inserted into this empty, for this purpose reserved, elevator area with a specified delay between each person. The second step is to remove the waiting passengers, as many as possible, depending on the capacity of the cabin, with the same delay time, from the discrete grid and mark them as “now in the elevator”. With this method the movements in front of the elevator are reduced to a minimum and so is the possibility of a deadlock.

8 Physical Handicapped People

To consider the special needs of people with physical handicaps this simulator also contains the possibility to generate people which rely on a wheel chair. It exists an adjustment for the probability that a random generated person is physical handicapped. The major different attributes of these persons are:

- The need of a special speed distribution.
- An increased required space. Instead of 4 x 4 cells wheel chairs need a space of 6 x 6 cells.
- As opposed to persons with no physical handicap wheel chair users do obviously not have the possibility to reduce their used space.
- The priority in the elevator waiting queues. This is not necessary for the simulation but in the real world commonly used.
- The inability of using stairs or overcoming other obstacles. To simplify the model it will not be considered how small these obstacles are. That means that already one stair, which can be handled in the real world by many of wheel chair users, is in the simulation an insurmountable obstacle.

The last point needs a little more effort in the implementation into the software. To solve this problem an independent routing algorithm is implemented, which works basically in the same way as the original with the one exception, that stairs are handled in same way as walls.

This additional algorithm implicates also that for each cellular automaton a second static floor field is used, doubling the amount of memory.

9 Conclusion

Combining different model types to implement hybrid simulation systems can solve some classical problems, but some other problems may occur. Some advantages were mentioned. CAs were introduced in the simulation system to represent a model computing vacation times. A standard software for implementing parts like queuing servers would be a huge amount of work.

Some disadvantages like implementing interfaces for connecting the different “sub models” had to be accepted. The work on this project is in progress, the quantitative results for the whole system with about 9.000 rooms and - in the future - maybe about 30.000 students has to be validated –for a <morespace> overview see [8].

References

[1] A. Kirchner, H. Klüfel, K. Nishinari, A. Schadschneider, M. Schreckenber, *Discretization effects and the influence of walking speed in cellular automata models for pedestrian dynamics.*

[2] F. Weifeng, Y. Lizhng, F. Weicheng. *Simulation of bi-direction pedestrian movement using a cellular automata model.* Physica A 2003.

[3] M. Bruckner. *Modellierung der Fußgängerdynamik im universitären Betriebmit Hilfe von Zellulären Automaten in der Programmiersprache JAVA. Vienna University of Technology 2009*

[4] P. Molná., *Modellierung und Simulation der Dynamik von Fußgängerströmen. Shaker Verlag, Aachen 1996.*

[5] T. Fujiyama, N. Tyler. *An Explicit Study on Wallking Speeds of Pedestrians on Stairs.* Centre for Transport Studies. University College London.

[6] T. Kretz, A. Grünebohm, M. Kaufman, F. Mazur, M. Schreckenberg, *Experimental study of pedestrian counterflow in a corridor.* Journal of Statistical Mechanics: Theory and Experiment. 2006

[7] V. J. Blue, J. L. Adler, *Cellular automata microsimulation for modeling bi-directional pedestrian walkways.* Transporation Research Part B 2001

[8] S. Tauböck, F. Breitenecker, D. Wiegand, N. Popper. *The <morespace> Project: Modelling and Simulation of Room Management and Schedule Planning at University by Combining DEVS and Agent-based Approaches*. SNE Simulation Notes Europe 21(1), 11-20, ARGESIM Vienna 2011. doi: 10.11128/sne.tn.21.10039

Submitted: June 2012
Accepted: August 1, 2012