A Fuzzy Logic Inspired Cellular Automata Based Model for Simulating Crowd Evacuation Processes

Prodromos Gavriilidis[1], Ioannis Gerakakis[1], Ioakeim G. Georgoudas[1(✉)], Giuseppe A. Trunfio[2(✉)], and Georgios Ch. Sirakoulis[1(✉)]

[1] Department of Electrical and Computer Engineering, School of Engineering, Democritus University of Thrace, University Campus, Kimmeria, 67100 Xanthi, Greece
{pgavri,igerak,igeorg,gsirak}@ee.duth.gr

[2] DADU, University of Sassari, Piazza Duomo, 6, 07041 Alghero, SS, Italy
trunfio@uniss.it

Abstract. This work investigates the incorporation of fuzzy logic principles in a cellular automata (CA) based model that simulates crowd dynamics and crowd evacuation processes with the usage of a Mamdani type fuzzy inference system. Major attributes of the model that affect its response, such as orientation, have been deployed as linguistic variables whose values are words rather than numbers. Thus, a basic concept of fuzzy logic is realised. Moreover, fuzzy *if-then* rules constitute the mechanism that deals with fuzzy consequents and fuzzy antecedents. The proposed model also maintains its CA prominent features, thus exploiting parallel activation of transition rules for all cells and efficient use of computational resources. In case of evacuation, the selection of the appropriate path is primarily addressed using the criterion of distance. To further speed up the execution of the Fuzzy CA model the concept of the inherent parallelization was considered through the GPU programming principles. Finally, validation process of the proposed model incorporates comparison of the corresponding fundamental diagram with those from the literature for a building that has been selected for hosting the museum ‘CONSTANTIN XENAKIS’, in Serres, Greece.

Keywords: Crowd modelling · Cellular automata · Fuzzy logic
Evacuation · Flow-density diagram · Speed-density diagram

1 Introduction

Crowd evacuation is a research area that has been thoroughly investigated by the scientific community. Many researchers from different disciplines have applied various methodologies to approach realistically issues related to the movement of people when massively abandoning an area. The major challenge for all deployed mechanisms is to improve the safety standards of evacuation processes. In such studies, a large number of people are involved and the interactions between them can be hardly described by conventional equations because of the psychological factors that influence their behavior. Moreover, the layout of the facilities has significant impact on the evolution of the evacuation. Thus, evacuation dynamics incorporates nonlinear characteristics and is very complex [1].

It is of high interest that modeling approaches of crowd movement can achieve an acceptable level of realism, and can be efficiently validated with empirical data in order to provide robust results. According to a recent review about evacuation models [2], empirical research often focuses on the relationship between walking speed and density as well as the relationship between flow, people and density. These relationships are called the 'fundamental diagram', because of their importance in determining the optimal dimensions of pedestrian facilities [3]. Often, the main modeling issue is the ability to successfully handle congestion, in order to prevent unpleasant circumstances, such as stampede, trampling and casualties [4–6]. Concurrently, the applications of fuzzy logic have increased significantly. According to [7], fuzzy logic is a theory that tries to broaden the limits of a set of acceptable values by defining not crisp boundaries in which membership is a matter of degree. Furthermore, by introducing the notion of a *linguistic variable*, whose values are words rather than numbers, fuzzy logic could be considered as a methodology for computing with words rather than numbers that try to lower the cost of solution at the expense of decreased but acceptable precision.

Literature review shows that the combination of cellular automata (CA) models with fuzzy logic is quite effective. For instance, Bisgambiglia et al. [8] presented a method that incorporates fuzzy inference systems in activity-based CA simulations. Betel and Flocchini [9] investigated the relationship between fuzzy and Boolean CA, whereas Cattaneo et al. [10] and Adamatzky [11] developed CA models where the local transition rule is described by a fuzzy function. Moreover, Chaia et al. presented a safety evaluation of driver cognitive failures and driving errors on right-turn filtering movement based on fuzzy CA [12]. Finally, Al-Ahmadi et al. developed a fuzzy CA model of urban dynamics [13].

The approach proposed in this paper combines fuzzy logic and CA to build a reliable model that simulates crowd evacuation. The motivation for incorporating fuzzy logic stems from the fact that it enables computing with words, which 'are inherently less precise than numbers, but their use is closer to human intuition' [7]. For instance, it is the orientation of the direction (e.g. 'north', 'south-east', etc.) that plays a dominant role in the movement of an individual rather than an exact angle measured in degrees. The same can be adopted for the state of a cell, which can be adequately characterised by words, such as 'free', 'occupied' or 'obstacle' for the needs of a model that mainly targets to quick response and tries to lower the cost of solution. Specifically, a Mamdani type fuzzy inference system (FIS) has been developed using the MATLAB Fuzzy Logic Toolbox^{TM}. Major features of the model, such as state of a CA cell and direction have been represented as linguistic variables. Additionally, fuzzy *if-then* rules are properly constructed based on the descriptions of the input and output variables. The structure of the model follows the CA principles, that is, it focuses on optimized utilisation of computational resources and decreased complexity simultaneously maintaining accurate modeling of the evacuation process with microscopic and macroscopic characteristics. Furthermore, and to further speed up the execution of the introduced Fuzzy CA (FCA) model, the concept of the inherent parallelization was discussed in the view of the GPU programming principles in order to achieve a fast execution of the FCA. The simulation process is characterized by fundamental features of crowd evacuation, such as transition from uncoordinated to coordinated movement due to common purpose, arching in front of exits, herding behaviour [1]. Finally, the evaluation of the model process is addressed by the comparison of the flow-density and speed-density response of the model with corresponding representations from literature for a building that has been decided to host the museum 'CONSTANTIN XENAKIS', in Serres, Greece.

In the following, the theoretical principles of the proposed evacuation model are presented (Sect. 2). In Sect. 3, the GPU implementation of the proposed model is presented, whereas in Sect. 4, simulation scenarios and corresponding results are presented and discussed. Finally, conclusions are drawn in Sect. 5.

2 Model Description

The model is CA-based: the space is a two-dimensional grid of identical cells, which is homogeneous and isotropic. Each cell may be either free or occupied by an individual or an obstacle. The state of each cell, which is represented by $C^t_{i,j}$, with i and j being the coordinates of the cell and t the evolution time, is described as:

$$C^t_{i,j} = \{o,\, id\} \tag{1}$$

with o representing whether the cell is free or occupied and id representing the class of the cell, i.e. whether it is a person or an obstacle. The neighbourhood consists of the eight closest neighbour cells (Moore neighbourhood), thus allowing each person to move towards eight directions. To update the state of a given cell, it is required the knowledge of the target exit $C^t_{i_{exit},j_{exit}}$ as well as of the status of all neighbouring cells. Therefore, the evolution rule is defined as:

$$C^{t+1}_{i,j} = R\left(C^t_{i,j}, C^t_{i\pm1,j,}, C^t_{i,j\pm1}, C^t_{i-1,j+1}, C^t_{i+1,j-1}, C^t_{i+1,j+1}, C^t_{i_{exit},j_{exit}}\right) \tag{2}$$

The transition function R of the FCA that defines the position of each person during each time step is realized by a FIS, particularly a Mamdani type, that is built using the MATLAB Fuzzy Logic Toolbox^{TM} tool [14,15]. Both the state of each cell of the Moore neighbourhood and the orientation of the person relative to the closest exit are considered as inputs. The orientation is defined as:

$$\text{difference}_x = x_{exit} - x_{person}, \quad \text{difference}_y = y_{exit} - y_{person} \tag{3}$$

that is, as the position of the an individual relative to its closest exit. It should be noted that notation x represents the vertical axis, whereas y the horizontal one. The difference between the ordinate value of an exit (that corresponds to the outcome of a selection process described below) and that of the person designates the north-south orientation; when the difference is negative then the exit is located northern in regard to the person (denoted as ‘exit-x is North’ at the corresponding *if-then* rule), whereas when the difference is positive then the exit is located southern (‘exit-x is South’). Accordingly, the difference between the abscissa values of the exit and that of the person designates the east-west orientation; when it is negative then the exit lays western in regard to the person (‘exit-y is West’), whereas when it is positive then the exit lays eastern (‘exit-y is East’). The combination of the upper two parts defines the overall orientation. The implementation of the rule can be represented by a triangular membership function, as shown in Fig. 1. This is just the collection of three points that form a triangle. Thus, when the difference is negative, the value -5 is assigned to the membership function (‘North/West’ case), when the difference is positive the value 5 is assigned (‘South/East’ case), whereas value 0 is assigned when the individual shares the same ordinate (abscissa) with the exit (‘Non’ case).

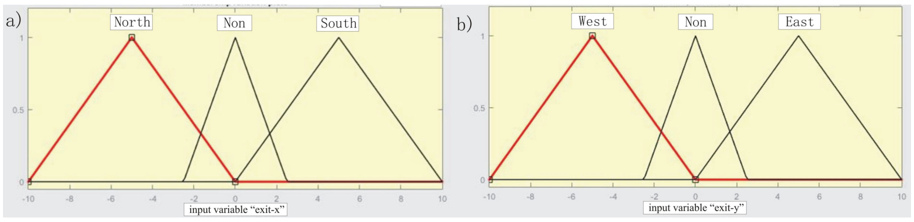

Fig. 1. The graphical representations of the membership functions that define (a) north-south orientation, and (b) east-west orientation regarding the direction of motion of an individual towards the chosen exit. ‘Non’ function represents the case of zero difference, i.e. when the individual shares the same ordinate (abscissa) with the exit.

In order all possible cell states, i.e. free, occupied and obstacle, to be represented for each part of all eight possible directions (N, NW, W, SW, S, SE, E, NE), a triangular membership function is utilized (Fig. 2). Value 0 is assigned when the corresponding state of the cell is free, whereas value 2 represents occupation by an individual and value 1 denotes the existence of an obstacle. Thus, the triangular function is adequate for representing the corresponding rule.

The list of rules that define the behaviour of the system is defined by a set of forty-eight (48) *if-then* rule statements that cover all possible cases of movement. These rules are used to define the conditional statements in terms of fuzzy logic. A sample of the whole set of rules is provided below:

1. If exit - x is North and exit - y is West and NW - State is Unoccupied then Output is NW
2. If exit - x is North and exit - y is West and NW - State is Occupied and W - State is Unoccupied then Output is W
3. If exit - x is North and exit - y is West and NW - State is Occupied and W - State is Occupied and N - State is Unoccupied then Output is N.

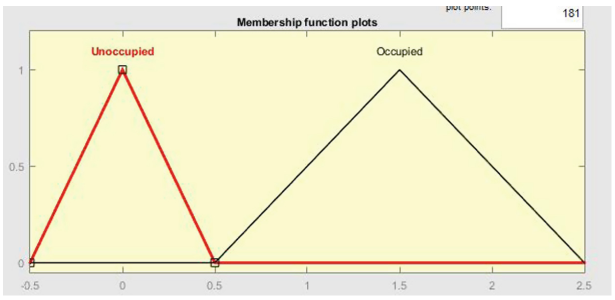

Fig. 2. The membership function that represents the state of each of the cells of the Moore neighbourhood.

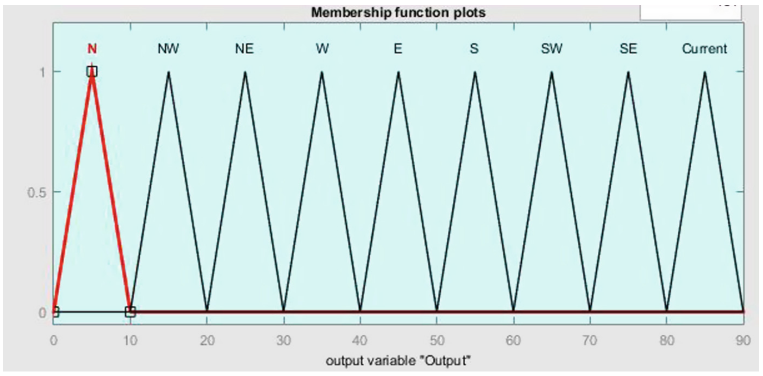

Fig. 3. The graphical representation of the response of the fuzzy inference system.

The corresponding response of the model is depicted in Fig. 3. The selection of the most appropriate exit for an individual to move during the upcoming time step is realized by utilizing the criterion of distance. In particular, the closest exit is computed according to the minimum Euclidean distance of the cell at (i, j) from all available exits:

$$R = \sqrt{(i - i_{exit})^2 - (j - j_{exit})^2} \tag{4}$$

Initially, the maximum distance from each exit is calculated according to the layout of the site. Maximum distance is algorithmically represented by a vector, named *max_dist*, the size of which is equal to the number of exits within the site. Then the distance of each individual is normalised by dividing the elements of *max_dist* by its greatest element. In case that an element of the normalized *max_dist* is greater than (or equal to) 2/3 and smaller (or equal to) than unity (1), then the corresponding distance is characterised 'long' and it is assigned value 2 in the corresponding element of a look-up table. Provided that the ratio is greater than (or equal to) 1/3 and smaller than 2/3 then the distance is characterised as moderate and is assigned value 1, whereas when the ratio is smaller than 1/3 then the distance is characterised as short and the corresponding value in the look-up table is equal to 0. The distance criterion is initially applied when the members of the crowd start to move aiming at the initialization of the look-up table and the characterization of each of the exits for every single person. Afterwards, the function *eval_pos* is activated, whose argument is a vector that carries the characterisations of the distances to all the exits (short, medium, long). Function *eval_pos* calculates a vector that includes all exits that share the similar characterisation of *short*, regarding to the minimum distance. In case that the output vector contains a single element, then that particular exit is indicated as the closest one, otherwise one of the exits that share the same minimum distance from the individual is chosen randomly.

Furthermore, the model has incorporated the auto-defined obstacle avoidance method, which is an automated process that enables people to overcome obstacles based on the effect of a virtual field generated near obstacles. The method is thoroughly described in [16]. In case that more than one person tries to reach the same cell then one of them is selected randomly. Finally, individuals are considered similar in terms of decision-making process, but they can be attributed different characteristics regarding speed. The model enables the assignment of dissimilar moving steps to groups of pedestrians, thus enhancing its reliability.

3 GPU Implementation

To enable fast simulations, we devised an implementation of the proposed model exploiting Graphics Processing Units (GPU) as parallel computing devices. In particular, we adopted a hybrid CPU-GPU approach based on the NVIDIA GPGPU platform with the well-known CUDA language. In the latter, the parallel computation is obtained by activating at each CA step a GPU thread associated to every CA cell occupied by a person (i.e., to every *active* cell). Such threads correspond to device (i.e. GPU) functions in C language, which are called *kernels*. When a kernel is issued by the CPU, a number of threads (i.e., one for each active cell, in our case) execute its code in parallel on different data. In the above CA simulation, the kernels operate on two distinct memory regions, representing the current and next states for the CA cells, respectively, where the state refer to the cell's occupancy by the simulated moving crowd, presence of exits and obstacles. The simulation begins by transferring from the host memory (i.e. that directly accessed by the CPU) to the GPU global memory the initial CA states, stored as arrays in order to favour faster coalesced accesses. Besides the state of the cells, we used some additional auxiliary arrays in the GPU global memory for storing the neighbourhood structure and the model parameters. During each CA step, the kernel execute the CA transition function described in Sect. 2, operating on the basis of the values from the current CA and exploiting some auxiliary functions (e.g. implementing the fuzzy membership functions); after, it writes the new state value into the appropriate elements of the next CA. At the end of each CA step, a *device-to-device* memory copy operation is used to re-initialise the current CA values with the next values. When the CA state is required by the CPU during the simulation (e.g. for depicting a graphical output), a *device-to-host* memory copy is carried out.

To implement the CA transition function, we developed a single kernel, which is executed on a dynamic grid of threads to improve the scalability with respect to both the number of involved pedestrians and the size of the CA. This was accomplished by keeping track of moving people in GPU global memory by means of an array B of integers. Each element of B encodes a pair $\langle id, c\rangle$, where id identifies a specific person contained in the cell c. Before the beginning of the simulation, B is initialized by the CPU through a *host-to-device* memory copy. Subsequently, *movePersonKernel* is executed on a dynamic one-dimensional grid of threads corresponding to the elements of the array B, which is updated by the kernel itself. More in details, the kernel operates on two global memory areas: B, which is the input container from where *movePersonKernel* takes the cells to process the current iteration; and B^*, which includes the active cells (i.e., those containing people) for the next iteration. At the end of each CA step, the pointers to B and B^* are simply exchanged. The first step of *movePersonKernel* consists of retrieving the pair $\langle id, c\rangle$ corresponding to the current thread. Then, a movement of the person id is considered only if a specific counter, initialized according to the person speed, is zero. Otherwise, the counter is simply decremented and B^* is updated with the insertion of $\langle id, c\rangle$ (i.e., the person id remains in the same cell). If the speed counter is zero and, according to the fuzzy rules outlined above, the person id can move to a neighbouring cell c^* different from an exit, then B^* is updated with the insertion of the pair $\langle id, c^*\rangle$. At the end of each iteration (i.e. when the kernel returns), the size of B^* is retrieved from global memory. If B^* is empty, the simulation ends because all people are outside the simulation area (i.e., they reached an exit). It is worth noting that to minimize expensive synchronizations, the new container B^* is cooperatively built by the involved threads by managing a hierarchy of smaller containers stored in shared memory, as suggested in [17,18]. In particular, thread-level arrays B_t are used, with the maximum size of a neighbourhood, in which the insertion can be executed by the owner thread without the need of synchronizations. Then, in each block, the thread-level arrays are copied into a single block-level array B_b by using a parallel prefix-scan approach. The latter can be implemented in a very efficient way in CUDA, only requiring two thread synchronizations and no atomic operations. Instead, a single *atomicAdd* is used by the first thread of each block for obtaining the offset required to copy its block-level array B_b into the final container B. A particular case that *movePersonKernel* takes into account is the conflict that may happen when more than one individuals are trying to reach the same cell. Also, this case is handled though a single *atomicMax* operation on the array containing the next states of the CA.

4 Simulation Results

In this study, the venue that is selected for all simulation purposes is the building that has been decided to host the museum 'CONSTANTIN XENAKIS'. It is located in Serres, a city of the administrative region of Central Macedonia, in Northern Greece. The building has not been redecorated yet, thus any useful conclusion regarding its standards of safety could be taken into consideration. The floor plan of the building can be found on the following site: http://serreonpoliteia.com/?page_id=10.

The following considerations were applied concerning the aforementioned place used for simulation purposed. As outputs of the under-study building, two exits as well as its main entrance are considered. It is also regarded that the building windows cannot be used for evacuation purposes. Both the length and the width of each cell within the CA grid is assumed to be equal with 0.3 m. Within the building, though, there are walls which are less than 0.3 m thick. Therefore, every part of the construction that is less than 0.3 m thick is supposed 0.3 m thick. Moreover, for constructions that are more than 0.3 m thick their corresponding dimension is calculated by dividing this dimension with 0.3 and applying rounding rules. Thus, for instance, an exit that is 4.75 m wide, it will be represented by $(4.75\,\mathrm{m})/(0.3\,\mathrm{m}) = 15.8333 \approx 16$ cells. Consequently, the maximum width of the building is calculated equal to 76 cells and its maximum length is equal to 148 cells. According to the simulation scenario 400 individuals are randomly assigned to positions within the museum (Fig. 4).

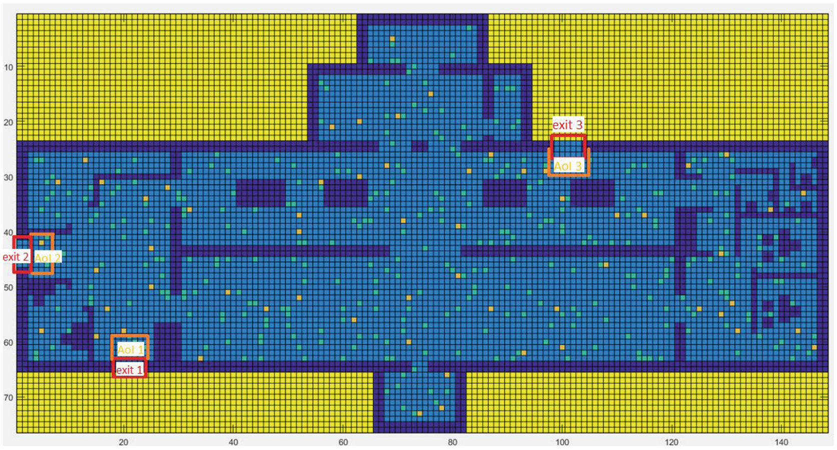

Fig. 4. Simulation scenario: random initialisation, main exits and corresponding areas of interest (AoI).

The model is evaluated by comparing the fundamental characteristics and graphical representations generated by the simulated three scenarios with the corresponding diagrams from literature. Particularly, the overall density of the crowd is measured by counting all individuals in the area of interest and then dividing by the area of this region. The total flow is calculated by dividing the total number of individuals who go through the exits per simulation step with the length of this intersection. The flow of people results from experimental relations of speed of people with the density of people. Taking as an assumption that people move smoothly, the flow per meter of width is given by $Q(\rho) = \rho V(\rho)$, where Q represents the flow per meter of width and ρ is the density of individuals. Each person covers an area equal to that of single cell, i.e. $0.4 \times 0.4 = 0.16\,\text{m}^2$. For each step, the number of people within the area of interest (AoI) is counted and the value of the density is calculated from the relationship:

$$\rho = \frac{\text{num. of people in RoI}}{\text{area of AoI}} \tag{5}$$

There are three AoI, each corresponding to an exit (Fig. 4). The total AoI area consists of $5 \times 8 = 40$ cells, i.e. $6.4\,\text{m}^2$. Regarding the simulation scenario and for the AoI of the first exit (AoI 1) the corresponding curve of flow vs. density is depicted in Fig. 5(a). In Fig. 5(b), the corresponding flow-density curves from literature are depicted [19]. As can be seen, the corresponding curves present common behavioural attributes, both qualitatively and quantitatively, thus enhancing the validity of the proposed approach.

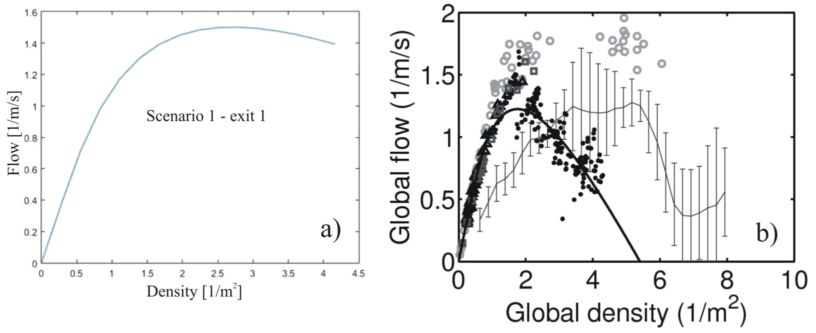

Fig. 5. (a) Simulation scenario. AoI of exit 1. Flow vs. density; (b) Flow-density results from literature.

5 Conclusions

Results prove that fuzzy type logic can find application in real-world evacuation conditions and in particular when describing crowd dynamics. More specifically, they respond to expected behaviours. According to the directional selection criterion, persons have full sense of orientation towards the desired output and perform movements as in real-world conditions. Furthermore, the implementation of the fuzzy CA rules for intuitive exit selection has been achieved, taking into account the distance of a person from the exit, so that they correspond to real conditions. The model need further to be validated with real data. Thus, it could be better calibrated, and it could be parameterised more efficiently.

References

1. Helbing, D., Johansson, A.: Pedestrian, crowd and evacuation dynamics. In: Meyers, R.A. (ed.) Encyclopedia of Complexity and System Science, vol. 16, pp. 6476–6495. Springer, New York (2010). https://doi.org/10.1007/978-0-387-30440-3_382
2. Vermuyten, H., Beliën, J., De Boeck, L., Reniers, G., Wauters, T.: A review of optimisation models for pedestrian evacuation and design problems. Saf. Sci. **87**, 167–178 (2016)
3. Schadschneider, A., Seyfried, A.: Empirical results for pedestrian dynamics and their implications for cellular automata models. In: Pedestrian Behavior - Models, Data Collection and Applications, pp. 27–44 (2009)
4. Georgoudas, I.G., Sirakoulis, G.C., Andreadis, I.T.: An anticipative crowd management system preventing clogging in exits during pedestrian evacuation processes. IEEE Syst. J. **5**(1), 129–141 (2010)
5. Vermuyten, H., Lemmens, S., Marques, I., Beliën, J.: Developing compact course timetables with optimized student flows. Eur. J. Oper. Res. **251**(2), 651–661 (2016)
6. Zarboutis, N., Marmaras, N.: Design of formative evacuation plans using agent-based simulation. Saf. Sci. **45**(9), 920–940 (2007)
7. https://www.mathworks.com/help/fuzzy/what-is-fuzzy-logic.html
8. Bisgambiglia, P.A., Innocenti, E., Gonsolin, P.R.: A new way to use fuzzy inference systems in activity-based cellular modeling simulations. In: IEEE International Conference on Fuzzy Systems (2017)
9. Betel, H., Flocchini, P.: On the relationship between fuzzy and Boolean cellular automata. Theor. Comput. Sci. **412**(8–10), 703–713 (2011)
10. Cattaneo, G., Flocchini, P., Mauri, G., Vogliotti, C.Q., Santoro, N.: Cellular automata in fuzzy backgrounds. Phys. D: Nonlinear Phenom. **105**(1–3), 105–120 (1997)
11. Adamatzky, A.I.: Hierarchy of fuzzy cellular automata. Fuzzy Sets Syst. **62**(2), 167–174 (1994)
12. Chaia, C., Wong, Y.D., Wang, X.: Safety evaluation of driver cognitive failures and driving errors on right-turn filtering movement at signalized road intersections based on Fuzzy Cellular Automata (FCA) model. Accid. Anal. Prev. **104**, 156–164 (2017)
13. Al-Ahmadi, K., See, L., Heppenstall, A., Hogg, J.: Calibration of a fuzzy cellular automata model of urban dynamics in Saudi Arabia. Ecol. Complex. **6**(2), 80–101 (2009)
14. Zadeh, L.A.: Fuzzy logic. Computer **1**(4), 83–93 (1988)
15. Mamdani, E.H.: Applications of fuzzy logic to approximate reasoning using linguistic synthesis. IEEE Trans. Comput. **26**(12), 1182–1191 (1977)
16. Georgoudas, I.G., Koltsidas, G., Sirakoulis, G.C., Andreadis, I.T.: A cellular automaton model for crowd evacuation and its auto-defined obstacle avoidance attribute. In: Bandini, S., Manzoni, S., Umeo, H., Vizzari, G. (eds.) ACRI 2010. LNCS, vol. 6350, pp. 455–464. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15979-4_48
17. Trunfio, G.A., Sirakoulis, G.C.: Computing multiple accumulated cost surfaces with graphics processing units. In: 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, pp. 694–701. IEEE (2016)
18. Teodoro, G., Pan, T., Kurc, T.M., Kong, J., Cooper, L.A.D., Saltz, J.H.: Efficient irregular wavefront propagation algorithms on hybrid CPU-GPU machines. Parallel Comput. **39**(4–5), 189–211 (2013)
19. Johansson, A., Helbing, D., A-Abideen, H.Z., Al-Bosta, S.: From crowd dynamics to crowd safety: a video-based analysis. Adv. Complex Syst. **11**(4), 497–527 (2008)