FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Assignment of master's thesis

Title: Evacuation model with leading and following agents focused on evacuation of (pre)schools
Student: Bc. Matej Šutý
Supervisor: Ing. Pavel Hrabák, Ph.D.
Study program: Informatics
Branch / specialization: Knowledge Engineering
Department: Department of Applied Mathematics
Validity: until the end of summer semester 2023/2024

Instructions

The thesis should combine the idea of using a hierarchical system of multi-agent coordination for evacuation simulations (developed in Janovská, 2021) with expert knowledge on pre-school children's behaviour during evacuation (captured by Najmanová, 2020). The work should result in an evacuation model/software prototype, where the dynamics of the following agents (children) are consistent with the expert knowledge, and the planning actions of leading agents (teachers) are inspired by real-world instructions and limitations.

1. Survey evacuation models enabling assisted evacuation with a focus on evacuation of (pre)-school activities. Focus on cellular models.
2. Survey studies dealing with applications of multiagent planning algorithms in evacuation simulations.
3. In cooperation with the fire engineering expert (H. Najmanová) identify basic principles of children's behaviour and teachers' activities during evacuation. Suggest a transformation of those principles to rules of agent interaction in the cellular model.
4. Design and implement the cellular model of evacuation with the above-mentioned features enabling the application of planning algorithms for the actions of leading agents.
5. Perform several simulation experiments and compare various leading agent strategies with respect to total evacuation time.

Electronically approved by Ing. Magda Friedjungová, Ph.D. on 27 December 2022 in Prague.

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Master's thesis

Evacuation model with leading and following agents focused on evacuation of (pre)schools

Bc. Matej Šutý

Department of Applied Mathematics
Supervisor: Ing. Pavel Hrabák, Ph.D.

April 19, 2023

Acknowledgements

THANKS (remove entirely in case you do not with to thank anyone)

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60 (1) of the Act.

In Prague on April 19, 2023 …………………

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis

Šutý, Matej). *Evacuation model with leading and following agents focused on evacuation of (pre)schools.* Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2023.

Abstrakt

V několika větách shrňte obsah a přínos této práce v českém jazyce.

Klíčová slova Replace with comma-separated list of keywords in Czech.

Abstract

Summarize the contents and contribution of your work in a few sentences in English language.

Keywords Replace with comma-separated list of keywords in English.

Contents

List of Figures

Introduction

Cooperation

Chapter 1

State-of-the-art

Chapter 2

Analysis and design

2.1 Model

This thesis proposes a novel hierarchical system for the coordination of multi-agent evacuation simulations in a preschool environment. The hierarchical structure is a crucial aspect of the model, as it facilitates the efficient management of the various agents within the simulation highlighted in the work of Janovská [2]. The proposed system consists of two distinct agent types: children and a leader. The children have the capability to form pairs and follow the orders of a leader, while the leader is responsible for navigating towards the goals and influencing children in its close proximity. Additionally, both agent types can have varying speeds of movement to reflect real-world behavior. A centralized planning approach is employed to control the formation of pairs, the location of the leader in the queue, and the assignment of goals among others.

The model presents a unique and innovative approach to the simulation of multi-agent evacuations in a preschool setting, which has significant implications for the field of emergency management and preparedness.

2.1.1 Floor field model

The baseline for the evacuation model with leading and following agents is a widely used extended cellular automata floor field model [3]. A cellular automaton is a mathematical or computational system composed of discrete elements that can be in a finite number of states - Moore neighbourhood for a single agent and extended Moore neighbourhood for a pair, further details in Section2.2.9. These elements are updated according to a set of rules. While deterministic in behavior, the variance in results due to stochastic selection can mimic the randomness observed in human behavior, making it a desired feature of the model. The floor field CA model, which is the inspiration for the model used in this thesis, utilizes several floor fields, including static, dynamic, and leader proximity, to model pedestrian movement in a two-dimensional lattice. Agents, representing pedestrians with individual parameters, move on the lattice and interact with the floor fields. The model update rules include updating the dynamic field, calculating the transition probability for a move to a neighboring cell, choosing a random target cell based on the probability, resolving conflicts when two or more agents attempt to enter the same cell. The evacuation model in this thesis represents people in evacuation as agents who move on a rectangular two-dimensional grid that acts as a room, corridor or other area.

2.1.2 Hierarchical structure

The hierarchical system follows the research of controlling swarm with leading and following agents [2]. Such a system suits the environment of pre-school, where children (following agents) are used to following a teacher, parent or other adult person with authority (leading agent). The leading agent follows the optimal path (set by static floor field) to the current goal and sets the static floor field for the goal of following agents. At the same time, the leading agent can change it's position in the crowd and influence the following agents by it's proximity. More details are in Section 2.3.3. Leading agent takes into consideration following agents that are left behind and adjusts its speed to allow them to catch up. Also, following agents are semi-autonomous which means that they can evacuate the room when given the opportunity of being close to the exit.

2.1.3 Planning

Through communication with fire engineering expert H. Najmanová, the fundamental principles of children's behavior and teachers' activities during evacuation have been translated into a set of simple rules and strategies for the leading agent. These rules, when implemented, have been shown to impact the course of evacuation in several ways. Firstly, the speed at which agents move is affected, which consequently results in different total evacuation time (TET). Secondly, the structure and coherence of the group of agents is influenced. Lastly, the microscopic behavior of agents, including the formation of pairs, is observed to change. This thesis provides further research for many ideas from the dissertation thesis of Najmanová [1].

The developed model provides a framework for assigning goals and different rules for leading and following agents. These rules are executed sequentially and are checked for whether they are achieved or not. Static assignment of goals and rules was sufficient for the needs of the research in this thesis. However, the framework can be easily adjusted to process goals and rules in real-time, for example using analyzer of the state of the simulation which generates goals. This enables the modeling of the complex interactions that occur during evacuation processes, and provides a basis for analyzing the effectiveness of different evacuation strategies.

2.2 Methods

2.2.1 Static floor field

Each cell in the grid in the static floor field *SFF* holds the value of shortest distance to the goal. The value is pre-computed using breadth-first search *BFS* algorithm which allows diagonal movement. Alternatively, the distance can be computed by other algorithms, e.g. novel approximate algorithm described in [4]. The (virtual) leading agent has full information about the topology of the map and he moves optimally towards the closest goal. The follower agents, solitary or in pairs, follow the leading agent and they do not attempt to find their own way to the exit [1]. The virtual leader agent updates *SFF* for the follower agents - either with the current goal location or the virtual leader agents position.

2.2.2 Occupancy floor field

Each cell in the grid can hold at most one physical agent (leading or following). The virtual leader agent, does not occupy a cell even though its position is the same as the cells position. The occupancy floor field *OFF* is a boolean matrix of the same dimensions as the room. The position of cells occupied by a physical agent hold *true* value, otherwise the value is *false*. The occupancy floor field updates in each step of the model.

2.2.3 Attraction and sensitivity parameters

The selection of the next cell by an agent is influenced by both the agent's own state - sensitivity parameters, timestep, and partner agent, as well as the state of the agent's surroundings, such as the SFF of the cell, distance to the leader, occupancy of the cell, and obstacles in the corner. The agent computes the attraction of each cell in its surroundings using a mixed strategy based on the method proposed by Šutý [5] shown in Equation 2.1. The attraction of each cell is then normalized across all cells to compute the probability of selecting a particular cell from the set. The probability of an agent moving from cell x to cell y, denoted by $P(y \leftarrow x \mid N)$, is calculated based on two members, P_O and P_S.

$$P(y \leftarrow x \mid N) = k_O P_O(y) + (1 - k_O) P_S(y) \tag{2.1}$$

Specifically, $P_O(y)$ takes into account the occupancy of cell y and returns a normalized value in the range $[0, 1]$. On the other hand, $P_S(y)$ considers the static potential of cell y and guides the agent towards the exit, along with the diagonal motion indicator $D(y)$. Both P_O and P_S are normalized across neighboring cells from N.

$$P_O(y) = \frac{\exp(-k_S S(y))(1 - O(y))(1 - k_D D(y))}{\sum_{z \in N} \exp(-k_S S(z))(1 - O(z)(1 - k_D D(z))} \tag{2.2}$$

$$P_S(y) = \frac{\exp(-k_S S(y))(1 - k_D D(y))}{\sum_{z \in N} \exp(-k_S S(z))(1 - k_D D(z))} \tag{2.3}$$

The equation for computing cell attraction serves as the baseline for solitary agents moving to Moore neighborhood. When paired partner agents are involved, each agent computes the attraction of both cells in each maneuver. For instance, agent a_1 in cell c_1, paired with partner agent a_2 in cell c_2, computes the attraction of maneuver $m_1 \in M$ in the following manner:

$A_1 = A(a_1 \to c_{1*})$ and $A_2 = A(a_2 \to c_{2*})$, where c_{1*} and c_{2*} represent the cells after the maneuver. The attraction of maneuver m_1 is then determined as $A(m_1) = \min(A_1, A_2)$, and the probability of selecting maneuver m_1 by agents a_1 and a_2 is given by $P(m_1|a_1, a_2) = \frac{A(m_1)}{\sum_{m_i \in M} A(m_i)}$.

2.2.4 Conflicts

The finite number of cells in discrete rectangular grid results in conflicts when two or more agents attempt to enter the same cell. The winner of this conflict, allowed to enter the cell, is picked randomly with uniform probability for each participant. This deviates from the conflict solution in research of aggresivity of Šutý[5], where winner of the conflict was the agent with highest aggresion or a conflict happened where no winner was selected. The described method was omitted in this research because it would bring more variability and less readability in the scope of this thesis' research. The framework is ready to include the method if needed. Intuitively the author would recommend to assign highest aggressivity value to the leading agent and lower aggressivity value or distribution of values to the following agents.

2.2.5 Orientation

The grid in this model is a rectangular lattice of square cells where the agents can move from one cell to another in 8 directions of Moore neighbourhood and in special cases of a maneuver of a pair to the extended Moore neighbourhood as defined in Section 2.2.9. The agents have four orientations: North represents the top of the grid and South the bottom, East is the right-hand side of the grid and West is the left-hand side. The orientations are perpendicular one to each other. Even though the movement is possible in 8 directions the agent has only 4 orientations to facilitate the structure of paired agents. Two agents in a pair have the same orientation and are located in adjacent non-diagonal cells. Movement to a non-diagonal cell changes orientation while movement to diagonal cell preserves the agents' orientation. However, there are a few exceptions for paired agents as described in detail in Section 2.2.9.

2.2.6 Directed agents

Agents can move to cells in Moore neighbourhood in 8 directions or stay in the same cell. Directed agents take into consideration the direction of the movement. The agents move in a discretized rectangular grid. The agent has 4 possible orientations - North, East, South, West - which are global and not relative to the agent itself. This means that agent which has orientation East is directed to the right-hand side of the grid.

Moore neighbourhood allows movement in 4 non-diagonal and 4 diagonal directions. In case of the non-diagonal directions, the agent orientation after the move is the same as the direction e.g. agent with orientation North moving to adjacent cell on the right will change orientation to East. In case of diagonal directions there are 2 types of movements: the first is movement to diagonal cells where the steering angle is +-45 degrees and the second is movement to diagonal cells where the steering angle is +-135 degrees. In the case of the first movement the agents keeps his orientation unchanged, e.g. agent facing West moves to diagonal upper left cell will have West orientation after the movement. In the case of the second movement the agent changes the orientation to the opposite orientation e.g. North to South and vice versa, East to West and vice versa.

2.2.7 Partner agents

Partner agents form a pair of two directed agents that are tightly bound so that they are in adjacent cells to each other. Two partner agents in the pair cannot be in cells diagonal one to each other. Their movement is synchronous, which means that if any of the agent is not able to move to the desired cell (conflict, bound agent did not move), the partner agent will abort its movement. The algorithm which checks whether both agents will successfully move is described below. In this model, the two agents in the pair are in a hierarchy of one agent being the leader while the second agent is not. The leader is responsible for calculating the probabilities of movement to cells for itself and it's partner according to maneuvers. The leader is the agent which has it's partner on the right. The partner agents are directed agents and both have the same orientation. Note that some maneuvers change the leadership in the pair.

2.2.8 Pair formation

The children in pre-school age are being taught to form a pair and hold hands when walking through corridor or crossing a road as these situations pose risks such as getting lost or encountering traffic. The model in this thesis simulates the coordinated evacuation with supervisor where a risk element is present.

The pupils are located in a classroom in a cluster where some pupils are close to each other and others are more isolated. The model does not consider any friendship preferences between children and assumes that pupils close to each other are more likely to form a pairs. It also assumes that pupils form pairs in group of even number of pupils so that there are no solitary children. If a pupil can't form pair immediately it will do so when other solitary pupil is nearby.

The Algorithm 1 below finds a way to form pairs of pupils which are not yet in pair.

Algorithm 1 Finding pairs

```
$G = (V, E)$
while $\exists v \in V : d(v) > 1$ do
    Let $v^* \in V : d(v^*) = \max_{u \in V} d(u)$.
    Let $w \in V : d(w) = \max_{(v^*, w) \in E} d(w)$.
    Remove $(v^*, w)$.
end while
for $(v, w) \in E$ do
    formPair(v,w)
end for
```

Locations of pupils (directed agents) not in pair are transformed to a graph. The vertices of the graph correspond to the pupils' locations, and an edge is formed between two vertices in adjacent cells. The vertices in the graph may have different degrees, and cycles may be present. The algorithm iteratively selects the vertex with the highest degree and removes the edge connecting it to the vertex with the highest degree until all vertices have at most one edge. The vertices connected by an edge represent a pair.

2.2.9 Maneuvers

Each agent in a pair has the potential to move to 8 cells in the Moore neighbourhood, allowing for a vast number of possible maneuvers. However, most of these maneuvers do not maintain the structure of the pair and are therefore prohibited. Specifically, only 18 viable maneuvers are allowed for each orientation of the paired agents.

Some maneuvers can alter the leadership of the pair or the orientation of the agents. Additionally, in some cases, the agents may have different movement speeds. During the synchronous atomic movement of a maneuver, both agents may move to diagonal cells, or one agent may move to a diagonal cell while the other remains in its current cell. Alternatively, one agent may even move outside its Moore neighbourhood in order to preserve the structure of the pair. In the Figure 2.2 can be seen the notable maneuvers.

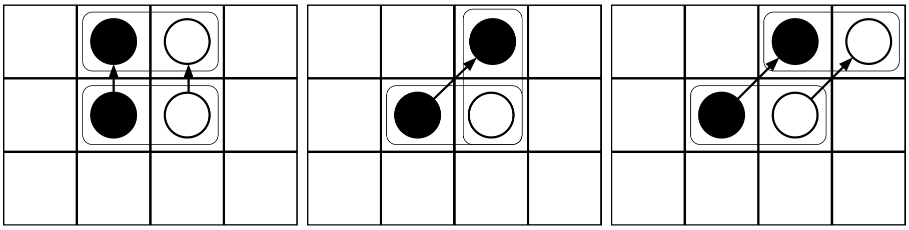


Figure 2.1: Simple maneuvers of paired agents.

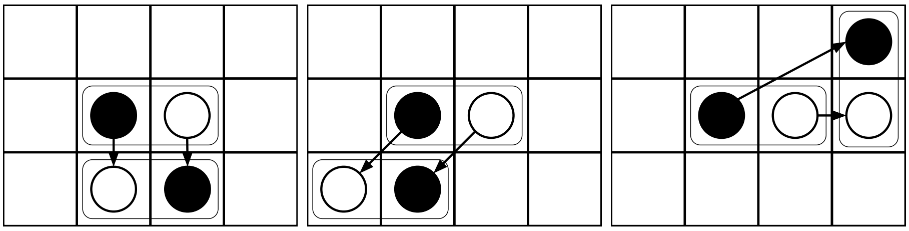


Figure 2.2: Complex maneuvers of paired agents.

2.2.10 Leader

The solitary agent responsible for navigation is the leader agent, which has complete information about the map topology and goals.

The leader agent is not directed and can move to cells in the Moore neighborhood based on the SFF of the current goal. The virtual leader is a leader agent, which does not occupy a cell but his position is is used as a goal to set SFF for follower agents. The SFF for the leader agent differs from the SFF of follower agents. The SFF for follower agents is calculated in every step based on the virtual leader's position. The virtual agent navigates the followers when the leader moves to the end of the crowd.

Agent's proximity to the leader (not virtual leader) proportionally increases the static potential value by increasing the sensitivity to static potential k_S:

$$d = \text{distance}(\text{leader}, \text{follower})$$
$$k_S = k_S * (1 + \frac{1}{d})$$

With a higher static potential value, the agent is less likely to deviate from the optimal trajectory set by the SFF of the virtual leader. This method is described in more detail in Section 2.3.3. The leader has simple rules for navigating towards the closest goal and checking if all follower agents reached the goal. One rule is the leader's ability to command the follower agents to continue to the goal while the leader moves to the most distant agent. Additionally, the (virtual) leader agent waits near the goal area and attracts the follower agents until they all reach it.

2.3 Strategies and rules

2.3.1 Leading agent strategy

In the hierarchical system a leading agents is responsible for navigating the following agents. The leading agent can achieve it in different ways as defined by the strategy. There are three leading agent strategies:

(A) Navigate towards the exit and evacuate.

Leading agent follows the optimal path to the exit. Leading agent positioned at the front of the crowd, the following agents are navigated by SFF calculated for them by virtual leader. Any agent, including the leading agent, that reaches the exit evacuates the room immediately.

(B) Navigate towards a location with specific position of the leading agent in the crowd.

Virtual agent follows the optimal path to the location. The leading agents position can be at the front or at the back of the crowd. The role of a virtual leader is essential in this strategy because the virtual agent is always at the front of the crowd and sets SFF for following agents. The virtual leader agent dynamically evaluates the distance to the most distant following agent and adjusts its own speed (and the speed of the leading agent if positioned at the front) so that the distant agent is not left too far behind. In the case of a leading agent positioned at the back of the crowd, leading agent adjusts its speed dynamically so that it is always at the back of the crowd. For detailed explanation visit Section 2.3.4. The close proximity of the leading agent at the back of the crowd increases the speed of the following agent at the back of the crowd so that they could join the crowd and make it more compact. In the end, the virtual agents tries to make the crowd compact by adjusting the speed at the front and the leading agent at the back influences the following agents at the back to make the crowd more compact.

(C) Leading agent standing guard at a location and following agents navigate towards next goal location.

Leading agent is standing at a strategic location, for example next to the exit or at an apex of a corner and the following agents are navigated by SFF set by virtual leader to their goal location. Following agents in the crowd pass the leading agent at a close proximity and adjust their speed within the crowd due to its influence. The leader joins the crowd at the back.

The three strategies can be used to organise children as described in the disertation thesis of Najmanová [1]:

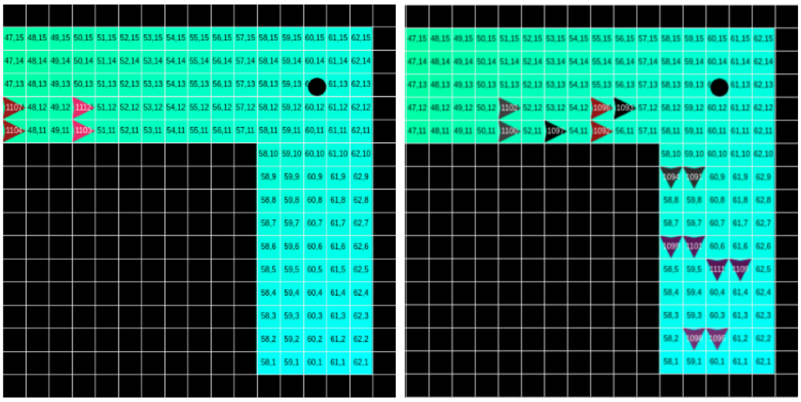


Figure 2.3: Leader strategy standing guard at a location and waiting (left), following agents passing the leader and navigating towards goal (right).

> *“Based on the leaving strategy employed, the following situation were observed during the experimental evacuation drills:*
>
> *1. Leaving as a group at once: All children were gathered in front of a closed exit from a classroom forming a standing queue, first the whole group was completed the door was opened and children started to leave the classroom.*
>
> *2. Leaving as a group gradually: All children were leaving together, however they were not gathered and checked in front of an exit from a classroom but when they arrived at the door as a moving queue, the door was already opened and supervised by a staff member and they could leave the room smoothly.*
>
> *3. Individual leaving: Even though supervised children could leave a classroom individually instructed to wait at a specified place inside or outside the building.”[1]*

2.3.2 Breaking pairs

The movement of paired agents is executed in maneuvers, which need to be atomic and consistent with the structure of a pair. When paired agents is close to the congested exit cell, in this thesis a single cell, the paired agents would struggle to execute movement maneuver due to the highly occupied cells around the exit. To address this issue, the pairs are broken in a close proximity to the exit. It is important to note, that breaking of pairs occurs only in close proximity to the exit cell and not in close proximity to the current goal. The topology of the map assumes a suitable path and wide enough corridors to fit paired agents thus there is no need for a method to solve dynamic breaking and creation of pairs because of a current state of evacuation. However, the

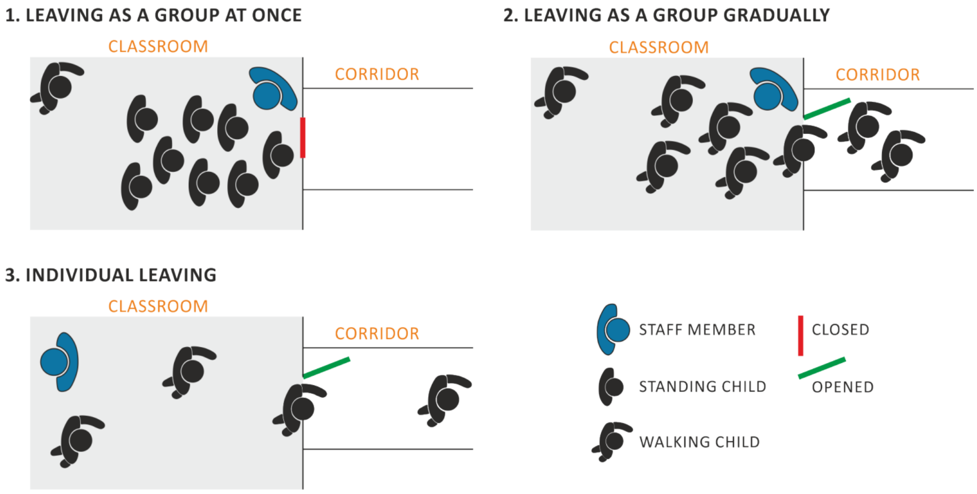


Figure 2.4: Image taken from [1].

framework can be adjusted to fit the specific needs as the dynamic creation or breaking of pairs is trivial and already present in the model.

2.3.3 Penalization and discipline

The behavior of children is known to be strongly influenced by the presence of authority figures, such as parents, teachers, or other responsible adults. In the proposed model, the leader assumes the role of authority and guides the children as they move through the environment. Specifically, children who are in close proximity to the leader exhibit higher levels of *discipline* and move in a more orderly manner, preserving the structure of the queue of pairs and following the optimal path as determined by the static force field (SFF). To promote this behavior, the static force value assigned to each agent is adjusted according to their proximity to the leader. In particular, as the distance d between an agent and the leader decreases, the static force value S assigned to that agent is multiplied by a factor of $(1 + \frac{1}{d})$. By moving back and forth within the queue, the leader can selectively increase the discipline of agents in close range and have greater control over the overall behavior of the group.

The paired partner agents form a queue and move one pair after other. Due to the nature of *maneuvers* described in Section 2.2.9 the paired agents sometimes rotate and change orientation because an non-optimal maneuver was selected stochastically. To limit this behavior a penalization for maneuvers with incorrect change of orientation is introduced. The penalization value can be set to fit needs of the simulation. In this model the value was set to 0.5 and every maneuver which results in unwanted change of direction is multiplied by the penalization to lower the chance of it being selected.

Maneuver resulting in rotation is wanted in case of agents turning in corner due to the change of leaders position. The correctness of the orientation is thus determined by the orientation of the most attractive maneuver. In example, paired agents facing North follow the leader which leads the queue situated on the North. The attractivity of maneuver moving both agents to the North is the highest and thus the correct orientation is North and maneuvers resulting in different orientation are penalized. In other example, paired agents facing North approach a right-turn in corridor where leader passed the turn and is located on the East. The maneuver which results in rotation of the agents to the East is the most attractive and thus the orientation East is selected as correct and maneuvers resulting in different orientation are penalized.

The nature of discrete rectangular grid lowers the resolution of movement around obstacles which needs to be addressed. A obstacle crossing penalization is introduced, which aims to restrict diagonal movement of agents across obstacles. The degree of penalization may be customized based on the requirements of the simulation, with a value of 0.5 being adopted in this instance. Non-obstacle crossing maneuvers do not incur any penalty. The degree of attraction is scaled by the penalization factor, with higher penalization values resulting in less obstacle-crossing maneuvers. Lower penalization of crossing penalization maneuvers in environments with narrow corridors or numerous obstacles can result in smoother movement of paired agent queues.

2.3.4 Adaptive time span and speed

This model allows agents to adjust their movement speed during an evacuation, with the leader represented as an adult with an average speed of $1.2m/s$, as per the findings in [6]. Follower agents, who are modeled as children, are assigned a speed of $0.9m/s$, based on experimental measurements of adult pairs traveling at a speed of $0.97m/s$. Each model step consists of two timesteps, representing the unit duration of movement. For instance, paired agents have a movement duration of three timestep units with a speed of $0.9m/s$, resulting in a speed of $0.3m/timestep$. The leader agent has a speed of $0.4m/timestep$. This update frequency of two timesteps per model step allows for synchronization of movement every three steps between the leader agent and paired agents. Specifically, each agent is equipped with an internal counter, denoted by τ, which is evaluated against the model's timestep clock, denoted by T. The timestep clock advances by two timesteps for each model step. Whenever τ is less than or equal to T, the agent initiates cell selection and increments τ by the duration of the maneuver.

Paired agents can dynamically adapt their speed in specific conditions. Close proximity to the leader and empty cell in front of them increases their speed. The leader agent can move back and forth in the queue and locally increase the speed of paired agents to repair the queue structure or close

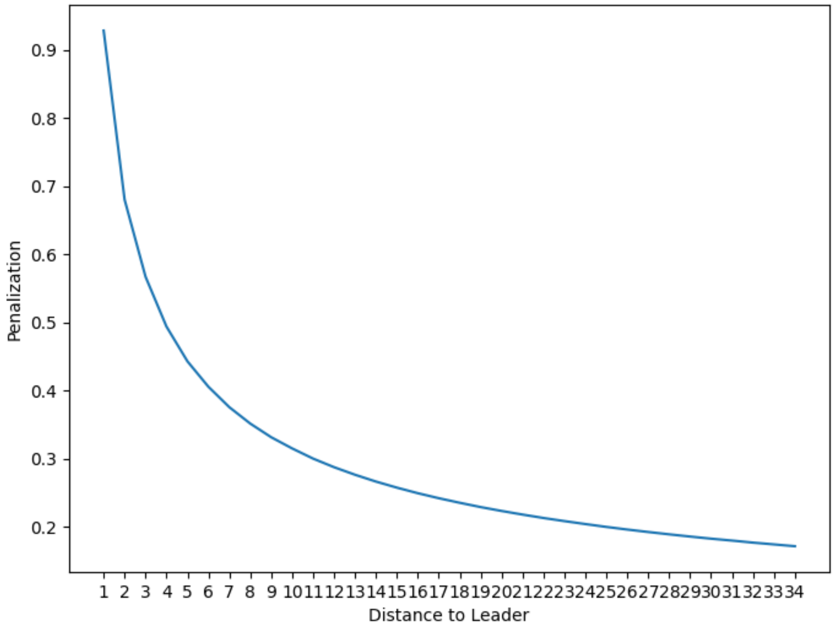


Figure 2.5: Penalization of maneuvers resulting in incorrect orientation are penalized proportionally to the distance to the leader.

the gaps between agent pairs. The change in speed reverts when previous conditions are not met.

The leading agent, modeled as an adult person, has higher speed than following agents. For n following agents in the simulation and leading agent at the front the model calculates distance from the virtual leader to the most distant agent d and lowers his speed when $\frac{n}{2} \leq d$. In case of leading agent at the back, the distance d is calculated from the leading agent to the agent at the back. When $d < 5$, the leading agent keeps its current speed, otherwise its speed is increased to catch up with the crowd. The leading agent being very close to the most distant agent from the virtual agent at the front of the crowd increases the speed of the slow agent to catch up with the rest of the crowd.

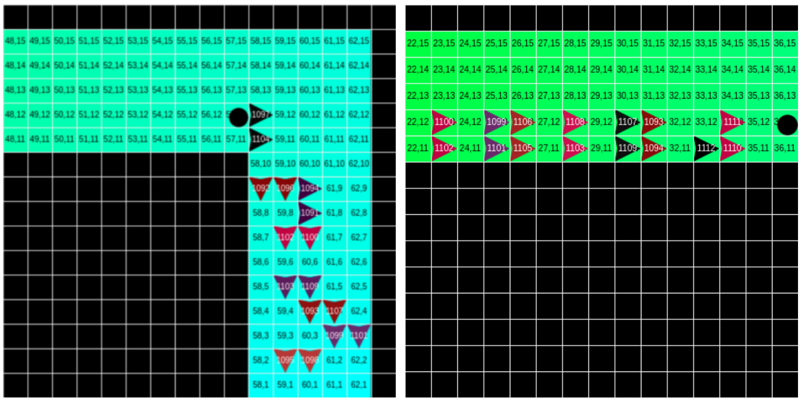


Figure 2.6: Leader position at the back (left) and at the front of the crowd (right).

Chapter 3

Implementation

The implementation of the model framework was carried out in Python, version 3.10, using several accompanying libraries. The complete source code is accessible at `https://gitlab.fit.cvut.cz/sutymate/mt-master-thesis`, along with comprehensive documentation and installation guidelines.

The maps, an integral component of the model, are also included in the repository.

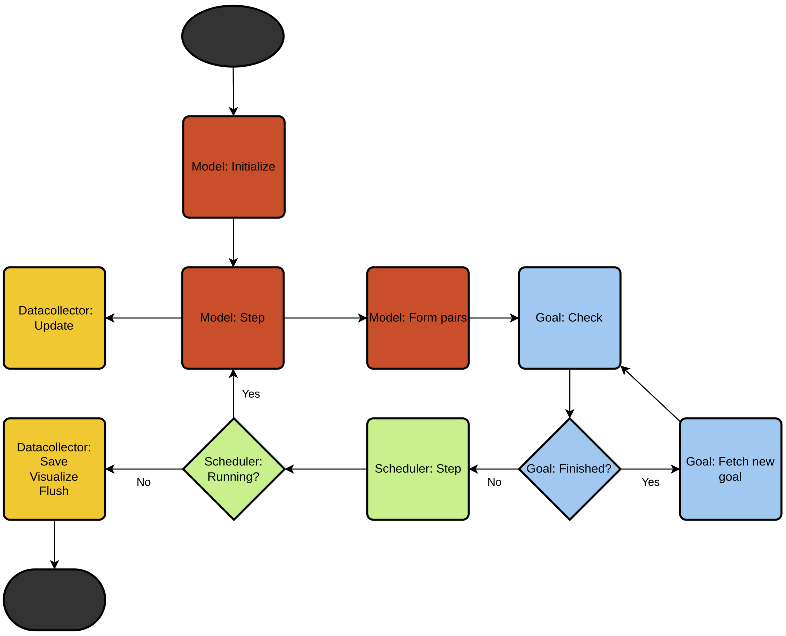


Figure 3.1: Schema of one simulation run.

In short, the architecture of the model is made out of several objects resting in the *RoomModel* class. *RoomModel* orchestrates initialization of the model, running it, collecting and saving data, creating graphs.

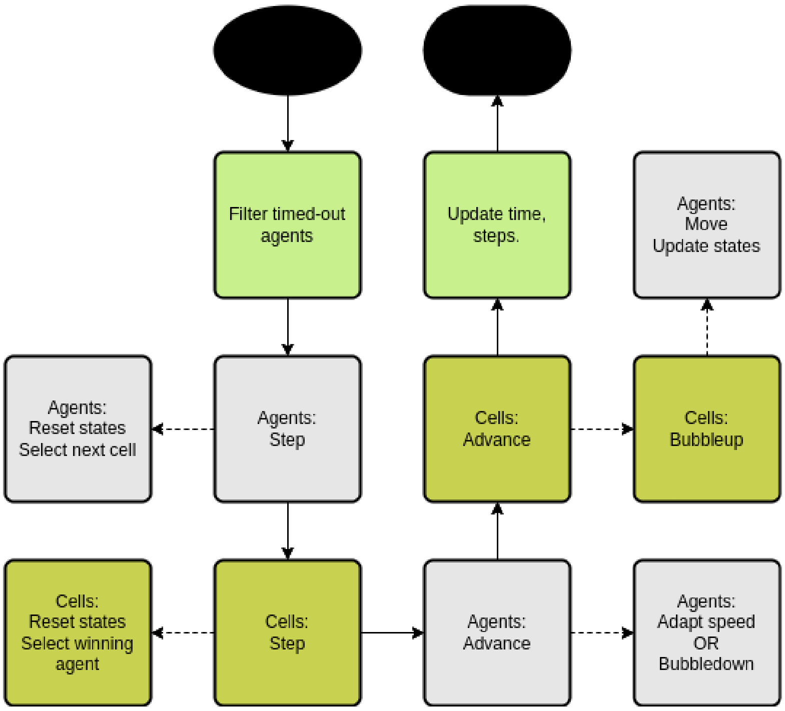

Filter timed-out agents
Update time, steps.
Agents: Move Update states
Agents: Reset states Select next cell
Agents: Step
Cells: Advance
Cells: Bubbleup
Cells: Reset states Select winning agent
Cells: Step
Agents: Advance
Agents: Adapt speed OR Bubbledown

Figure 3.2: Task flow in one step of *SequentialActivation* object.

SequentialActivation is a scheduler of agent movement in the cells. As can be seen in the Figure 3.2, the sequence starts by selecting agents that have $\tau < T$ as described in Section 2.3.4. These agents calculate *attractivity* of the cells in their neighbourhood and stochastically select their desired cell. Next, the cells which are selected by agents for entrance execute winner selection, which means that they select the agent from all candidates that will enter the cell. Next, all participanting agents verify whether they will enter their desired cell. If so, they will update their τ with the duration of the movement. If they can't enter their desired cell, they will *bubbledown* the information to bound agents behind them and cancel their movement as they won't move from their position, see Algorithm 2.

Next, the cells with winning agents *bubbleup* the bounded agents to the front of the bounded chain up to the agent which: a) has empty cell to enter, b) the bounded chain forms a cycle, which is detected and broken, see Algorithm 3. In either case, the agents participating in *bubbleup* will successfully move to their desired step. Lastly, the scheduler updates internal states such as number of steps and timestep value.

Algorithm 2 Agent bubbledown method

```
function BUBBLEDOWN(AGENT)
    if $agent.next\_cell.winner == agent$ then
        $agent.next\_cell.winner \leftarrow null$
    end if
    $agent.next\_cell \leftarrow null$
    $agent.confirm\_move \leftarrow false$
    if $agent.partner.next\_cell \neq null$ then
        $bubbledown(agent.partner)$
    end if
    if $agent.head \neq null$ then
        $agent.head.tail \leftarrow null$
        $agent.head \leftarrow null$
    end if
    if $agent.tail \neq null$ then
        $agent.tail.bubbledown()$
    end if
end function
```

Algorithm 3 Cell bubbleup method

```
function BUBBLEUP(CELL)
    $head \leftarrow cell.winner$
    $origin \leftarrow cell.winner$
    while $head.head \neq null$ do
        $head \leftarrow head.head$
        if $head = origin$ then
            $origin.head.tail \leftarrow null$
            $origin.head \leftarrow null$
        end if
    end while
    return $head$
end function
```

3.1 Mesa

Mesa is a popular framework for ABM research [7]. The API offers many options for visualization, setting of parameters, data collection, etc. This framework has been successfully used in the previous research of SFF CA model of pedestrian evacuation in 2020 [5]. Moreover, Mesa was used in research of software for urban planning practice [8], multi-drone truck routing problem [9] and in the simulation of balancing consumer and business value of recommender systems [10].

3.2 Visualization

Mesa framework has built-in visualization module for web browser, which is suitable for fast development and visual analysis. The user can adjust the speed of simulation, start the simulation and watch the course or look in detail step by step. Simulations can be rerun. Figure 3.3 shows the Mesa visualization in browser for a map *small.txt*. The black circle represents a physical leading agent, the two arrow agents of same color are paired agents with orientation to the west. The color gradient shows *SFF* with high values in green and low values in blue. Black grid cell represent obstacles. The parameter interface is described in Section 3.2.1.

3.2.1 Run-time parameters

The user can better understand the simulation when it runs under various settings. Mesa offers a user-friendly interface for setting global parameters of the simulation. As can be seen in the Figure 3.3, user can set various parameters:

- Sensitivity to static field k_S, where higher value increases the attractivity of the optimal cells as can be deducted from the Equation 2.1.
- Sensitivity to occupied cell k_O, where higher value prevents the agent from selecting occupied cell for next move.
- Sensitivity to diagonal movement k_D, where higher value prevents the agent from selecting diagonal cell for next move.
- Movement duration for leader in timesteps, where higher value means a longer duration of leading agent's movement results in lower speed.
- Movement duration for followers in timesteps, where higher value means a longer duration of following agents' movement results in lower speed.
- Penalization for incorrect orientation, where higher value penalizes maneuvers resulting in incorrect orientation and lowers their attractivity.
- Switch for leader location at the front forces the agent to keep position at the front of the crowd. When the simulation follows goals defined in the map file, this setting will be overwritten by the goal rules.

After setting new parameters, the simulation needs to be restarted.

3.2.2 Datacollector

RoomDatacollector class collects, saves and visualizes data from all simulation runs based on the definition in *Simulation experiments module.* Datacollector provides a simple interface for running multiple experiments and saving, showing or just aggregating the data. At the same time, *RoomDataCollector* is a derived class from the Mesa *DataCollector* class, which can be used to visualize data in real time in the browser visualization.

3.2.3 Definition of map and rules

The maps used by the model are defined in an ASCII text file and can be created or edited for user's needs. The model requires the following structure:

```
maps/
 data/<map_name>.data
 topology/<map_name>.txt
```

An example of the input file format used in the simulation is shown in Listing 3.2.3. The map itself is defined at the beginning of the file using # symbol to represent obstacle, empty space for empty space and various types of agents as follows: *L* for leading agent's positions, *D* for directed agents' positions, other symbols are deprecated. After the map topology definition follows a custom hash value (in example 42), which validates the map topology to the precomputed SFF values in `/maps/data/<map_name>.data`. After that follow sequentially the goals together with rules. As described in Section 2.3, there are three goal types:

- Navigate towards the exit and evacuate is indicated by *E* followed by x, y coordinates and keyword *All*.
- Navigate towards a location with specific position of the leading agent in the crowd is indicated by *L* followed by integer x, y coordinates. Then an integer t tells the leader how long to stay in the position after the goal was reached. Position of the leader is marked by keyword *Front* and *Back*. Keyword *All* ends the sequence.
- Leading agent standing guard at a location and following agents navigate towards next goal location is indicated by *L* followed by integer x, y coordinates of leading agent's location. Then an integer t tells the leader how long to stay in the position after the goal was reached. Position of the leader is marked by keyword *Front* and *Back*. Keyword *All* ends the sequence.

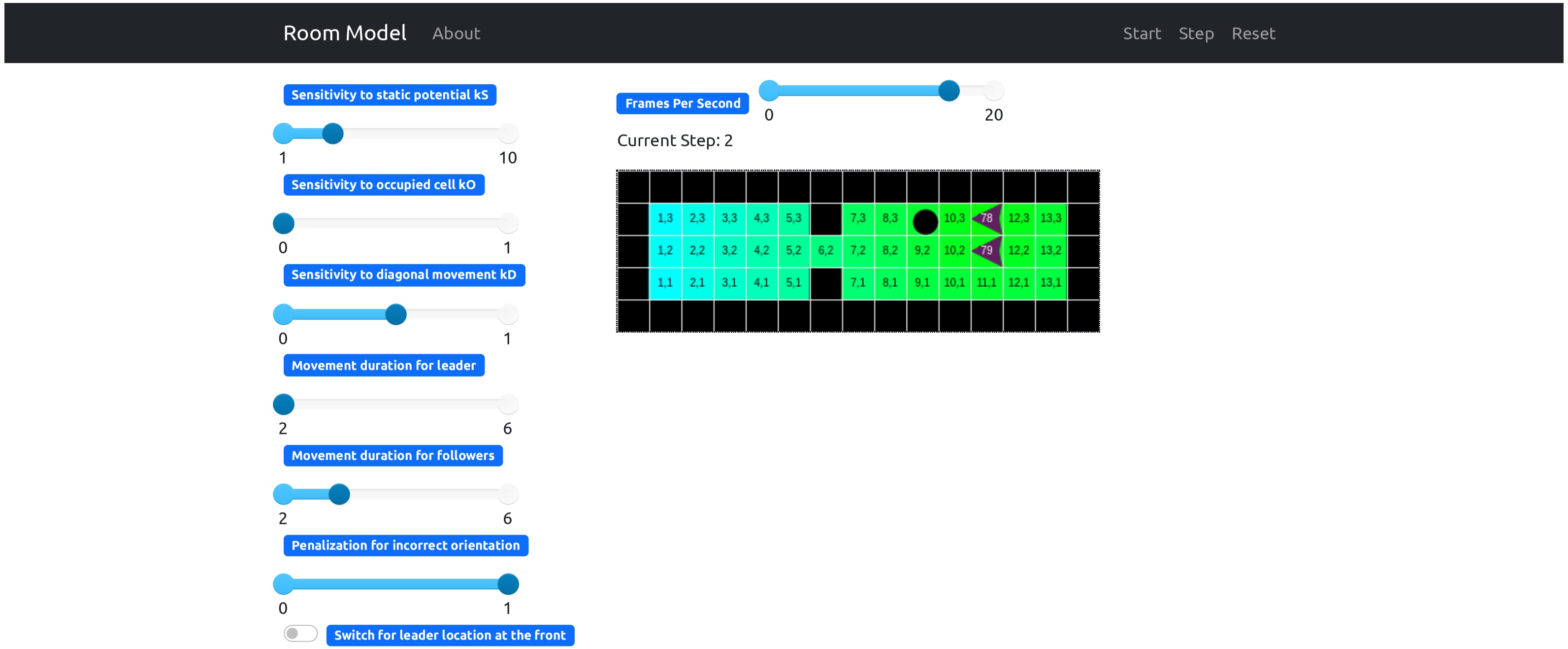
```
###############
#     #    LD #
#           D #
#     #       #
###############
 42
```

```
G 7 3 0 Back All
L 2 2 0 Front All
E 1 2 All
```

3.3 Simulation experiments module

3.3.1 Data

3.3.2 Graphs



Sensitivity to static potential kS

1 10

Sensitivity to occupied cell kO

0 1

Sensitivity to diagonal movement kD

0 1

Movement duration for leader

2 6

Movement duration for followers

2 6

Penalization for incorrect orientation

0 1

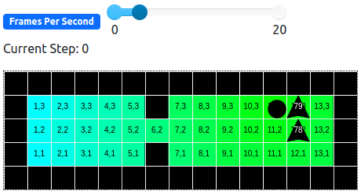
Switch for leader location at the front

Frames Per Second

0 20

Current Step: 2

Figure 3.3: Visualization



Current Step: 0

	1,3	2,3	3,3	4,3	5,3		7,3	8,3	9,3	10,3	●	▲ 79	13,3	
	1,2	2,2	3,2	4,2	5,2	6,2	7,2	8,2	9,2	10,2	11,2	▲ 78	13,2	
	1,1	2,1	3,1	4,1	5,1		7,1	8,1	9,1	10,1	11,1	12,1	13,1	

Figure 3.4: Map *small* initialized.

Chapter 4

Simulation experiments

4.1 Maps

Table of maps, description, reasoning for dimensions and topology

4.2 Parameters, rules, settings

Table or matrix with explanation

4.2.1 Room size for experiments

Based on the legislative of the civil engineering law in Czech republic children in pre-school buildings need at least $1.65m^2$ per person during education process in classroom. Around 70 pupils in preschool.

Minimal width of corridor is 1.2m and 2.0 is recommended Doors at least 90cm

Chapter 5

Results

5.1 Gaps between pair

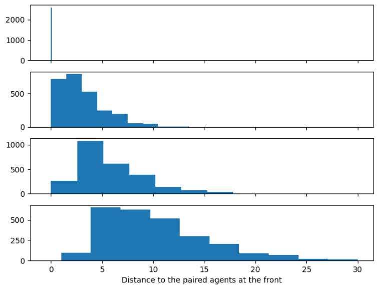


Figure 5.1: Map gaps

5.2 Pairs overtaking

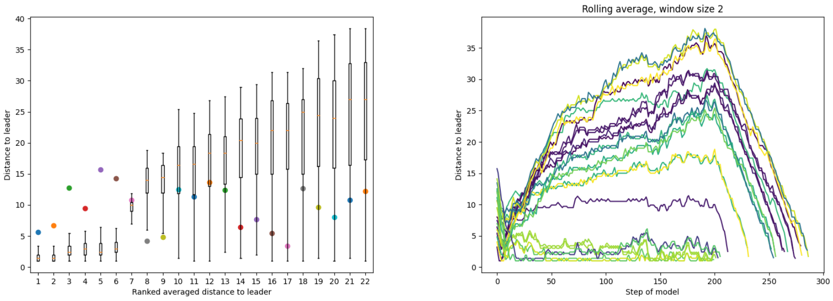


Figure 5.2: Map01

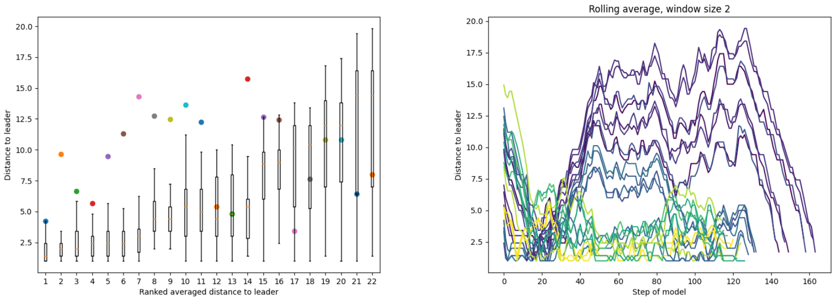


Figure 5.3: Map12

5.3 Crowd structure

5.4 Influence of penalization of incorrect maneuvers

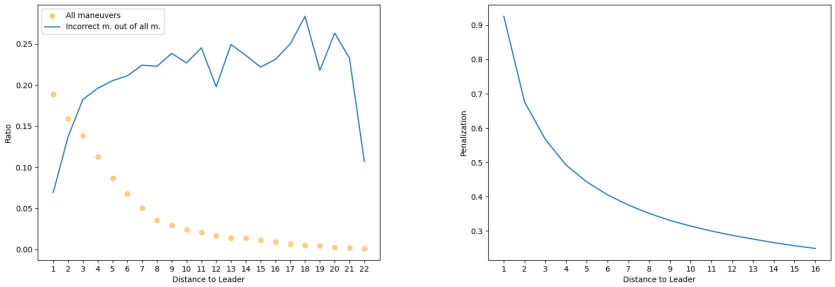


Figure 5.4: Map12

5.5 Specific flow

5.6 Total evacuation time

5.7 Pair formation

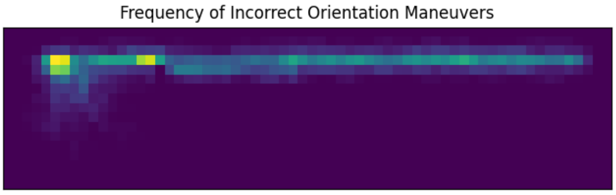


Figure 5.5: Map12

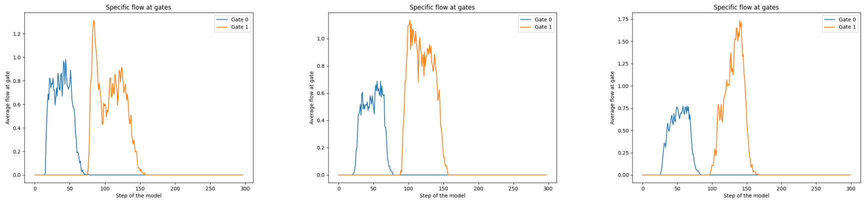


Figure 5.6: Specific flow maps 01, 02, 03.

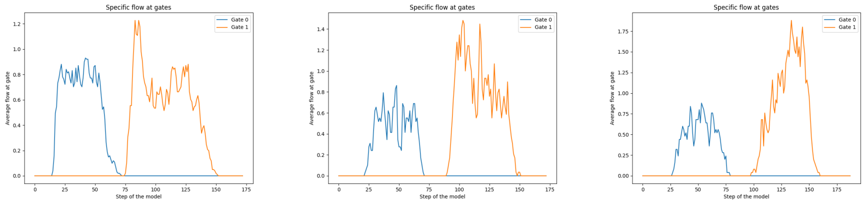


Figure 5.7: Specific flow maps 11, 12, 13.

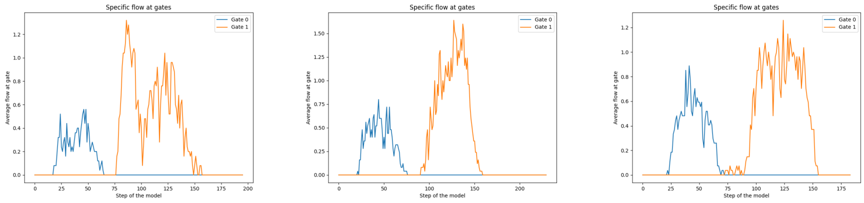


Figure 5.8: Specific flow maps 11, 12, 13.

Conclusion

Bibliography

[1] Najmanová, H. *Evacuation of Pre-school Children Aged from 3 to 6 Years.* Dissertation thesis, Czech Technical University in Prague, 2021.

[2] Janovská, K. *Hierarchické řízení rojů při evakuaci.* B.S. thesis, Czech Technical University in Prague, 2022.

[3] Nishinari, K.; Kirchner, A.; et al. Extended Floor Field CA Model for Evacuation Dynamics. *IEICE Transactions on Information and Systems*, volume E87-D, 07 2003.

[4] Huang, R.; Zhao, X.; et al. Static floor field construction and fine discrete cellular automaton model: Algorithms, simulations and insights. *Physica A: Statistical Mechanics and its Applications*, volume 606, 2022: p. 128150, ISSN 0378-4371, doi:https://doi.org/10.1016/j.physa.2022.128150. Available from: `https://www.sciencedirect.com/science/article/pii/S0378437122007099`

[5] Šutý, M. *Conflict solution in cellular evacuation model.* B.S. thesis, Czech Technical University in Prague, 2021. Available from: `https://dspace.cvut.cz/bitstream/handle/10467/95146/F8-BP-2021-Suty-Matej-thesis.pdf`

[6] Gorrini, A.; Bandini, S.; et al. Group Dynamics in Pedestrian Crowds Estimating Proxemic Behavior. *Transportation Research Record: Journal of the Transportation Research Board*, volume 2421, January 2014: pp. 47–55, doi:10.3141/2421-06.

[7] Kazil, J.; Masad, D.; et al. Utilizing Python for Agent-Based Modeling: The Mesa Framework. In *Social, Cultural, and Behavioral Modeling*, edited by R. Thomson; H. Bisgin; C. Dancy; A. Hyder; M. Hussain, Cham: Springer International Publishing, 2020, pp. 308–317.

[8] Yap, W.; Janssen, P.; et al. Free and open source urbanism: Software for urban planning practice. *Computers, Environment and Urban Systems*, volume 96, 2022: p. 101825.

[9] Leon-Blanco, J. M.; Gonzalez-R, P. L.; et al. A multi-agent approach to the truck multi-drone routing problem. *Expert Systems with Applications*, volume 195, 2022: p. 116604.

[10] Ghanem, N.; Leitner, S.; et al. Balancing consumer and business value of recommender systems: A simulation-based analysis. *Electronic Commerce Research and Applications*, volume 55, 2022: p. 101195.

Appendix A

Acronyms

CA Cellular automaton

SFF Static floor field

OFF Occupancy floor field

TET Total evacuation time

ABM Agent-based model

Appendix B

Contents of enclosed CD

```
readme.txt ........................ the file with CD contents description
exe ..................................... the directory with executables
src ......................................the directory of source codes
  wbdcm ..................................... implementation sources
  thesis ..............the directory of LaTeX source codes of the thesis
text .......................................the thesis text directory
  thesis.pdf..........................the thesis text in PDF format
  thesis.ps.............................the thesis text in PS format
```