Undefined 1 (2014) 1–5
IOS Press

When Reactive Agents Are Not Enough: Tactical Level Decisions in Pedestrian Simulation

Luca Crociani, Andrea Piazzoni, Giuseppe Vizzari * and Stefania Bandini
CSAI - Complex Systems & Artificial Intelligence Research Center,
University of Milano-Bicocca, Milano, Italy
E-mail: {name.surname}@unimib.it

Abstract. Pedestrian and crowd simulation is generally focused on operational level decisions, providing the choice of exact steps of pedestrians in a representation of the environment, with the aim of replicating observed patterns of space utilization, trajectories and timings. When relatively large environments are considered, though, tactical level decisions become equally important: in general, multiple paths can be followed to reach a target from an entrance or starting point, and path length might not be the only reasonable criterion. This paper presents a hybrid agent architecture for modeling different types of decisions in a pedestrian simulation system, encompassing a floor-field based operational level (based on a "least effort" principle) and an adaptive tactical level component, provided with a graph-like representation of the envornment, considering both perceived congestion and characteristics of potential paths in the related decision. The model is experimented and evaluated both qualitatively and quantitatively in benchmark scenarios to show its adequacy and expressiveness.

Keywords: Pedestrian Simulation, Tactical Level, Hybrid Agents

1. Introduction

James A. Hendler, in a provocative letter from the editor in an IEEE Intelligent Systems issue in 2007 [22], expressed his perception that results provided by this line of research were not up to the initially raised expectations. Among the different responses this letter triggered, Peter Mc Burney and Micheal Luck [28] indicated several successful cases of application of agent technologies and, as already pointed out in [27], they also noted that optimization and simulation represent the primary application domains of these approaches.

In most disciplines studying complex systems by means of simulation, agent based approaches have indeed become a widespread choice, even though specifically adopted models, mechanisms and technologies are not necessarily up-to-date or in line with the most current results in the computer science and engineering area about agent technologies [3].

The simulation of pedestrians and crowds is an example of a consolidated but still lively research context: both the automated analysis and the synthesis of pedestrian and crowd behaviour, as well as attempts to integrate these complementary and activities [35], present open challenges and potential developments in a smart environment perspective [30]. These challenges are not only related to technical or technological issues, but they often lead to interdisciplinary questions on how to model human behaviour.

Even if we only consider choices and actions related to walking, modelling human decision making activities and actions is a complicated task: different types of decisions are taken at different levels of abstraction, from path planning to the regulation of distance from other pedestrians and obstacles present in the environment. Moreover, the measure of success and validity of

*Corresponding author.

a model is definitely not the *optimality* with respect to some cost function, as in robotics, but the *plausibility*, the adherence to data that can be acquired by means of observations or experiments. Putting together *tactical* and *operational* level decisions, often adopting different approaches (typically behaviour-based for operational decisions, and at knowledge level for tactical ones) in a comprehensive framework preserving and extending the validity that, thanks to recent extensive observations and analyses (see, e.g., [9]), can be achieved at the operational level represents an urgent and significant open challenge.

This paper presents a hybrid agent architecture for modeling different types of decisions in a pedestrian simulation system. In particular, the present work focuses on *tactical level* decisions that are essentially related to the choice of a route to follow in an environment comprising several rooms connected by openings. These decisions are then enacted at the operational level by means of a floor-field based model, in a discrete simulation approach. The described model that integrates within an organised and comprehensive framework different spatial representations, types of knowledge and decision making mechanisms, allows the agent taking decisions based on a static a-priori knowledge of the environment and dynamic perceivable information on the current level of crowdedness of visible path alternatives.

The paper presents the relevant state of the art in the following Section. The tactical level part of the model is formally presented in Section 3 whereas its experimental application in benchmark scenarios showing the adequacy in providing adaptiveness to the contextual situation is given in Section 4.

2. Related Works

The research on pedestrian dynamics is basically growing on two lines. On the *analysis* side, literature is producing methods for an automatic extraction of pedestrian trajectories (e.g. [9,10]), charactertization of pedestrian flows (e.g. [24]) automatic recognition of pedestrian groups [33], recently gaining importance due to differences in trajectories, walking speeds and space utilization [5]. The *synthesis* side – where the contributions of this work are concentrated – has been even more prolific, starting from preliminary studies and assumptions provided by [21] or [19] and leading to quite complex, yet not usually validated, models exploring components like panic [11] or other emotional variables. To better understand the model presented in the next section, the following will provide a brief description of related works on pedestrian dynamics modeling and simulation.

[32][1] provides a well-known scheme to model the pedestrian dynamics, describing 3 levels of behavior:

- **Strategic level**: the person formulates his/her abstract plan and final objective motivating the overall decision to move (e.g. "I am going to the University today to follow my courses and meet my friend Paul");
- **Tactical level**: the set of activities to complete the plan is computed and scheduled (e.g. "I am going to take the 8:00 AM train from station XYZ then walk to the Department, then meet Paul at the cafeteria after courses, then ..."");
- **Operational level**: each activity is physically executed, i.e., the person performs the movement from his/her position to the current destination (e.g. precise walking trajectory and timing, such as a sequence of occupied cells and related turn in a discrete spatial representation and simulation).

Most of the literature has been focussed on the reproduction of the physics of the system, so on the lowest level, where a significant knowledge on the fundamental diagram achieved with different set of experiments and in different environment settings (see, e.g, [39,40]) allows a robust validation of the models.

Literature of this level can be classified regarding the scope of the modeling approach. Macroscopic models describe the earliest approach to pedestrian modeling, based on analogies between behavior of dense crowds and kinetic gas [21] or fluids [19], but essentially abstracting the concept of individual. A microscopic approach is instead focused on modeling the individual behavior, effectively improving the simulations precision also in low density situations.

The microscopic approach is as well categorized in two classes describing the representation of space and movement: continuous models simulate the dynamics by means of a force-based approach, which finds its basis on the well-known social force model by [20]. These models design pedestrians as particles moved by virtual forces, that drive them towards their destination and let them avoid obstacles or other pedestrians. Latest models in this class are the centrifugal force model by [14] and the stride length adaptation model by [37].

[1] A similar classification can be found in vehicular traffic modeling from [29].

Other examples consider also groups of pedestrians by means of attractive forces among persons inside the group [31].

The usage of a discrete environment is mostly employed by the cellular automata (CA) based models, and describes a less precise approach in the reproduction of individuals trajectories that, on the other side, is significantly more efficient and still able to reproduce realistic aggregated data. This class derives from vehicular modeling and some models are direct adaptations of traffic ones, describing the dynamics with ad hoc rules (e.g. [7,8]). Other models employs the well-know *floor field* approach from [12], where a *static* floor field drives pedestrians towards a destination and a *dynamic* floor field is used to generate a lane formation effect in bi-directional flow. [34] is an extension of the floor field model, introducing the *anticipation* floor field used to manage crossing trajectories and encourage the lane formation. [25] discussed methods to deal with different speeds, in addition to the usage of a finer grid discretization that decreases the error in the reproduction of the environment, but significantly impacts on the efficiency of the model. An alternative approach to represent different speeds in a discrete space is given by [6]. [36] is another extension of the floor-field model, where groups of pedestrians are also considered.

The tactical level has gained interest only recently in the literature of pedestrian dynamics modeling and simulation, despite its relevance for the simulation of a realistic behavior (especially by thinking to evacuation situations). Path planning algorithms have been widely investigated and proposed in the field of computer graphics and gaming by means of graph-based methods (e.g. [16,17]), but with aims not necessarily matching the requirements of pedestrian simulation, since the point is mainly to reach a visual realism.

Relevant recent works, such as [18] and [38], start exploring the implications of tactical level decisions during evacuation. In particular, [18] modifies the floor-field Cellular Automata approach for considering pedestrian choices not based on the shortest distance criterion but considering the impact of congestion on travel time. [38] explores the implications of four strategies for the route choice management, given by the combination of applying the shortest or quickest path, with a local (i.e., minimize time to vacate the room) or global (i.e., minimize overall travel time) strategy. The global shortest path is calculated with the well-known Floyd-Warshall algorithm, implying computational times that can become an issue when a large number of nodes is present or when special features in the simulated population are considered (i.e. portion of the path where the cost differs from an agent to another). This paper proposes an alternative and efficient approach to find a global path, where each agent will be able to consider additional costs in sub-paths without adding particular weight to the computation.

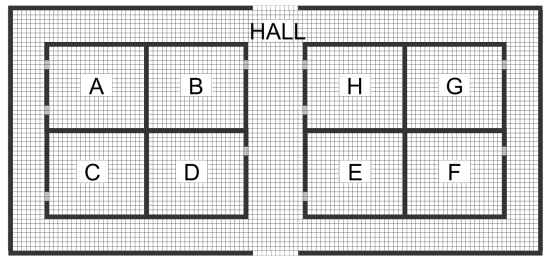
3. A Model for Tactical Level of Pedestrians

The model described in this work provides a methodology to deal with tactical choices of agents in pedestrian simulation systems. Due to constraints on the length of the paper, the description of the part of the model dedicated to the operational level, thoroughly described in [4], will not be provided.

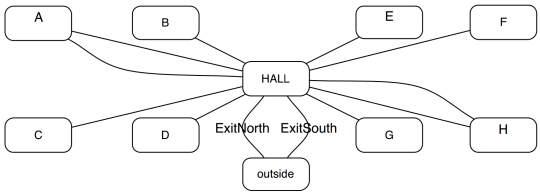
3.1. A Cognitive Representation of the Environment for Static Tactical Choices

The framework that enables agents to perform choices on their plan implies a graph-like, topological, representation of the walkable space, whose calculation is defined in [15] and briefly reported in this section. This model allows agents to perform a static path planning, since dynamical information such as congestion is not considered in the graph. These additional elements will be considered in the extension that is presented in the next section which represents the innovative part of this paper.

The environment abstraction identifies *regions* (e.g. a room) as nodes of the labeled graph and *openings* (e.g. a door) as edges. This so-called *cognitive map* is computed starting from the information of the simulation scenario, provided by the user and necessarily containing: (i) the description of the walkable space, that is, the size of the simulated environment and the positions of obstacles and walls; (ii) the position of final destinations (i.e. exits) and intermediate targets (e.g. a ticket machine); borders of the logical regions of the environment that, together with the obstacles, will define them. Approaches to automatically configure a graph representation of the space, without any additional information by the user, have been already proposed in the literature (e.g. [26]), but they are not leading to a cognitively logical description, i.e., a *topological* map. A cognitive definition of the space allows, in fact, a proper definition of portions of the environment where, for example, the behavior of a person is systematically different (e.g. the change of speed profile in



(a)



(b)

Fig. 1. An example environment (a) with the resulting cognitive map (b), by applying the procedure from [15].

stairs or ramps), or that contain relevant intermediate targets for the agent plan (e.g. a ticket machine).

The cognitive map is defined as a graph $\mathcal{CM} = (\mathcal{V}, \mathcal{E})$ generated with a procedure included in the floor field diffusion, starting from the statements that each user-defined opening generates a floor field from its cells and spread only in the regions that it connects, and that each region has a flag indicating its properties among its cells. The floor fields diffusion procedure iteratively adds to $\mathcal{CM}$ the couple of nodes found in the diffusion (duplicates are avoided) and labeled with the region id and the edge labeled with the id of the opening. Each *final destination*, different from the normal openings since it resides in only one region, will compose an edge linking the region to a special node describing the external *universe*. Intermediate targets will be mapped as attributes of its region node. An example of environment together with the resulting cognitive map is presented in Fig. 1.

To allow the calculation of the *paths tree*, that will be described in the following section, functions $Op(\rho)$ and $Dist(\omega_1, \omega_2)$ are introduced describing respectively: the set of openings accessible from the region ρ[2] and the distance between two openings linking the same arbitrary region. While the first one is trivial and outputs the edges linking ρ, the function $Dist(\omega_1, \omega_2)$ describes the distance that will be perceived by agents for their path planning calculation. To obtain a scalar from the sets of cells associated to ω_1 and ω_2, the value of the floor field in their center cell is used, defined as:

$$Center(\omega) = \left(\left\lfloor \frac{\sum x_i}{|\omega|} \right\rfloor, \left\lfloor \frac{\sum y_i}{|\omega|} \right\rfloor \right), (x_i, y_i) \in \omega.$$

The distance between ω_1 and ω_2 is then calculated as the average between the floor field values in the two center cells, i.e., the value of the floor field of ω_1 in $Center(\omega_2)$ and vice-versa.

3.2. Region classes

This model allows to assign a class to each region, identifying different environments. This is particularly important since a typical environment is composed of different elements, such as stairs, halls, ramps and so on. The behavior inside each of these classes is different: the speed is different and different agents may have different preferences. Moreover, some regions may also be one way, such as an escalator or a mobile ramp, or imply different behaviors in regard to the direction.

As an example, we have defined these classes:

- *normal*
- *stair*
- *ramp*
- *escalator*
- *mobile ramp*

These example are a combination of the different aspects we have identified. Stairs and escalators may be precluded to some agents, while ramps and mobile ramps are accessible to everyone. Escalators and mobile ramps are one way and have a constant speed, while stairs and ramp can be undertaken in both direction, but maybe with different behavior. To identify the direction of the classes, we can simply mark one (or more) opening. For ramps and stairs the marked opening will mean that the opening is at the top of the region. For escalators or mobile ramps it will describe the direction.

3.3. Modeling Adaptive Tactical Decisions with A Paths Tree

To enhance the route choice and enable dynamical, adaptive decisions of the agents in an efficient way, a new data structure has been introduced, containing in-

[2] Its Id, described in the label of the edge mapped to it.

formation about the cost of *plausible* paths towards the exit from each region of the scenario.

Using the well-known Floyd-Warshall algorithm, in fact, can solve the problem but introduces issues in computational time: the introduction of dynamical elements in the paths cost computation (i.e. congested paths) implies a re-computation of the cost matrix underlying the algorithm every step. More in details, the penalty of a congested path is a subjective element for the agents, since they are walking with different desired velocities, thus the calculation cost increases also with the number of agents.

The approach here proposed implies an off-line calculation of the data-structure that we called *paths tree*, but is computationally efficient during the simulation and provides to the agents direct information about the travel times describing each path. The method is described in the following paragraphs.

3.3.1. The Paths Tree

We define the *Paths Tree* as a tree data-structure containing the set of plausible paths towards a destination, that will be its root. Before describing what we mean with the attribute plausible, that can be seen as a fuzzy concept, a general definition of path must be provided.

A path is defined as a finite sequence of openings $X \rightarrow Y \rightarrow \ldots \rightarrow Z$ where the last element represents the final destination. It is easy to understand that not every sequence of openings represents a path that is walkable by an agent. Firstly, a path must be a sequence of consecutive oriented openings regarding the physical space.

Definition 3.1 (Oriented opening)**.** Let $E = R_1, R_2$ be an opening linking the regions R_1 and R_2, (R_1, E, R_2) and (R_2, E, R_1) define the oriented representations of E.

An oriented opening will therefore describe a path from an arbitrary position of the first region towards the second one.

Definition 3.2 (Valid path)**.** let C a sequence of oriented openings $X \rightarrow \ldots \rightarrow Z$. C is a valid path if and only if:

- $|C| = 1$
- $|C| = 2$: by assuming $C = X- > Y$, the third element of the triple X must be equal to the first element of Y
- $|C| > 2$: each sub-sequence S of consecutive openings in C where $|S| = 2$ must be a valid path.

The last oriented opening in the path leads to the *universe* region.

Given a set of paths, the agent will consider only complete paths towards its goal, starting from the region where the agent is located.

Definition 3.3 (Start and Destination of a path)**.** Given p a path $(R_1, E, R_x) \rightarrow \ldots \rightarrow (R_y, O, \text{universe})$, the function $RS(p) = R_1$ will return the region R_1 where an agent can start the path p. $S(p) = E$ and $D(p) = O$ will respectively return the first opening (E) and the destination (O) of the path.

Definition 3.4. Let p a path, T(p) is the function which return the expected travel time from the first opening to the destination.

$$T(p) = \sum_{i \in [1, |p|-1]} \frac{Dist(opening_i, opening_{i+1})}{speed} \quad (1)$$

Another basic rule is that a path must be loop-free: by assuming the aim to minimize the time to reach the destination, a plan passing through a certain opening more than once would be not plausible.

Definition 3.5 (opening loop constraint)**.** A path $X \rightarrow \ldots \rightarrow Z$ must not contain duplicated openings.

This will not imply that an agent cannot go through a certain opening more than once during the simulation, but this will happen only with a change of the agent plan.

By assuming to have only convex regions in the simulated space, we could easily achieve the set of plausible paths by extending 3.5 as to let a path not containing duplicated regions. However, since the definition of region describes also rooms, concave regions must be considered. Some paths may, thus, imply to pass through another region and then return to the first one to reduce the length of the path.

As we can see in Figure 2, both paths starts from r_1, go through r_2, and then return to r_1. However, only the path represented by the continuous line is plausible, even if the constraint 3.5 is respected by both of them. Before the definition of the constraint that identifies the correct paths, the concept of *sub-path* has to be defined.

Definition 3.6 (Sub-path)**.** Let p be a path, a sub-path p' of p is a sub-sequence of oriented openings denoted as $p' \subset P$ which respects the order of appearance for the openings in p, but the orientation of openings in p' can differ from the orientation in p. p' must be a valid path.

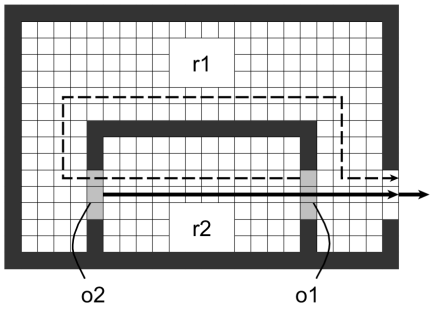


Fig. 2. A concave region can imply the plausibility of a path crossing it twice, but its identification is not elementary.

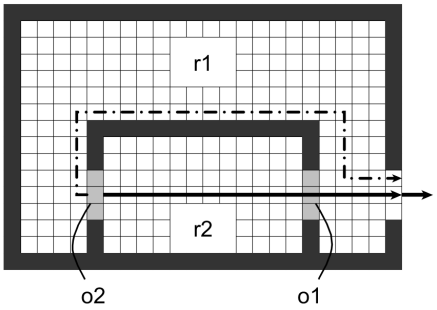


Fig. 3. The correct paths for this environment. Inside $r2$ the choice between the two openings is also determined by the congestion.

The reason of the orientation change can be explained with the example in Fig. 3: given the path $p = (r_1, o_2, r_2) \rightarrow (r_2, o_1, r_1) \rightarrow universe$, the path $p' = (r_2, o_2, r_1) \rightarrow universe$ is a valid path and is considered as a sub-path of p, with a different orientation of o_2. In addition, given the path $p_1 = (r_2, o_2, r_1) \rightarrow universe$, the path $p_2 = (r_1, o_2, r_2) \rightarrow (r_2, o_1, r_1) \rightarrow universe$ is as well a minimal path if and only if the travel time of p_2 is less than p_1. It is easy to understand that this situation can emerge only if r_1 is concave. As we can see, the starting region of the two paths is different, but the key element of the rule is the position of the opening o_2. If this rule is verified in the center position of the opening o_2, this path will be a considerable path by the agents surrounding $o2$ in $r1$.

In Figure 3 the correct paths for this example environment are shown. An agent located in $r2$ can reach $r1$ and then the destination D using both openings considering the congestions. An agent located in $r1$ can go directly to the exit or chose the path $o2 \rightarrow o1 \rightarrow D$.

Definition 3.7 (Minimal path)**.** p is a minimal path if and only if it is a valid path and $\forall p' \subset p : S(p') = S(p) \wedge D(p') = D(p) \implies T(p') > T(p)$

The verification of this rule is a sufficient condition for the opening loop constraint 3.5 and it solves the problem on the region loop constraint independently from the configuration of the environment (i.e. convex or concave regions).

At this point the constraint that defines a minimal path has been provided. This can be used to build the complete set of minimal paths towards a destination before running the simulation. It must be noted that an arbitrary path represents a set of paths itself, since it can be undertaken at any region it crosses. Indeed every path p provides also information about the sub-paths achieved by cutting the head of p with an arbitrary number of elements. So a minimal representation of the set is a tree-like structure defined as:

Definition 3.8 (Paths tree)**.** Given a set of minimal paths towards a destination, the Paths-Tree is a tree where the root represents the final destination and a branch from every node to the root describes a minimal path, crossing a set of openings (other nodes) and regions (edges). Each node has an attribute describing the expected travel time to the destination.

3.3.2. An Algorithm to Compute the Paths Tree

The algorithm we propose builds the the Paths Tree in a recursive way, starting from a path containing only the destination and adding nodes if and only if the generated path respects the definition of minimality.

Formally the Paths Tree is defined as $PT = (N, E)$ where N is the set of nodes and E the set of edges. Each node n is defined as a triple (id, o, τ) where:

- $id \in \mathbb{N}$ is the id of the node
- $o \in \mathcal{O}$ is the name of the opening
- $\tau \in \mathbb{R}^+$ is the expected travel time for the path described by the branch.

Each edge e is defined as a triple (p, c, r) where:

- $p \in \mathcal{O}$ is the id of the parent
- $c \in \mathcal{O}$ is the id of the child
- $r \in \mathcal{R}$ is the region connecting the child node to its parent.

To allow a fast access to the nodes describing a path that can be undertaken from a certain region, we added a structure called M that maps each r in the list of $p : (p, c, r) \in E$ (for every c).

Given a destination $D = (r_x, \text{universe})$, the paths tree computation is defined with the following procedures.

Algorithm 1 Paths tree computation

1: add $(0, D, 0)$ to N
2: add 0 to $M[r_x]$
3: $\forall s \in \mathcal{O}$ ShortestPath[s] $\leftarrow \infty$
4: expand region$(0, D, 0, R_x, ShortestPath)$

With the first line, the set N of nodes is initialized with the destination of all paths in the tree, marking it with the id 0 and expected travel time 0. In the third row the set of $ShortestPath$ is initialized. This will be used to track, for each branch, the expected travel time for the shortest sub-path, given a start opening s. $ExpandRegion$ is the core element of the algorithm, describing the recursive function which adds new nodes and verifies the condition of minimality. The procedure is described by Alg. 2.

Algorithm 2 ExpandRegion

Require: input parameters $(parentId,\ parentName,\ parentTime, RegionToExpand, ShortestPath)$
1: $expandList \leftarrow \emptyset$
2: $opList = Op(RegionToExpand) \setminus parentName$
3: **for** $o \in opList$ **do**
4: $\tau = parentTime + \frac{D(o,parentName)}{speed}$
5: **if** $CheckMin(ShortestPath, o, \tau)$ == True **then**
6: add (id, o, τ) to N
7: add $(parentId, id, r)$ to E
8: $ShortestPath[o] \leftarrow \tau$
9: $nextRegion = o \setminus r$
10: add id to $M[nextRegion]$
11: add $(id, o, \tau, nextRegion)$ to $expandList$
12: **end if**
13: **end for**
14: **for** $el \in expandList$ **do**
15: $ExpandRegion(el, ShortestPath)$
16: **end for**

In line 2 a list of openings candidates is computed, containing possible extensions of the path represented by $parentId$. Selecting all the openings present in this region (except for the one labeled as $parentName$) will ensure that all paths eventually created respect the validity constraint 3.2.

At this point, the minimality constraint 3.7 has to be verified for each candidate, by means of the function $CheckMin$ explained by Alg. 3. Since this test requires the expected travel time of the new path, this has to be computed before. A failure in this test means that the examined path – created by adding a child to the node $parentId$ – will not be minimal. Otherwise, the opening can be added and the extension procedure can recursively continue.

In line 6, id is a new and unique value to identify the node, which represents a path starting from the opening o and with expected travel time τ; line 7 is the creation of the edge from the parent to the new node. In line 8, $ShortestPath[o]$ is updated with the new discovered value τ. in line 9 the opening is examined as a couple of region, selecting the one not considered now. In fact, the element $nextRegion$ represents the region where is possible to undertake the new path. In line 10 the id of the starting opening is added to $M[nextRegion]$, i.e., the list of the paths which can be undertaken from $nextRegion$. In line 11 the node with his parameter is added to the list of the next expansions, which take place in line 13-14. This passage has to be done to ensure the correct update of $ShortestPath$.

Algorithm 3 CheckMin

Require: input parameter $(ShortestPath, o, \tau)$
1: **if** $ShortestPath[o] > \tau$ **then**
2: **return** True
3: **else**
4: **return** False
5: **end if**

To understand how the constraint of minimality is verified, two basical concepts of the procedure need to be clarified. Firstly, the tree describes a set of paths towards a unique destination, therefore given an arbitrary node n, the path described by the parent of n is a subpath with a different starting node and leading to the same destination. Furthermore, the expansion procedure implies that once reached a node of depth l, all the nodes of its path having depth $l - k$, $k > 0$ have been already expanded with all child nodes generating other minimal paths.

Note that the variable $ShortestPath$ is particularly important since, given p the current path in evaluation, it describes the minimum expected travel time to reach the destination (i.e. the root of the tree) from each opening already evaluated in previous expansions

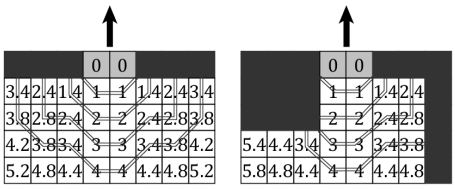


Fig. 4. Examples of surroundings of different sizes, for two configurations of the environment.

of the branch. Thus, if τ is less than $ShortestPath[o]$, the minimality constraint 3.7 is respected.

3.3.3. Congestion Evaluation

The explained approach of the paths tree provides information on travel times implied by each path towards a destination. By only using this information, the choice of the agents will be still static, essentially describing the shortest path. This may also lead to an increase of the experienced travel times, since congestion may emerge without being considered.

For the evaluation of congestion, we provide an approach that estimates, for each agent, the additional time deriving by passing through a jam. The calculation considers two main aspects: the size of the eventually arisen congestion around an opening; the average speed of the agents inside the congested area. Since the measurement of the average speed depends on the underlying model that describes the physical space and movement of the agents, we avoid to explain this component with full details, by only saying that the speed is estimated with an additional grid counting the *blocks* (i.e. when agents maintain positions at the end of the step) in the surrounding area of each opening. The average number of blocks defines the probability to move into the area per step, thus the speed of the agents inside. For the size of the area, our approach is to define a minimum radius of the area and (i) to increase it when the average speed becomes too low or (ii) to reduce it when it returns normal.

As we can see in Figure 4, the presence of an obstacle in the room is well managed by using floor field while defining the area for a given radius. If a lot of agents try to go through the same opening at the same time, a congestion will arise, reducing the average speed and letting the area to increase its size. When this one becomes too big and the farthest agents inside are not slowed by the congestion, the average speed will start increasing until the area re-sizing will stop.

During this measurement the average speed value for each radius is stored. Values for sizes smaller than the size of the area will be used by the agents inside it, as it will be explained in the next section. Two functions are introduced for the calculation:

- $size(o)$: return the size of the congestion around the opening
- $averageSpeed(o, s)$: return the average speed estimated in the area of size s around the opening o.

3.3.4. Agents Dynamical Path Choice

At this point we have defined which information an agent will use to make its decision:

- the Paths Tree, computed before the simulation, will be used as a list of possible path choice;
- the position of the agent, will be used to adjust the expected travel time considering the distance between the agent and the first opening of a path ($d(a, o)$);
- the information about congestion around each opening, computed during the simulation, will be used to estimate the delay introduced by each opening in the path.

The agent, who knows in which region R_x he is located, can access the Paths Tree using the structure $M[R_x]$. The structure returns a list of nodes, each representing the starting opening for each path. At this point the agent can compute the expected travel time to reach each starting opening and add it to the travel time τ of the path.

To consider congestion, the agent has to estimate the delay introduced by each opening in a path, by firstly obtaining the size of the jammed area.

$$size_a(o) = \begin{cases} size(o) & \text{if } d(a,o) \geq r(o) \\ d(a,o) & \text{otherwise} \end{cases} \quad (2)$$

where $size_a(o)$ represents the estimate of the size of the area o by agent a, which is related to the actual size ($size(o)$) and the current position of the agent; more precisely, if its distance from the opening is smaller than the radius of the area (i.e. it is actually inside the area) it will not consider the whole size of the congestion but rather the remaining steps to the opening.

At this point, the agent can suppose that for the length of the area it will travel at the average speed around the opening.

$$delay(o) = \frac{size_a(o)}{averageSpeed(o)} - \frac{size_a(o)}{speed_a} \quad (3)$$

where $averageSpeed(o)$ is the average speed of agents in the area around the opening and $speed_a$ is the desired walking speed of agent a.

If the agent is slower than the average speed around an opening, the delay will be lower than 0. In this case it is assumed that the delay is 0, implying that the congestion will not increase his speed.

At this point the agent can estimate the delay introduced by all openings.

$$pathDelay(p) = \sum_{o \in p} delay(o) \quad (4)$$

This is an example of omniscient agents, since they can always know the status of each opening. Another option is to suppose that the agent can only see the state of the opening located in the same region of the agent. In this situation the agent must be able to remember the state of the opening when it left a region, otherwise the information used to estimate the travel time at each time will not be consistent during the execution of the plan.

$$Time(p) = \tau_p + \frac{d(a, S(p))}{speed_a} + pathDelay_a(p) \quad (5)$$

Where:

- τ_p : the expected travel time of the path p
- $\frac{d(a,S(p)}{speed_a}$: the expected time to reach $S(p)$ from the position of the agent
- $pathDelay_a(p)$: the estimation of the delay introduced by each opening in the path, based on the memory of the agent (which may or may not be updated for each opening).

3.4. Agents classes

The previous model is general for each class of agents defined by the speed parameter. To represent different classes of agents, we define a function $Speed(agentClass, regionClass, direction)$ which will return the speed of the agent class inside the region of type $regionClass$, given the direction. It is essentially a generalization of the value $speed_a$ used in Equation 3.3.4.

For the experiments presented in this paper, three classes of agents have been identified:

- *normal*: it is a normal agent, with no particular preferences
- *special*: it is a special agent, with a low base speed and which is slowed even more on stairs and ramps
- *selective*: it is a selective agent, which deliberately avoid stairs and escalator

Given the function $Speed$, a paths tree for each class of agent is generated, by adapting the speed parameter in the Function 1, given the room and the direction if needed. In order to represent the selective agent, it is sufficient to let the function $Speed$ return 0 for the region the agent wants to avoid. In this way, the expected travel time of the path will be infinite and this will not be added to the tree, leading the agents to never choose it. This approach can also be used to block the creation of paths that do not respect the eventual *direction* constraint of certain regions: more in details, the function $Speed$ will return 0 also for the region of type *escalator* if the direction of the expansion does not match its direction.

4. Experiments and Analysis of the Model

In this section, first of all we will show the behavior of various classes of simulated pedestrians, as explained in the previous section, in a simple environment that could represent an entrance to a building, comprising a stair and a ramp that can be used by people with mobility impairment. This structure is illustrated in Fig. 5(a). A flow of about 1000 persons populates the scenario, with an arrival rate of about 1.5 persons/second and divided into the three classes with a distribution of 60% normal, 20% special and 20% region selective. Fig. 5 (c) and (d) show two frames of the simulation where the agents of the three classes have been visualized with different colors, respectively red, green and blue. This approach automatically lets the *region selective* agents (blue colored) choose the ramp instead of the stairs.

By looking at the two pictures, the different behavior of agents belonging to the diverse classes is clearly understandable: while the stairs are component of the tactical plan of the largest part of the *normal* and *special* agents, none of the *region selective* pedestrians employed them. This choice has been a-priori constrained with the paths tree computed for this agent class, shown in Fig. 5(b), where the branch describing the stairs path (the dashed one in the figure) has not been added due to the "infinite" time that it would imply (unless helped, a pedestrian with specific mobility

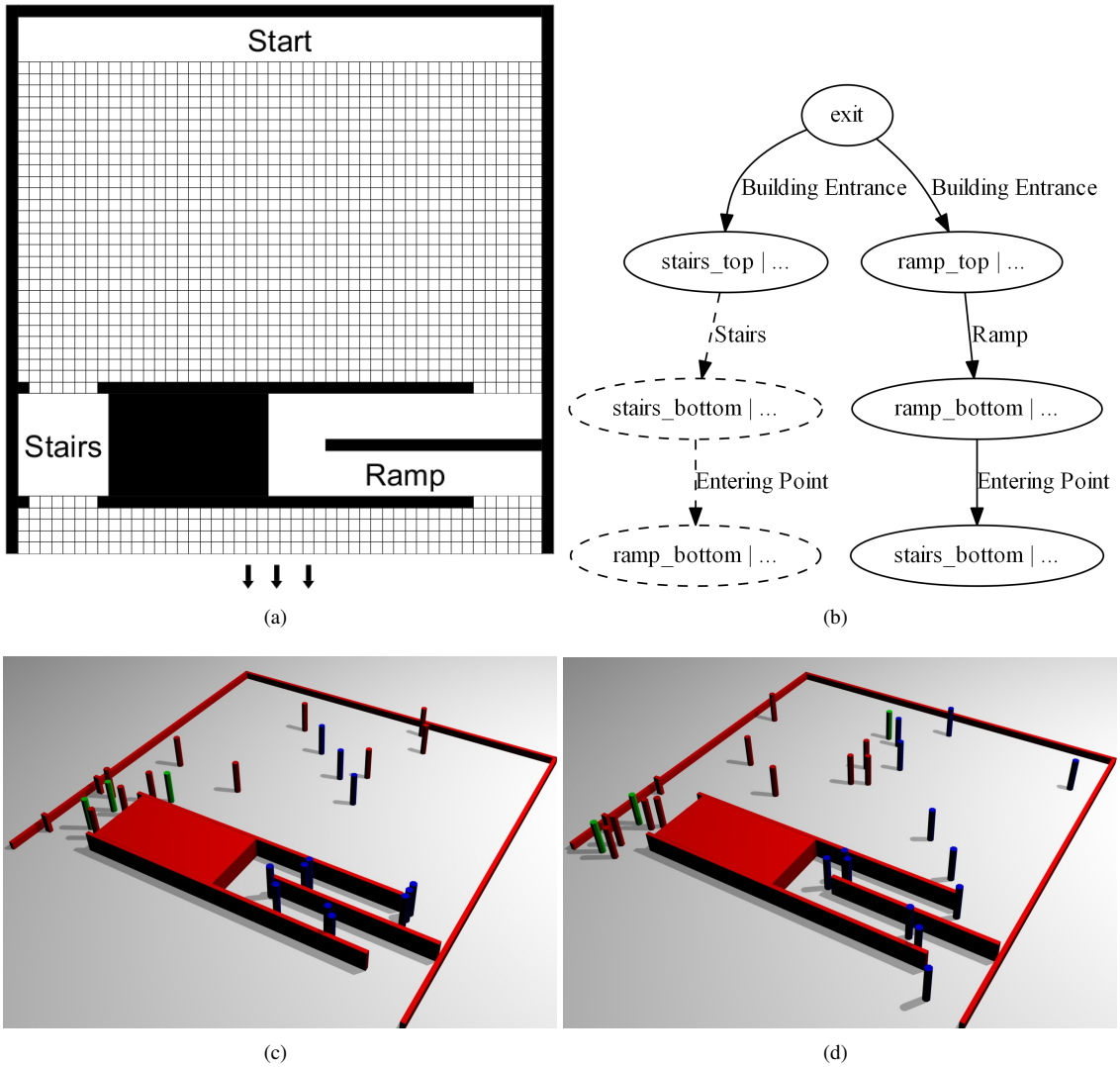


Fig. 5. The qualitative test scenario (a), representing an entrance to a building with a small stair and a ramp for people with mobility impairments. In (b) a "merge" between the paths tree dedicated to *normal* agents and for the *region selective* agents is shown. The dashed branch, passing through the stairs region, appears only in the normal agents tree. Two screenshots of the simulation are shown in (c) and (d).

impairments would simply be unable to climb a stair). On the other hand, the longer distance implied by the usage of the ramp causes agents belonging to the other classes to generally avoid it in favor of the stairs.

Due to the low densities arisen in the scene, the first scenario does not show any sign of the adaptive agent behavior at tactical level. The second scenario, instead, focuses on this point to show the potential of the proposed approach and the possibility to tune its mechanisms to vary the *sensitivity* of the agent dynamical path re-calculation. This parameter influences the frequency of recomputation of the path to be followed due to changes in the perceived level of congestion in the next opening: a high level of sensitivity will lead to more frequent recomputation, potentially detecting more quickly the opportunity to change a sub-optimal plan but also potentially leading to higher computational costs and even excessive oscillations in agents' decisions.

The second experiment proposes an evacuation in a hypothetical scenario, simulated with a consistent incoming flow of people. A graphical representation of the environment and flow configuration is depicted in Fig. 6(a): it is a sample situation in which two flows

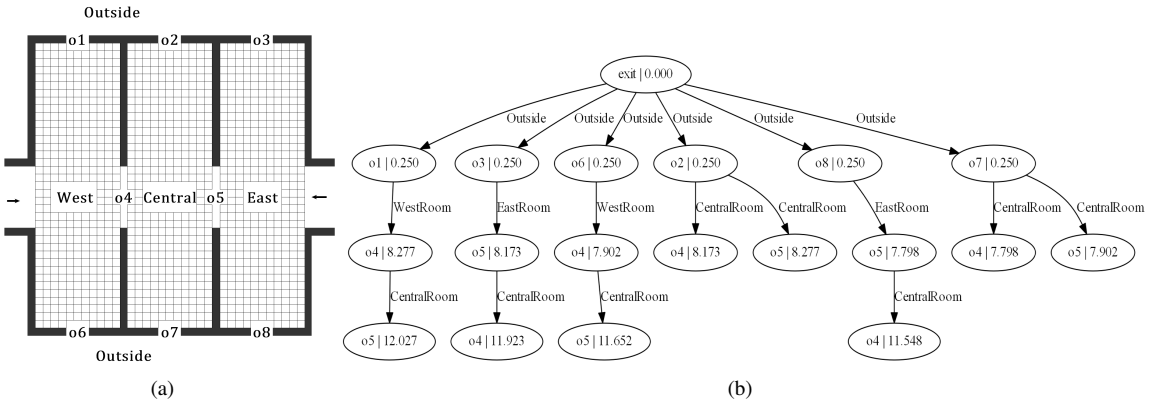


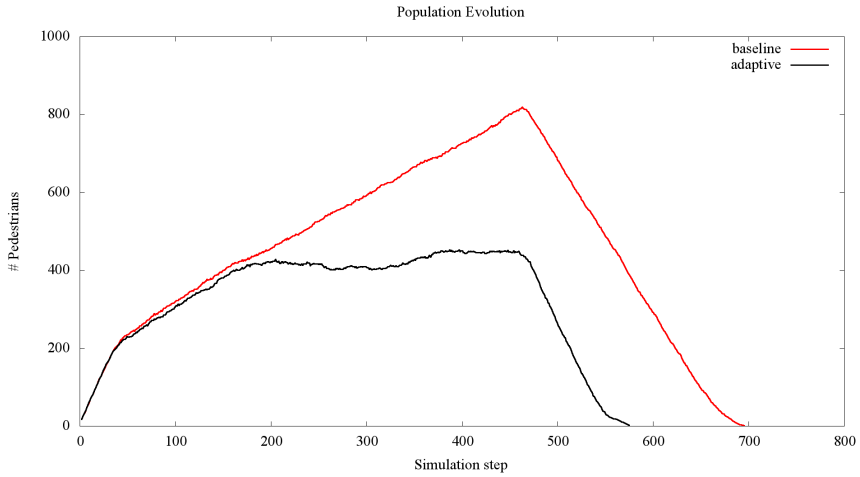
Fig. 6. The environment (a) and respective paths tree (b) used for the second experiment. The traveling times written in the paths tree nodes refer only to the normal agent class.

of pedestrians enter an area with six exits, distributed among 3 equal rooms, at a rate of 10 pedestrians per second. Only agents of the *normal* class will be used. An important peculiarity is the slightly asymmetrical configuration of the environment, that causes shorter distances towards the three southern exits. This is reflected by the illustrated paths tree in Fig. 6(b) where, to give an example, the paths starting from $o4$ and $o5$ and leading out through $o2$ take a little more time than the ones going out by using $o7$. As we will see, this slight asymmetry would significantly affect the results of the simulations but we will start by analysing and characterising the effect of the adaptive mechanism for path selection at tactical level, compared to a baseline shorter distance choice strategy. In particular, the results of this comparison concern the evolution of the number of pedestrians still in the scenario at a give time, shown in Figure 7(a), and the evolution of travel time for pedestrians entering the simulation environment at a given time, shown in Figure 7(b). The first diagram, highlights the fact that when pedestrian agents only choose their paths according to the covered distance, the so-called "least effort" principle, in this situation they actually employ much more time to vacate the area, since they do not exploit some trajectories that are sub-optimal from the point of view of the distance to be covered, but much more appealing since they represent "the road not taken", with very little congestion. Analogously, the evolution of the average travel time during the simulation shows that, in this situation, congestion has a much lower impact on adaptive agents with an awareness of the spatial representation of the environment, as long as the environment offers alternatives that are not subject to congestion.

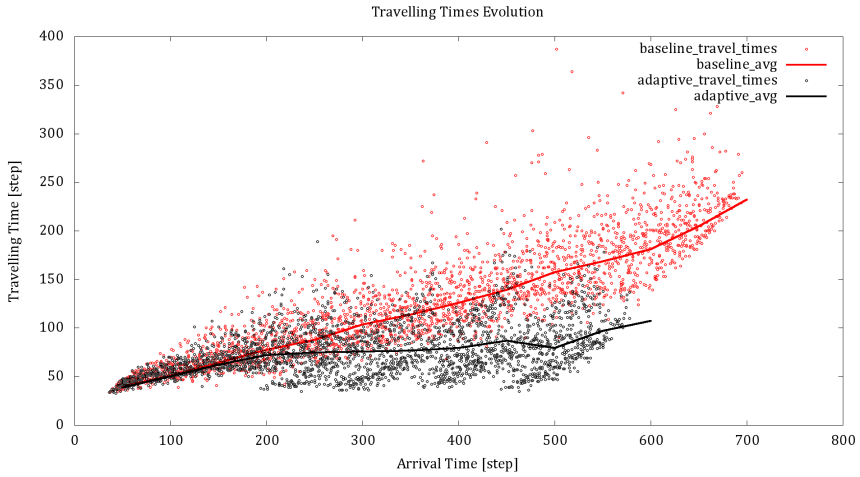
The above mentioned diagrams are surely important in quantitatively estimating the overall impact of this adaptive choice strategy, and they can represent a first way of validating it should actual data about the evacuation of a building presenting suitable characteristics were available. However, these metrics are too aggregated to actually illustrate the actual spatial utilisation of the environment by the pedestrians. Cumulative mean density maps [13][3] were also acquired to describe in a quali-quantitative way the behaviour of adaptive agents and they are shown in Fig. 8.

In particular, the diagrams show results of two simulations in which two different adaptive tactical level choice strategies have been adopted. First of all, it must be noted that all exits are used, although with a different frequency, whereas the static shortest path route choice would have selected just four exits. Moreover, northern and southern exits are generally chosen almost equally by the agents, despite the slight difference in the actual distance, due to the above introduce asymmetry. This is due to the fact that a random error of $\pm 10\%$ has been added to the overall calculation of the traveling time $Time(p)$ in order to consider the fact that pedestrians do not have an *exact* estimation of distances and delays caused by perceived congestion, in a more commonsense spatial reasoning framework [2]. The different adopted strategies consider a different triggering condition for the re-computation of

[3]These heat maps describe the mean local density value in each cell. It is calculated in a time window of 50 steps where, at each step, only values of occupied cells are collected.



(a)



(b)

Fig. 7. A diagram showing the evolution in time (step number) of the number of pedestrians still in the scenario (a) and a graph showing the evolution of travel time for pedestrians entering the simulation environment at a given time (b).

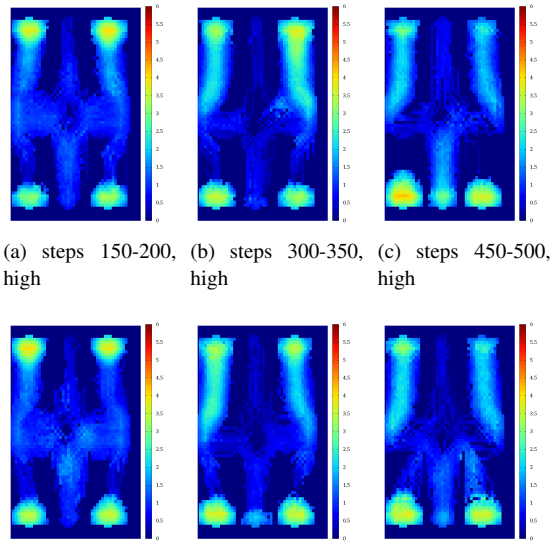
the travel times: agents, in fact, must reconsider their choices in the light of the perceived contextual conditions. For instance, an agent chooses an exit in the middle room because the closest one is congested only to realise, later on, that also the chose one is crowded. We decided to set a congestion threshold for not reconsidering an agent's choice: if the perceived congestion on the next step of plan does not overcome this threshold, the agent will continue with the planned course of actions. The first row of Figure 8 shows the evolution of the system with a relatively low threshold, and therefore a high sensitivity to congestion, and a lower sensitivity to congestion respectively in the top and bottom rows. The differences are relatively small, and they do not lead to significant changes in the average travel time, but we can see here that route adaptation is enacted a little later in case of low sensitivity to congestion, leading to a slightly higher congestion in the adopted exits.

As a concluding test, we investigated the increase in the computational times of the proposed approach, by simulating the 3-rooms environment above described with a population of 400 agents. In order to achieve comparable results, we configured two simulation scenarios: (i) a baseline scenario where the agents are not dynamically re-computing their plan, preserving the

one that they computed at their generation; (ii) a scenario where the agents are re-computing their plan at every step, in order to find a possible upper bound for the computational times derived by our approach. The simulations have been run in a desktop computer with a Intel Xeon processor at 2.53 GHz and 6 GB of RAM. The results are shown in Fig. 9. It is possible to see that the two series have a similar trend, meaning that the given approach is increasing the computational times in a linear way. At 400 agents, in the worst case scenario the computational times are increased of 20% from the baseline. Naturally, during a normal simulation the agent plan re-computation is not executed at every step (and the frequency of the operation is related to the sensitivity parameter discussed above), therefore in a standard simulation the overall execution times are expected to be quite closer to the baseline. In addition, it must be noted that no parallel implementations have been explored at the moment, and that they will be considered in the context of future works.

5. Conclusions

The paper has presented a hybrid agent architecture for modeling tactical level decisions, related to the choice of a route to follow in an environment comprising several rooms connected by openings, integrated with a validated operational level model, employing a floor-field based approach. The described model allows the agent taking decisions based on a static a-priori knowledge of the environment and dynamic perceivable information on the current level of crowdedness of visible path alternatives. The model was experimented in benchmark scenarios showing the adequacy in providing adaptiveness to the contextual situation. The future works, on one hand, are mainly aimed at defining requirements and an approach for the validation of the results achieved through the model: although we have an intuition that human agents adapt their behaviour to avoid heavily congested trajectories when obvious better alternatives are present (a phenomenology supported by the presented model), to which extent human choices are actually closer to optimality is object of ongoing interdisciplinary researches. This goal represents therefore an open challenge, since there are no comprehensive data sets on human tactical level decisions and automatic acquisition of this kind of data from video cameras is still a challenging task [23]. Even the mentioned [38], like most works on this topic, just explores the different alternatives that can be generated with distinct modeling choices, whereas a constrained form of validation was described in [1], although reported data are not sufficient for a quantitative cross validation of our approach. On the other hand, the addition of specific area actions (e.g. wait after reaching a certain point of interest indicated by a marker) and events (e.g. the arrival of a train) triggering agents' actions and, more generally, allowing the elaboration of more complicated agents' plans is also a planned extension on the side of model expressiveness.



(a) steps 150-200, high (b) steps 300-350, high (c) steps 450-500, high

(d) steps 150-200, low (e) steps 300-350, low (f) steps 450-500, low

Fig. 8. The test scenario respectively with a high and low sensitivity of the agents for the plan re-computation.

Acknowledgements

This work was partly supported by the ALIAS project ("Higher education and internationalization for the Ageing Society"), funded by Fondazione CARIPLO.

References

[1] M. Asano, T. Iryo, and M. Kuwahara. Microscopic pedestrian simulation model combined with a tactical model for route choice behaviour. *Transportation Research Part C*, 18(6):842 – 855, 2010.

[2] S. Bandini, A. Mosca, and M. Palmonari. Common-sense spatial reasoning for information correlation in pervasive computing. *Applied Artificial Intelligence*, 21(4&5):405–425, 2007.

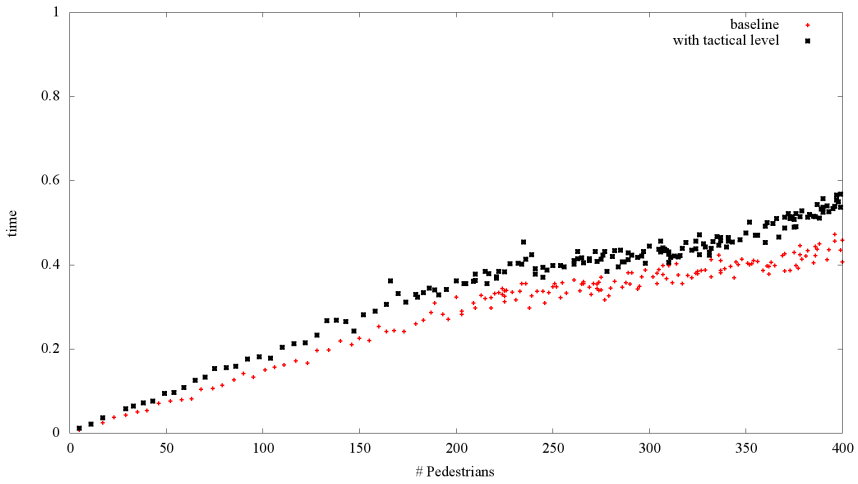


Fig. 9. Analysis of the computational times of the proposed approach against a baseline model with predefined tactical level decisions.

[3] S. Bandini, S. Manzoni, and G. Vizzari. Agent based modeling and simulation: An informatics perspective. *Journal of Artificial Societies and Social Simulation*, 12(4):4, 2009.

[4] S. Bandini, L. Crociani, and G. Vizzari. Heterogeneous speed profiles in discrete models for pedestrian simulation. In *Proceedings of the 93rd Transportation Research Board annual meeting*, 2014.

[5] S. Bandini, A. Gorrini, and G. Vizzari. Towards an integrated approach to crowd analysis and crowd synthesis: A case study and first results. *Pattern Recognition Letters*, 44:16–29, 2014.

[6] S. Bandini, L. Crociani, and G. Vizzari. Heterogeneous Pedestrian Walking Speed in Discrete Simulation Models. *Traffic and Granular Flow '13*, pages 273–279. Springer International Publishing, 2015.

[7] V. Blue and J. Adler. Modeling Four-Directional Pedestrian Flows. *Transportation Research Record: Journal of the Transportation Research Board*, 1710:20–27, 2000.

[8] V. J. Blue and J. L. Adler. Cellular Automata Microsimulation of Bi-Directional Pedestrian Flows. *Transportation Research Record*, 1678:135–141, 1999.

[9] M. Boltes and A. Seyfried. Collecting pedestrian trajectories. *Neurocomputing*, 100:127–133, 2013.

[10] M. Boltes, J. Zhang, A. Seyfried, and B. Steffen. T-junction: Experiments, trajectory collection, and analysis. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 158–165, 2011.

[11] T. Bosse, M. Hoogendoorn, M. C. A. Klein, J. Treur, C. N. van der Wal, and A. van Wissen. Modelling collective decision making in groups and crowds: Integrating social contagion and interacting emotions, beliefs and intentions. *Autonomous Agents and Multi-Agent Systems*, 27(1):52–84, 2013.

[12] C. Burstedde, K. Klauck, A. Schadschneider, and J. Zittartz. Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A*, 295(3 - 4):507–525, 2001. ISSN 0378-4371.

[13] C. J. E. Castle, N. P. Waterson, E. Pellissier, and S. Bail. A Comparison of Grid-based and Continuous Space Pedestrian Modelling Software: Analysis of Two UK Train Stations. In *Pedestrian and Evacuation Dynamics*, pages 433–446. Springer, 2011.

[14] M. Chraibi, A. Seyfried, and A. Schadschneider. Generalized centrifugal-force model for pedestrian dynamics. *Phys. Rev. E*, 82(4):46111, 2010.

[15] L. Crociani, A. Invernizzi, and G. Vizzari. A hybrid agent architecture for enabling tactical level decisions in floor field approaches. *Transportation Research Procedia*, 2:618–623, 2014.

[16] R. Geraerts. Planning short paths with clearance using explicit corridors. In *2010 IEEE International Conference on Robotics and Automation*, pages 1997–2004. IEEE, 2010.

[17] R. Geraerts and M. Overmars. The Corridor Map Method: A General Framework for Real-Time High-Quality Path Planning. *Computer Animation and Virtual Worlds*, 18:107–119, 2007.

[18] R.-Y. Guo and H.-J. Huang. Route choice in pedestrian evacuation: formulated using a potential field. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(04):P04018, 2011.

[19] D. Helbing. A fluid dynamic model for the movement of pedestrians. *Complex Systems*, 6:391–415, 1992.

[20] D. Helbing and P. Molnar. Social force model for pedestrian dynamics. *Phys. Rev. E*, 51(5):4282, 1995.

[21] L. F. Henderson. The Statistics of Crowd Fluids. *Nature*, 229 (5284):381–383, 1971.

[22] J. A. Hendler. Where are all the intelligent agents? *IEEE Intelligent Systems*, 22(3):2–3, 2007.

[23] S. D. Khan, G. Vizzari, and S. Bandini. Identifying sources and sinks and detecting dominant motion patterns in crowds. *Transportation Research Procedia*, 2195 – 200, 2014.

[24] S. D. Khan, G. Vizzari, S. Bandini, and S. Basalamah. Detecting dominant motion flows and people counting in high density crowds. *Journal of WSCG*, 22(1-2):21–30, 2014.

[25] A. Kirchner, H. Klüpfel, K. Nishinari, A. Schadschneider, and M. Schreckenberg. Discretization effects and the influence of walking speed in cellular automata models for pedestrian dynamics. *Journal of Statistical Mechanics*, 2004(10):P10011, 2004.

[26] T. Kretz, C. Bönisch, and P. Vortisch. Comparison of Various Methods for the Calculation of the Distance Potential Field. In *Pedestrian and Evacuation Dynamics 2008*, pages 335–346. Springer Berlin Heidelberg, 2010.

[27] M. Luck, P. McBurney, O. Sheory, and S. Willmott, editors. *Agent Technology: Computing as Interaction*. University of Southampton, 2005.

[28] P. McBurney and M. Luck. The agents are all busy doing stuff! *IEEE Intelligent Systems*, 22(4):6–7, 2007.

[29] J. A. Michon. A Critical View of Driver Behavior Models: What Do We Know, What Should We Do? In *Human Behavior and Traffic Safety*, pages 485–524. Springer US, 1985.

[30] D. Pianini, M. Viroli, F. Zambonelli, and A. Ferscha. HPC from a self-organisation perspective: The case of crowd steering at the urban scale. In *International Conference on High Performance Computing & Simulation, HPCS 2014, Bologna, Italy, 21-25 July, 2014*, pages 460–467. IEEE, 2014.

[31] F. Qiu and X. Hu. Modeling group structures in pedestrian crowd simulation. *Simulation Modelling Practice and Theory*, 18(2):190 – 205, 2010.

[32] A. Schadschneider, W. Klingsch, H. Klüpfel, T. Kretz, C. Rogsch, and A. Seyfried. Evacuation Dynamics: Empirical Results, Modeling and Applications. In *Encyclopedia of Complexity and Systems Science*, pages 3142–3176. Springer, 2009.

[33] F. Solera and S. Calderara. Social groups detection in crowd through shape-augmented structured learning. In *Image Analysis and Processing - ICIAP 2013 - 17th International Conference, Naples, Italy, September 9-13, 2013. Proceedings, Part I*, volume 8156 of *Lecture Notes in Computer Science*, pages 542–551. Springer, 2013.

[34] Y. Suma, D. Yanagisawa, and K. Nishinari. Anticipation effect in pedestrian dynamics: Modeling and experiments. *Physica A*, 391(1-2):248–263, 2012.

[35] G. Vizzari and S. Bandini. Studying pedestrian and crowd dynamics through integrated analysis and synthesis. *IEEE Intelligent Systems*, 28(5):56–60, 2013.

[36] G. Vizzari, L. Manenti, and L. Crociani. Adaptive Pedestrian Behaviour for the Preservation of Group Cohesion. *Complex Adaptive Systems Modeling*, 1(7), 2013.

[37] I. von Sivers and G. Köster. Dynamic Stride Length Adaptation According to Utility And Personal Space. *Transportation Research Part B*, 74:104–117, 2015.

[38] A. U. K. Wagoum, A. Seyfried, and S. Holl. Modelling dynamic route choice of pedestrians to assess the criticality of building evacuation. *Advances in Complex Systems*, 15(07):15, 2012.

[39] J. Zhang, W. Klingsch, A. Schadschneider, A. Seyfried. Transitions in pedestrian fundamental diagrams of straight corridors and T-junctions. *Journal of Statistical Mechanics*, 2011 (6):P06004, 2011.

[40] J. Zhang, W. Klingsch, A. Schadschneider, and A. Seyfried. Ordering in bidirectional pedestrian flows and its influence on the fundamental diagram. *Journal of Statistical Mechanics*, 2012(02):P02002, 2012.