Adaptive Hybrid Agents for Tactical Decisions in Pedestrian Environments

Luca Crociani, Andrea Piazzoni, Giuseppe Vizzari
CSAI - Complex Systems & Artificial Intelligence Research Center,
University of Milano-Bicocca, Milano, Italy
{luca.crociani,giuseppe.vizzari}@disco.unimib.it
a.piazzoni@campus.unimib.it

***Abstract*—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 mean of a floor-field based model, in a discrete simulation 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 paper presents the model formally, motivating the adopted choices with reference to the relevant state of the art. The model is also experimented in benchmark scenarios showing the adequacy in providing adaptiveness to the contextual situation.**

***Index Terms*—Pedestrian Simulation, Tactical Level, Hybrid Agents.**

I. Introduction

The simulation of complex systems is one of the most successful areas of application of agent-based approaches: even though models, mechanisms and technologies adopted by researchers in different disciplines are not necessarily up-to-date or in line with the most current results in the computer science and engineering area about agent technologies [1], the area still presents interesting challenges and potential developments for agent research.

The simulation of pedestrians and crowds is an example of already established yet 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 [2], present open challenges and potential developments in a smart environment perspective [3].

Even if we only consider choices and actions related to walking, modelling human decision making activities and actions is a complicated issue: 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 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., [4]), 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 mean 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 III-B whereas its experimental application in benchmark scenarios showing the adequacy in providing adaptiveness to the contextual situation is given in Section IV.

II. 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. [5], [4]), charactertization of pedestrian flows (e.g. [6]) automatic recognition of pedestrian groups [7], recently gaining importance due to differences in trajectories, walking speeds and space utilization [8]. The *synthesis* side – where the contributions of this work are concentrated – has been even more prolific, starting from preliminary studies and assumptions provided by [9] or [10] 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.

[12][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 perform 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, [14], [15]) 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, basing on analogies between behavior of dense crowds and kinetic gas [9] or fluids [10], 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 [16]. 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 on this class are the centrifugal force model by [17] and the stride length adaptation model by [18]. Other examples consider also groups of pedestrians by means of attractive forces among person inside the group [19].

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. [20], [21]). Other models employs the well-know *floor field* approach from [22], 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. [23] is an extension of the floor field model, introducing the *anticipation* floor field used to manage crossing trajectories and encourage the lane formation. [24] 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 [25]. [26] 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. [27], [28]), but with aims not necessarily matching the requirements of pedestrian simulation, since the point is mainly to reach a visual realism.

On the pedestrian dynamics side, a relevant recent work is the one from [29], mainly dealing with tactical level decisions during evacuation, providing an approach to find the *quickest* path towards the exit, i.e. the way that implies the fewest time. The approach is tested on the basis of four strategies for the route choice management, given by the combination of applying the shortest or quickest path, with a local (i.e., to go out from the room) or global (i.e., to reach the destination) strategy. The global shortest path is calculated with the well-known Floyd-Warshall algorithm, implying therefore a computational time that can become an issue by having a large number of nodes or by considering special features in the simulated population (i.e. portion of the path where the cost differs from an agent to another). The work in this paper will propose 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 costs to the computation.

III. A Model for Tactical Level of Pedestrians

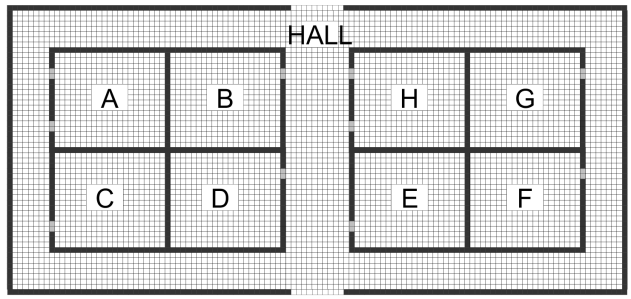
The model described in this work provides a methodology to deal with tactical choices of agents in pedestrian simulation systems. Due to constraints on the length of the paper, the description of the part of the model dedicated to the operational level, thoroughly described in [30], will not be provided.

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

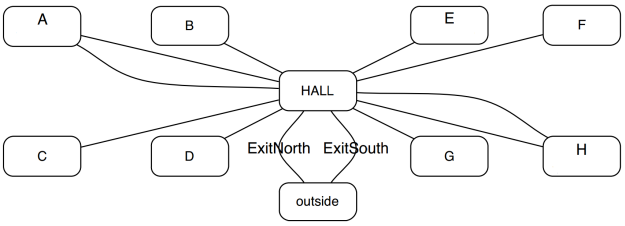
The framework that enables agents to perform choices on their plan implies a graph-like, topological, representation of the walkable space, whose calculation is defined in [31] and briefly reported in this section. This model allows agents to perform a static path planning, since dynamical information 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 and represent the innovative part of this paper.

The environment abstraction identifies *regions* (e.g. a room) as node 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

[1] A similar classification can be found in vehicular traffic modeling from [13].



(a)



(b)

Fig. 1. An example environment (a) with the resulting cognitive map (b), by applying the procedure from [31].

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. [32]), 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 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 to 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.

B. Modeling Adaptive Tactical Decisions with A Paths Tree

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

1) The Paths Tree: We define the *Paths Tree* as a tree data-structure containing the set of "rational" paths towards a destination, that will be its root. Before describing what we mean with the attribute rational, 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 III.1 (Oriented opening)**.** Let $E = R_1, R_2$ be a 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.

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

Definition III.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 III.3 (Start and Destination of a path)**.** Given p a path $(R_1, E, R_x) \rightarrow ... \rightarrow (R_y, O, \text{universe})$, the function $RS(p) = R_1$ will return the region R_1 where an agent can start the path p. $S(p) = E$ and $D(p) = p$ will respectively return the first opening (E) and the destination (O) of the path.

Definition III.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 rational.

Definition III.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 rational paths by extending III.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 by the 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 rational, even if the constraint III.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 III.6 (Sub-path)**.** Let p 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.

The reason of the orientation change can be explained with the example in Fig. 3: given the path $p = (r1, o2, r2) \rightarrow (r2, o1, r1) \rightarrow end$, the path $p' = (r2, o2, r1) \rightarrow end$ is a valid path and is considered as a sub-path of p, with

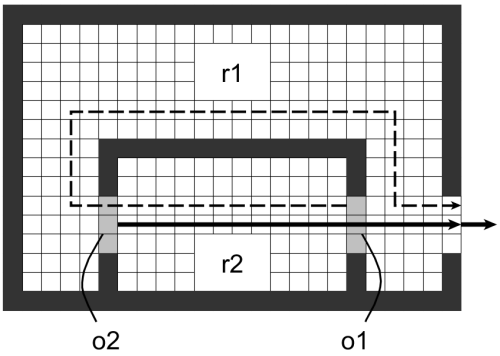


Fig. 2. A concave region can imply the plausibility of a path crossing it twice, but its identification is not elementary.

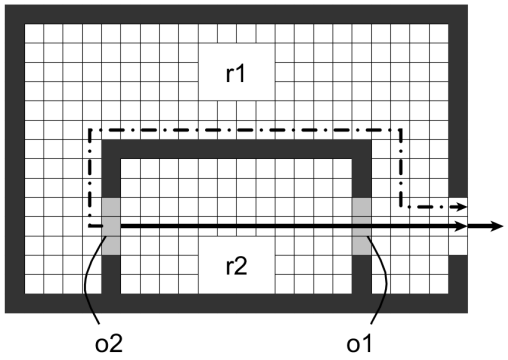


Fig. 3. The correct paths for this environment. Inside $r2$ the choice between the two openings is also determined by the congestion.

a different orientation of $o2$. In addition, given the path $p_1 = (r_2, o_2, r_1) \rightarrow end$, the path $p_2 = (r_1, o_2, r_2) \rightarrow (r_2, o_1, r_1) \rightarrow end$ 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 of the opening considering the congestions. An agent located in $r1$ can go directly to the exit or chose the path $o2 \rightarrow o1 \rightarrow D$.

Definition III.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 III.5 and solve 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 III.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 region (edges). Each node has an attribute describing the expected travel time to the destination.

2) An Algorithm to Compute the Paths Tree: The algorithm we are proposing build 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.

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 III.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** $CheckMinimality(ShortestPath, o, \tau)$ == 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**

At this point, the minimality constraint III.7 has to be verified for each candidate, by means of the function $CheckMinimality$ 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 CheckMinimality

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

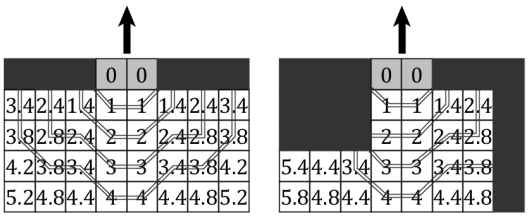


Fig. 4. Examples of surroundings of different sizes, for two configurations of the environment.

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 of the branch. Thus, if τ is less than $ShortestPath[o]$, the minimality constraint III.7 is respected.

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) increase it when the average speed becomes too low or (ii) 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 will be explained in the next section. Two function 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.

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.

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

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

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

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.

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

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.

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

Where:

- τ_p : the expected travel time of the path p

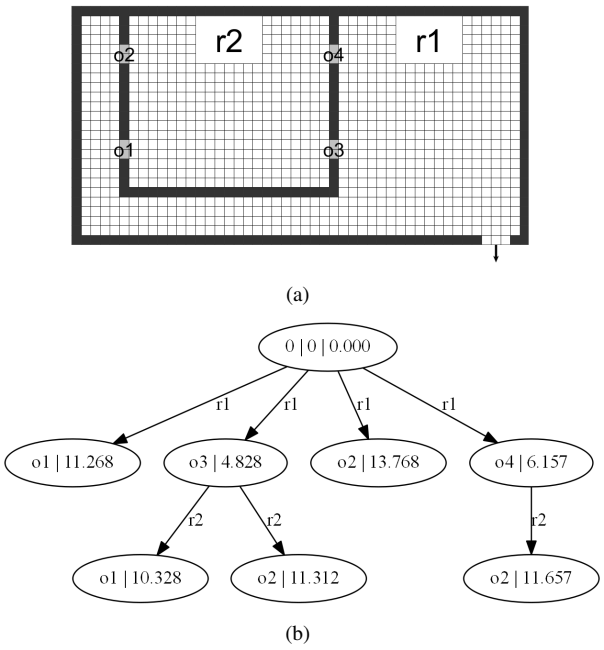


Fig. 5. Example scenario with the respective paths tree.

- $\frac{d(a,S(p)}{speed_a}$: 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).

IV. Applications in Example Scenarios

In order to show the reliability and potentials of the proposed approach, results of two example scenarios will be presented in this section.

The aim of the first experimental scenario is to show the output of the *paths tree* generation algorithm. Figure 5 shows the environment and the associated data structure. The tree contains two information on each node, describing the Id of the mapped opening and the estimated time associated to its path. In addition, each edge is labeled with the Id of the region crossed by the path. The result shows that the possible minimal paths have been represented in the tree. In particular, child nodes of $o1$ and $o2$, passing inside $r2$, have not been added since that would imply a path passing from $o3$ or $o4$ and describing a not rational way through $r2$ and the corridor at the bottom of $r1$. Paths like $p = o2 \rightarrow o4 \rightarrow end$ or $p = o2 \rightarrow o4 \rightarrow end$ have been considered instead, since they could be plausible for pedestrians being in the top left corner of the scenario.

Results of the second experiment are shown in Figure 6: the illustrated environment have been populated with 200 agents, generated in the *Start* object with an arrival frequency of around 7 pers/sec. The heat maps contain the *cumulative mean densities* of pedestrians, describing a cumulative value of density in each cell for a fixed time window (in this case the duration is 50 steps, equal to 12.5 seconds). It must be noted that at each step the values are accumulated only in pedestrians

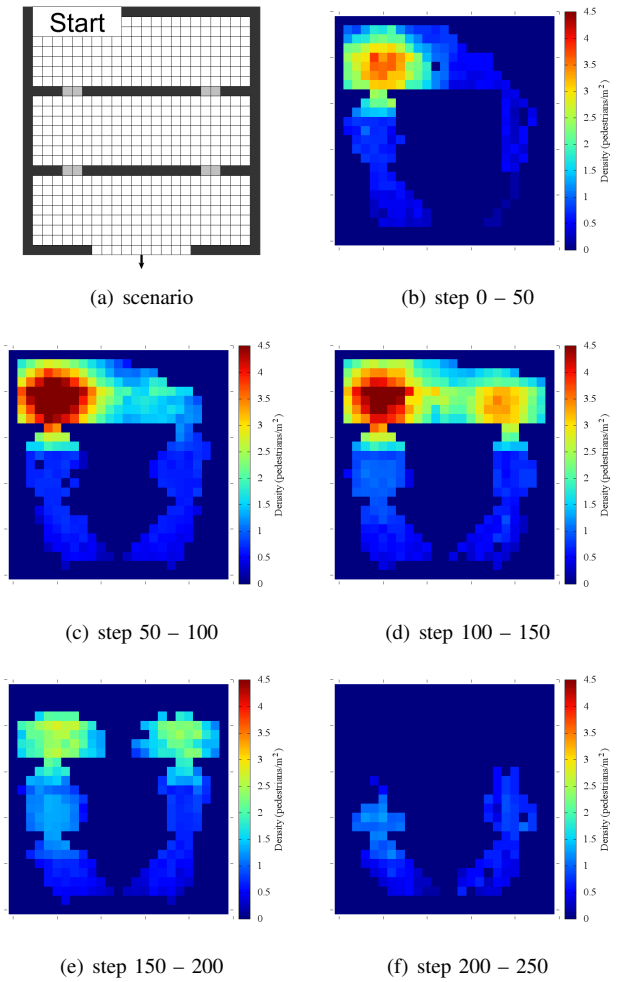


Fig. 6. The test scenario (a) with cumulative mean density maps of the simulation for various time windows (b – f).

positions, in order to give an idea of what pedestrians have perceived during the simulation. The stream of pictures shows that the arrival rate cannot be sustained from the opening at the top left of the environment, describing the shortest path towards the exit, thus a growing congestion arises in front of it. This increases the time perceived by the agents to employ the shortest path, leading a significant part of them to change their route preferring the other opening, that will get also congested in the third time window. The arrival rate stops at about step 150 and from this moment the environment starts to get empty.

V. 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: this represents 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 [33]. Even the mentioned [29], 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 [34], 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.

REFERENCES

[1] S. Bandini, S. Manzoni, and G. Vizzari, "Agent based modeling and simulation: An informatics perspective," *Journal of Artificial Societies and Social Simulation*, vol. 12, no. 4, p. 4, 2009.

[2] G. Vizzari and S. Bandini, "Studying pedestrian and crowd dynamics through integrated analysis and synthesis," *IEEE Intelligent Systems*, vol. 28, no. 5, pp. 56–60, 2013.

[3] 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*. IEEE, 2014, pp. 460–467.

[4] M. Boltes and A. Seyfried, "Collecting pedestrian trajectories," *Neurocomputing*, vol. 100, pp. 127–133, Jan. 2013.

[5] 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*, Nov. 2011, pp. 158–165.

[6] S. D. Khan, G. Vizzari, S. Bandini, and S. Basalamah, "Detecting dominant motion flows and people counting in high density crowds," *Journal of WSCG*, vol. 22, no. 1-2, pp. 21–30, 2014.

[7] 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*, ser. Lecture Notes in Computer Science, A. Petrosino, Ed., vol. 8156. Springer, 2013, pp. 542–551.

[8] 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*, vol. 44, pp. 16–29, 2014.

[9] L. F. Henderson, "The Statistics of Crowd Fluids," *Nature*, vol. 229, no. 5284, pp. 381–383, Feb. 1971.

[10] D. Helbing, "A fluid dynamic model for the movement of pedestrians," *arXiv preprint cond-mat/9805213*, 1998.

[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*, vol. 27, no. 1, pp. 52–84, 2013.

[12] 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*, R. A. Meyers, Ed. Springer, 2009, pp. 3142–3176.

[13] J. A. Michon, "A Critical View of Driver Behavior Models: What Do We Know, What Should We Do?" in *Human Behavior and Traffic Safety*, L. and Evans and R. C. Schwing, Eds. Springer US, 1985, pp. 485–524.

[14] J. Zhang, W. Klingsch, T. Rupprecht, A. Schadschneider, and A. Seyfried, "Empirical study of turning and merging of pedestrian streams in T-junction," *ArXiv e-prints*, p. 8, 2011.

[15] 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: Theory and Experiment*, no. 02, p. 9, 2011.

[16] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, 1995.

[17] M. Chraibi, A. Seyfried, and A. Schadschneider, "Generalized centrifugal-force model for pedestrian dynamics," *Phys. Rev. E*, vol. 82, no. 4, p. 46111, 2010.

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

[19] F. Qiu and X. Hu, "Modeling group structures in pedestrian crowd simulation," *Simulation Modelling Practice and Theory*, vol. 18, no. 2, pp. 190 – 205, 2010.

[20] V. Blue and J. Adler, "Modeling Four-Directional Pedestrian Flows," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1710, no. -1, pp. 20–27, Jan. 2000.

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

[22] C. Burstedde, K. Klauck, A. Schadschneider, and J. Zittartz, "Simulation of pedestrian dynamics using a two-dimensional cellular automaton," *Physica A: Statistical Mechanics and its Applications*, vol. 295, no. 3 - 4, pp. 507–525, 2001.

[23] Y. Suma, D. Yanagisawa, and K. Nishinari, "Anticipation effect in pedestrian dynamics: Modeling and experiments," *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 1-2, pp. 248–263, Jan. 2012.

[24] 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: Theory and Experiment*, vol. 2004, no. 10, p. P10011, 2004.

[25] S. Bandini, L. Crociani, and G. Vizzari, "Heterogeneous Pedestrian Walking Speed in Discrete Simulation Models," in *Traffic and Granular Flow '13*, M. Chraibi, M. Boltes, A. Schadschneider, and A. Seyfried, Eds. Springer International Publishing, 2015, pp. 273–279.

[26] G. Vizzari, L. Manenti, and L. Crociani, "Adaptive Pedestrian Behaviour for the Preservation of Group Cohesion," *Complex Adaptive Systems Modeling*, vol. 1, no. 7, 2013.

[27] R. Geraerts and M. Overmars, "The Corridor Map Method: A General Framework for Real-Time High-Quality Path Planning," pp. 107–119, 2007.

[28] R. Geraerts, "Planning short paths with clearance using explicit corridors," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, May 2010, pp. 1997–2004.

[29] 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*, vol. 15, no. 07, p. 15, 2012.

[30] 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.

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

[32] 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*, W. W. F. Klingsch, C. Rogsch, A. Schadschneider, and M. Schreckenberg, Eds. Springer Berlin Heidelberg, 2010, pp. 335–346.

[33] S. D. Khan, G. Vizzari, and S. Bandini, "Identifying sources and sinks and detecting dominant motion patterns in crowds," *Transportation Research Procedia*, vol. 2, no. 0, pp. 195 – 200, 2014, the Conference on Pedestrian and Evacuation Dynamics 2014 (PED 2014), 22-24 October 2014, Delft, The Netherlands.

[34] M. Asano, T. Iryo, and M. Kuwahara, "Microscopic pedestrian simulation model combined with a tactical model for route choice behaviour," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 6, pp. 842 – 855, 2010, special issue on Transportation Simulation Advances in Air Transportation Research.