An Intelligent Tool for the Automated Evaluation of Pedestrian Simulation

Evangelos Boukas[1], Luca Crociani[2], Sara Manzoni[2], Giuseppe Vizzari[2], Antonios Gasteratos[1], and Georgios Ch. Sirakoulis[1]

[1] Democritus University of Thrace, Xanthi, Greece
[2] University of Milano-Bicocca, Milan, Italy
evanbouk@pme.duth.gr,
{luca.crociani,manzoni,viz}@disco.unimib.it,
agaster@pme.duth.gr, gsirak@ee.duth.gr

Abstract. One of the most cumbersome tasks in the implementation of an accurate pedestrian model is the calibration and fine tuning based on real life experimental data. Traditionally, this procedure employs the manual extraction of information about the position and locomotion of pedestrians in multiple videos. The paper in hand proposes an automated tool for the evaluation of pedestrian models. It employees state of the art techniques for the automated 3D reconstruction, pedestrian detection and data analysis. The proposed method constitutes a complete system which, given a video stream, automatically determines both the workspace and the initial state of the simulation. Moreover, the system is able to track the evolution of the movement of pedestrians. The evaluation of the quality of the pedestrian model is performed via automatic extraction of critical information from both real and simulated data.

Keywords: pedestrian simulation, pedestrian detection, cellular automata, stereo vision, 3D reconstruction, agent based models.

1 Introduction

The pedestrian modeling has been studied the past decades extensively and different approaches have been followed, which can be classified mainly as *force*-based [1,2], *CA*-based [3,4] and *agent*-based models [5,6]. Yet, the research community has not reached the state of understanding or development that would allow the accurate modeling and simulation of the widest variety of pedestrian movement scenarios. The valorisation of pedestrian modeling and simulation techniques is mostly carried out by assessing their ability to assemble the evolution real life pedestrian movement circumstances.

The collection of crucial and meaningful data constitutes a challenge by itself. Usually the automated extraction of data from videos is employed on controlled groups of pedestrians, which are commonly equipped with wearable markers. However, during real life scenarios such markers do not exist and, therefore, such approaches are unfeasible and manual extraction is required. For example, works in [7,8,9] employ manual extraction of experimental data from videos. The collection and process of this data is a dull, repetitive, tiring and error-prone task, thus making it a perfect candidate for automation. This proposed work aspires to fill this gap, while the approach followed is analyzed in the next sections. The approach of our method consists of three distinct modules, namely the "Computer Vision module", the "Pedestrian Simulation module" and the "Evaluation module" (Fig. 1). The "Computer Vision module" includes all the software and hardware required to capture and analyze the real world environment, including the 3D formation of the environment and the pedestrians position and specific attributes such as speed and direction. The "Pedestrian Simulation module" receives as input the information about the starting scenario including the pedestrian attributes and then simulates the evolution of the movement. Finally the output of the simulation is contrasted with the real evolution of the pedestrian movement to appraise the quality of the simulation.

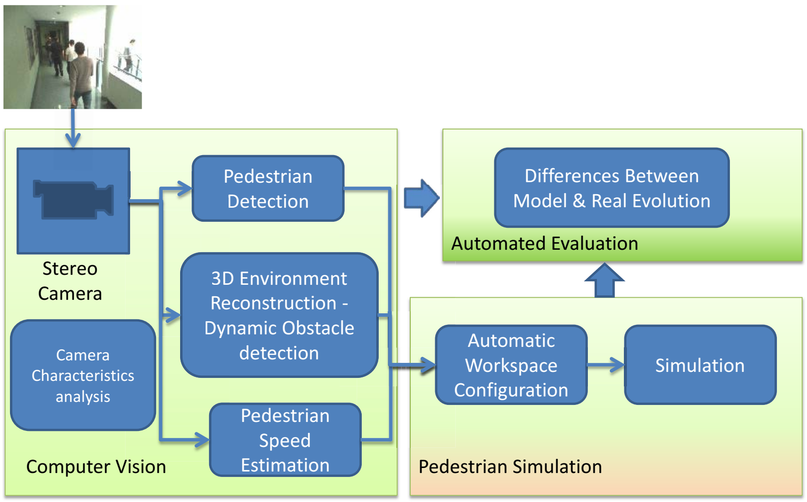
Stereo Camera
Pedestrian Detection
3D Environment Reconstruction - Dynamic Obstacle detection
Camera Characteristics analysis
Pedestrian Speed Estimation
Computer Vision
Differences Between Model & Real Evolution
Automated Evaluation
Automatic Workspace Configuration
Simulation
Pedestrian Simulation

Fig. 1. The Approach Schematic includes the three major modules of the system: "Computer Vision", "Pedestrian Simulation" and "Evaluation"

2 Computer Vision Module

In this section the computer vision system is outlined. Firstly, we present the camera setup, then the 3D reconstruction process is analyzed, followed by the pedestrian position and attributes extraction, finally we provide the transition from the 3D world to the 2D simulation scenario.

2.1 Camera Setup

In order to accurately capture the formation and attributes of the surrounding environment, including depth information, we must firstly design our computer vision system in terms of hardware, i.e. the camera devices that are going to be employed [10]. In order to achieve the maximum accuracy, at the required range as depicted in Fig. 2, we need to consider a special stereo camera setup and to define the specific hardware attributes [11]. Firstly, the range resolution is the minimal change in range that the stereo vision system can differentiate. In general, resolution deteriorates with distance. The function that calculates the range l within which the resolution r is better than, or equal to a desired value is the following:

$$l = \sqrt{\frac{0.5 \cdot r \cdot b \cdot w}{c \cdot tan(0.5 \cdot F)}} \tag{1}$$

where l is the distance in which the desired resolution is achieved, r is the specified range resolution, b is the baseline, w is the horizontal image resolution, F is the cameras' field of view (FoV) expressed in radians, and c is the disparity precision expressed in pixels. In particular, the disparity precision concerns the sub-pixel resolution during the calculation of the disparity map, obtained by interpolation. Eq. 1 shows that given the resolution r the range l may grow either by increasing the baseline b or by decreasing F, or both. Besides, more accurate stereo results are typically obtained by keeping the stereo angle (the angle between the line-of-sight of the two cameras to the minimum-distance object) as low as possible and in all cases below 15^o, due to the smaller correspondence search range [12]. Moreover, the function that relates the focal length f of the cameras with the field of view F is of important for the determination of a stereo system's parameters, and can be expressed as:

$$f = \frac{0.5 \cdot s \cdot 0.001 \cdot w}{tan(0.5 \cdot F)} \tag{2}$$

where s is the physical width of the sensor's pixels.

In order to overcome the step of the design and implementation of a special stereoscopic camera system, one could implement a vision system occupying a RGB-D sensor, such as the Microsoft Kinect or the Asus Xtion, that is able to capture both visual and depth information. The main disadvantage of fixed Commercial Off-The-Shelf (COTS) RGB-D solutions is that the accuracy drops

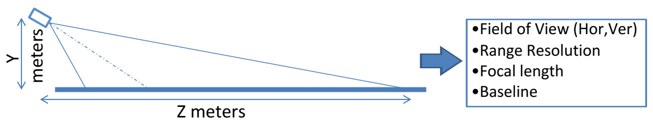

Fig. 2. The camera setup

significantly further than 5 *meters*. Specifically, the accuracy drops to less than 20 *centimeters* at a distance that is farther than 8 *meters*. An indicative example of the output of our vision system, which has been installed in a narrow corridor at the premisses of the Department of Informatics, Systems and Communication (DISCo) of the University of Milano-Bicocca, is presented in Fig. 3.

2.2 3D Scene Reconstruction

The next step comprises the 3D reconstruction of the scene. The latter is a straightforward procedure given the intrinsic and the extrinsic parameters of the utilized stereo rig. Making use of the depth information calculated in the disparity module, the position of each pixel onto the image plane are then expressed into 3D world coordinates. More specifically, pixels expressed in camera coordinates $(x_c, y_c, disp(x_c, y_c))$, with respect to the stereo geometry, are transformed in 3D points (x, y, z). The XY plane coincides with the image plane while the Z axis denotes the depth of the scene [13]. The relation between the world coordinates of a point $P(x, y, z)$ and the coordinates on the image plane $(x_c, y_c, disp(x, y))$ is expressed by the pin-hole model and the stereo setup as:

$$[x, y, z] = \left[\frac{x_c \cdot z}{f}, \frac{y_c \cdot z}{f}, \frac{b \cdot f}{disp(x_c, y_c)}\right] \tag{3}$$

where, z is the depth value of a pixel depicted in (x_c, y_c), b is the stereo camera's baseline, f the focal length of the lenses expressed in pixels and $disp(x_c, y_c)$ the corresponding pixel's disparity value. In Eq. 3 x and y denote the abscissa and the ordinate in 3D world coordinates, respectively, which as a pair correspond to the (x_c, y_c) pixel on the image plane, respectively. In the case of the usage of an RGB-D sensor the disparity is obtained after a transformation of the depth image, since the disparity and the depth image are inversely proportional.

2.3 Traversable/Obstacle Free Area Extraction

In the next step, the area is partitioned into traversable and not-traversable one [14]. Using disparity map, a reliable v-disparity image is computed, as shown in Fig. 3. In a v-disparity image each pixel value corresponds to the number of pixels in the input image that lie on the same image line (ordinate) and posses disparity value equal to its abscissa. The terrain in the v-disparity image is modeled by a linear equation, the parameters of which can be found using Hough transform [15], condition to the fact that the a significant number of the input images' pixels belong to the terrain and not to obstacles. A tolerance region on both sides of the terrain's linear segment is considered and any point outside this region can be safely considered as originating from a barrier. The linear segments denoting the terrain and the tolerance region overlaid on the v-disparity image are shown in Fig. 3. Then pixels of the image that lie in the traversable area can be traced.

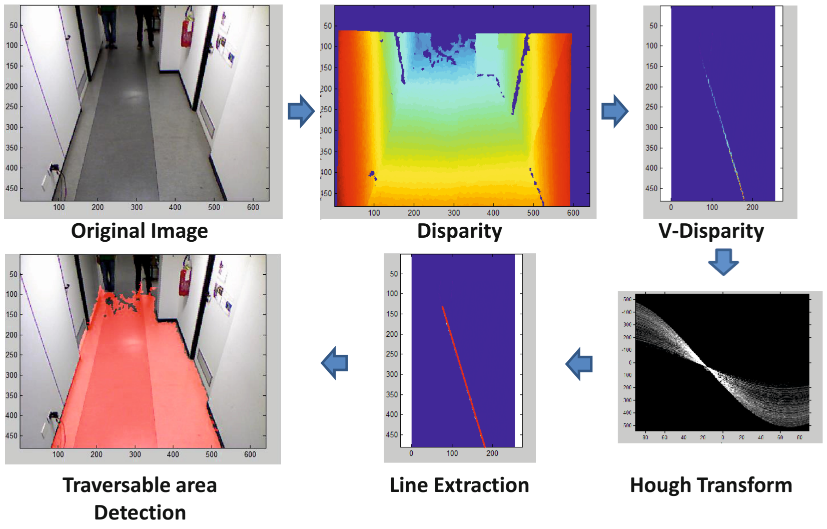

Fig. 3. The visual representation of the implemented algorithm for the computation of the floor

2.4 2D Simulation Scenario

The pedestrian simulation algorithm employees a cellular automaton (CA) that operates over a grid. Thus, the 3D reconstructed area should be transformed into a 2D grid. Based on the aforementioned procedure of the extraction of the obstacles in an area, each point of the point cloud that lays on an obstacle is projected on the floor plane, which has also been estimated by the v-disparity. Then, taking into consideration the the size of the CA cells, which may vary depending on the evaluated model (in our case it is $40cm \times 40cm$), the projected points are sampled on the cellular grid. Figure 4 depicts the steps required for this transformation. The resulting simulation scenario (workspace) is ready to be infused with virtual pedestrians. The following subsection describes the process of pedestrian position and attributes extraction.

2.5 Pedestrian Position and Velocity Extraction

Identifying moving objects in video sequence is a fundamental and critical task in video surveillance and, thus, it has been extensively studied in the past [16],[17],[18]. For the shake of executional acceleration and based on the fact that an indoors and almost "controlled" environment is assumed, a rather different approach is employed. The technique is based on the 3D perception of the environment which is already implemented in the previews parts of our system. The system can be partitioned to three major components:

- In-point-cloud analysis to extract pedestrian candidates
- Mean Shift clustering to count and locate pedestrians
- SIFT feature matching to track the pedestrians

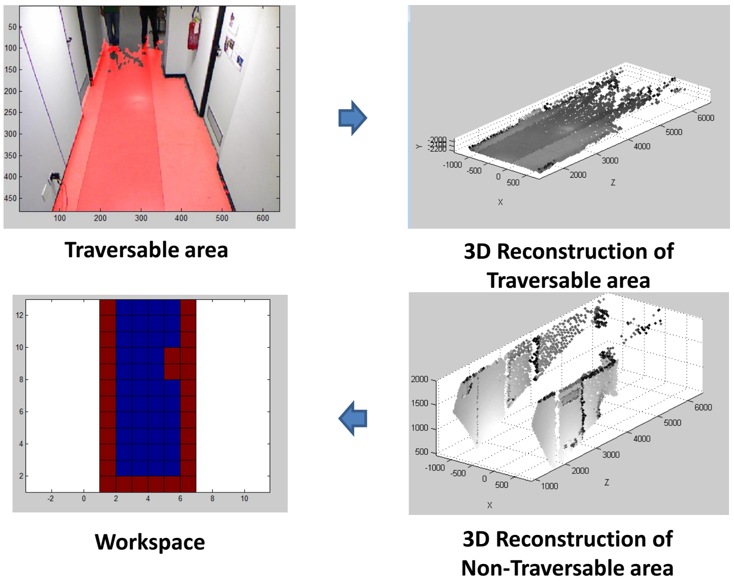
Traversable area
3D Reconstruction of Traversable area
Workspace
3D Reconstruction of Non-Traversable area

Fig. 4. Discrete steps of the algorithm for the automated formation of the workspace

Firstly, the extraction of those points, in the point cloud, that possibly correspond to pedestrians should be performed. This method is based on the fact that the empty observed environment has been classified to ground (traversable area) and to obstacles. A geometric analysis is performed directly on the point cloud, in order to extract those points that are above the floor and do not belong to an obstacle. This search is automatically windowed in an area that is defined from the 3D obstacles.

The number of pedestrians, which appear at each frame is unknown. Thus a clustering algorithm, namely the mean shift [19], is employed in order to segment the point cloud into pedestrians' sub-point clouds. The mean shift provides the ability to define the shape of the 3D clusters by adjusting the bandwidth. The bandwidth of our setup is set to 0.5 m. This selection is physically consistent with the size of pedestrians. An example of the extracted resulting clusters are presented in Fig.5. The next step comprises the tracking of the pedestrians, performed by employing the SIFT features. At each frame, SIFT features are extracted, detected and matched to the next frame's features. Next, they are tracked throughout the 3D reconstruction and pedestrian detection thusly leading to the tracking of the pedestrians. An example of pedestrian tracking through consecutive frames is presented in Fig.6.

The tracking of pedestrians provides the ability to extract an additional attribute, that is the personal velocity. The velocity of a point, that is matched in two consecutive frames, is calculated as the 3D Euclidean distance of its position in the respective reconstructed point clouds, divided by the time between these

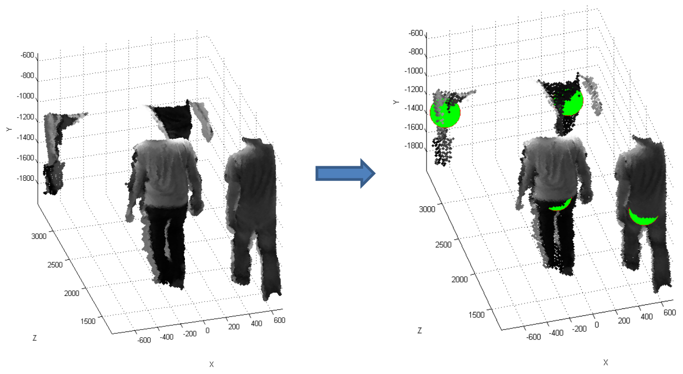

Fig. 5. The detection of the pedestrians exploiting the mean shift algorithm

frames. This time is given by the frame rate of the camera system. The velocity of the pedestrian can be calculated as the mean velocity of all the matched points of the same cluster.

3 Pedestrian Simulation Model

Having created the initial simulation scenario infused with the pedestrians we can simulate their movement and then validate the outputs of the simulator with the observed data. In this section the computational model used for pedestrian simulation will be briefly described[1], in order to understand the kinds of evaluation which can be performed.

3.1 Environment

The environment is modeled in a discrete way by representing it as a grid of squared cells with $40cm \times 40cm$ size (according to the average area occupied by a pedestrian [21]). Cells have a state indicating the fact that they are vacant or occupied by obstacles or pedestrians. The same cell can also be temporary occupied by two pedestrians, in order to allow simulation of overcrowded situations in which the density is higher than 6.25 ped/m^2 (i.e. the maximum density reachable by our discretisation).

The information related to the scenario[2] of the simulation are represented by means of *spatial markers*, special sets of cells that describe relevant elements in the environment. In particular, three kinds of spatial markers are defined: (i) *start* areas, that indicate the generation points of agents in the scenario.

[1] For a complete discussion of the model, see [20].

[2] It represents both the structure of the environment and all the information required for the realization of a specific simulation, such as crowd management demands (pedestrians generation profile, origin-destination matrices) and spatial constraints.

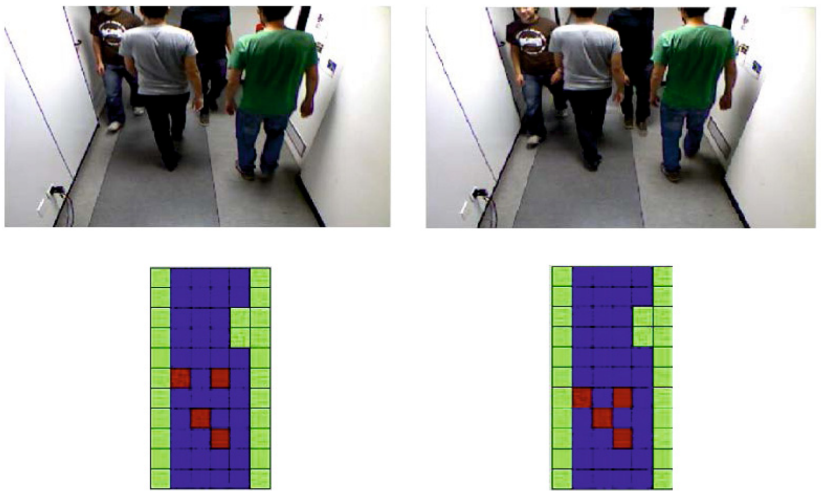

Fig. 6. An example of tracking through consecutive frames

Agent generation can occur in *block*, all at once, or according to a user defined *frequency*, along with information on type of agent to be generated and its destination and group membership; (ii) *destination* areas, which define the possible target locations of the pedestrians in the environment; (iii) *obstacles*, that identify all the non-walkable areas as walls and zones where pedestrians can not enter.

Space annotation allows the definition of virtual grids of the environment, as containers of information for agents and their movement. In our model, we adopt the *floor field* approach [3], that is based on the generation of a set of superimposed grids (similar to the grid of the environment) starting from the information derived from spatial markers. Floor field values are spread on the grid as a gradient and they are used to support pedestrians in the navigation of the environment, representing their interactions with static object (i.e., destination areas and obstacles) or with other pedestrians. Moreover, floor fields can be *static* (created at the beginning and not changed during the simulation) or *dynamic* (updated during the simulation). Three kinds of floor fields are defined in our model: (i) *path field*, that indicates for every cell the distance from one destination area, acting as a potential field that drives pedestrians towards it (static). One path field for each destination point is generated in each scenario; (ii) *obstacles field*, that indicates for every cell the distance from neighboring obstacles or walls (static); (iii) *density field*, that indicates for each cell the pedestrian density in the surroundings at the current time-step (dynamic).

Chessboard metric with $\sqrt{2}$ variation over corners [22] is used to produce the spreading of the information in the path and obstacle fields. Moreover, pedestrians cause a modification to the density field by adding a value $v = \frac{1}{d^2}$ to cells whose distance d from their current position is below a given threshold. Agents are able to perceive floor fields values in their neighborhood by means of a function $Val(f, c)$ (f represents the field type and c is the perceived cell). This approach to the definition of the objective part of the perception model moves the burden of its management from agents to the environment, which would need to monitor agents anyway in order to produce some of the simulation results.

3.2 Pedestrians and Movement

Formally, our agents are defined by the triple $Ped = \langle Id, \ Group, \ State \rangle$, where $State = \langle position, oldDir, \ Dest \rangle$, with their own numerical identifier, their group[3] (if any) and their internal state, that defines the current position of the agent, the previous movement and the final destination, associated to the relative path field.

Agent Behaviour. Agent behavior in a single simulation turn is organized into four steps: *perception, utility calculation, action choice* and *movement.* The *perception* step provides to the agent all the information needed for choosing its destination cell. In particular, if an agent does not belong to a group (from here called *individual*), in this phase it will only extract values from the floor fields, while in the other case it will perceive also the positions of the other group members within a configurable distance, for the calculation of the *cohesion* parameter. The choice of each action is based on an utility value assigned to every possible movement according to the function:

$$U(c) = \frac{\kappa_g G(c) + \kappa_{ob} Ob(c) + \kappa_s S(c) + \kappa_c C(c) + \kappa_d D(c) + \kappa_{ov} Ov(c)}{d} \tag{4}$$

$U(c)$ takes into account the behavioral components considered relevant for pedestrian movement, each one is modeled by means of a function that returns values in range $[-1; +1]$, if it represents an *attractive* element (i.e. its goal), or in range $[-1; 0]$, if it represents a *repulsive* one for the agent. For each function a κ coefficient has been introduced for its calibration: these coefficients, being also able to actually modulate tendencies based on objective information about agent's spatial context, complement the objective part of the perception model allowing agent heterogeneity. The purpose of the denominator d is to constrain the diagonal movements, in which the agents cover greater distances ($0.4 \times \sqrt{2}$ instead of 0.4) and assume higher speeds respect with the non-diagonal ones.

The first three functions exploit information derived by local floor fields: $G(c)$ is associated to goal attraction whereas $Ob(c)$ and $S(c)$ respectively to geometric and social repulsion. Functions $C(c)$ is a linear combination of the perceived positions of members of agent group in an extended neighborhood; they compute the level of attractiveness of each neighboring cell, relating to group cohesion phenomenon. Finally, $D(c)$ adds a bonus to the utility of the cell next to the agent according to his/her previous direction (a sort of *inertia* factor), while $Ov(c)$

[3] The model here described particularly considers social relationships between people. See [20] for a thorough discussion of this aspect.

describes the *overlapping* mechanism, a method used to allow two pedestrians to temporarily occupy the same cell at the same step, to manage high-density situations.

After the utility evaluation for all the cells in the neighborhood, the choice of action is stochastic, with the probability to move in each cell c as (N is the normalization factor): $P(c) = N \cdot e^{U(c)}$. On the basis of $P(c)$, agents move in the resulted cell according to their set of possible actions, defined as list of the eight possible movements in the Moore neighborhood, plus the action to keep the position (indicated as X): $A = \{NW, N, NE, W, X, E, SW, S, SE\}$.

3.3 Time and Update Mechanism

In the basic model definition time is also discrete; in an initial definition of the duration of a time step was set to 0.31 s. This choice, considering the size of the cell (a square with 40 cm sides), generates a linear pedestrian speed of about 1.3 m/s, which is in line with the data from the literature representing observations of crowd in normal conditions [21].

Regarding the update mechanism, three different strategies are usually considered in this context [23]: *ordered sequential*, *shuffled sequential* and *parallel* update. The first two strategies are based on a sequential update of agents, respectively managed according to a *static* list of priorities that reflects their order of generation or a *dynamic* one, shuffled at each time step. On the contrary, the parallel update calculates the choice of movement of all the pedestrians at the same time, actuating choices and managing conflicts in a latter stage. The two sequential strategies, instead, imply a simpler operational management, due to an a-priori resolution of conflicts between pedestrians. In the model, we adopted the parallel update strategy. This choice is in accordance with the current literature, where it is considered much more realistic due to consideration of conflicts between pedestrians, arisen for the movement in a shared space [4].

With this update strategy, the agents life-cycle must consider that before carrying out the *movement* execution potential conflicts, essentially related to the simultaneous choice of two (or more) pedestrians to occupy the same cell, must be solved. The overall simulation step therefore follows a three step procedure: (i) *update of choices* and *conflicts detection* for each agent of the simulation; (ii) *conflicts resolution*, that is the resolution of the detected conflicts between agent intentions; (iii) *agents movement*, that is the update of agent positions exploiting the previous conflicts resolution, and *field update*, that is the computation of the new density field according to the updated positions of the agents.

The resolution of conflicts employs an approach essentially based on the one introduced in [4], based on the notion of friction. Let us first consider that conflicts can involve two of more pedestrians: in case more than two pedestrians involved in a conflict for the same cell, the first step of the management strategy is to block all but two of them, chosen randomly, reducing the problem to the case of a simple conflict. To manage this latter, another random number $\in [0, 1]$ is generated and compared to two thresholds, $frict_l$ and $frict_h$, with $0 < frict_l < frict_h \leq 1$: the outcome can be that all agents yield when the extracted number is lower than $frict_l$, only one agent moves (chosen randomly) when the extracted number is between $frict_l$ and $frict_h$ included, or even two agents move when the number is higher than $frict_h$ (in this case pedestrian overlapping occurs).

4 Automated Simulation Evaluation

Given the data which can be obtained with the methodology described in Sec. 2.5, a preliminary evaluation of the pedestrian model is based on metrics describing the space utilisation of pedestrians, that is, the way they walked in the analyzed/simulated scenario, facing the presence of obstacles or other people. Real world data for the simulation evaluation have been achieved with a small set of experiments in a corridor section, by performing 3 different scenarios. In the tests the corridor was crossed by respectively: (i) 1 person per side; (ii) 1 person from one side and 2 from the other; (iii) 2 persons per side. The simulation environment and some frame of the video is shown in Fig. 4 - 5.

The evaluation of the simulation model is automated by the tool in a simple way. In this phase, the *pedestrian simulation module* receives major inputs for the simulation configuration from the computer vision module (i.e., the scenario setting with the time schedule of pedestrian generation, obtained by analysing the boundaries of the observed environment). The calibration parameter set, for each simulation, is provided by the *automated evaluation module.* The main objective of this module is the investigation of the correct calibration for the simulator: starting from an initial configuration of the calibration weights, provided by the user, and a set of variable calibration parameters it issues, through the *simulation module*, a set of simulations with different configurations of weights. Once a single simulation is finished, results are compared with the observed data according to a user defined metric, that analyses one or more effects of the human behaviour. While the comparison is not acceptable (i.e., the difference between data is greater than a user defined threshold), the range of the calibration weights is explored with new simulations. Metrics used for the evaluation, with simulation results are discussed in the following susections.

4.1 Average Pedestrian Distances

A well-known effect of the human behaviour is the preservation of particular, physical distances among other people, differentiated by the situation and the relationship had with them in different situations. Studies in the field of anthropology [24] inform about average values of these *social* distances. On the other hand, the adaptivity of the human behaviour leads to high variability of distances regarding different situations: with the increasing of pedestrian densities as well as with incoming flows from other directions. Automatic calculation of the observed distances between pedestrians is therefore needed for better understanding if the simulation model is able to reproduce them properly.

Starting from the position of every pedestrian gathered in each frame of the video, distances D_{P_i,P_j}, less than a threshold r_{ped}[4], are collected for calculating

[4] We assumed $r_{ped} = 1.2m$, in order to represent the *personal space* of pedestrians.

the average. Then, the evaluation is performed by comparing this value with the one obtained by using the positions of the agents during the simulation. For this evaluation, only the parameter κ_s has been tuned by the automated tool.

Table 1 compares the data gathered with the three experiments, described at the beginning of this section, with the results achieved by simulating each respective scenario. After the calibration phase, it has been found an optimal value of κ_s close to 30. It is possible to see that, while with 3 and 4 persons in the scenario the simulated data are close to the real ones, in the case with 2 pedestrians simulations have an error near to 0.3 meters. This is probably due to the missing of a mechanism for managing the *anticipation*, or reservation of space between simulated pedestrians, as already explained in [25]. This mechanism would improve the cooperation between agents, letting them to avoid trajectories which lead to conflicts and to too short distances with other persons in low density situations.

Table 1. Comparison of average pedestrian lowest distances

Scenario	Real World [m]	Simulation [m]
(i)	1.05	0.76
(ii)	0.79	0.81
(iii)	0.82	0.81

4.2 Average Distances with Obstacles

In order to analyze the reproduction of trajectories by the simulator, another indicator must describe the distances maintained with obstacles and walls in the environment. With this aim, this analysis uses the positions of pedestrians and the configuration of the environment for calculating the average distance between pedestrian and obstacles. In particular, for each pedestrian and each frame, the minimum distance between its position and the nearby obstacles is calculated. If this is below a distance threshold r_{obs} (for the simulation we used $r_{obs} = 1.2$), it will be added to the set used for the average calculation.

Table 2 compares real world and simulation results. After the calibration phase, value of κ_{obs} has been rounded to 3.

Table 2. Evaluation of average distances with obstacles

Scenario	Real World [m]	Simulation [m]
(i)	0.53	0.64
(ii)	0.61	0.63
(iii)	0.56	0.63

5 Conclusions

In this paper an automated tool for the evaluation of pedestrian simulation models has been presented. The developed tool has been tested using real data versus simulated ones, produced by an existing pedestrian simulator [20] and the overall testing procedure has been analyzed. The tool performs adequately, highly improving the calibration and evaluation of the simulation model both in accuracy and in overall time. The automation of the procedure opens new areas of research. Future work is mainly focused on two directions. Firstly on the improvement of the tool itself and, secondly, on the fully automated calibration of a pedestrian simulator. In particular, the simulation evaluation procedure will be improved including data about local densities distribution in the space, which can be calculated based on the position of pedestrians. In addition, improving the output of the computer vision module with even more accurate tracking techniques, will enable the system to estimate sturdy instant velocities of people.

References

1. Helbing, D., Molnár, P.: Social force model for pedestrian dynamics. Phys. Rev. E 51(5), 4282–4286 (1995)
2. Moussad, M., Perozo, N., Garnier, S., Helbing, D., Theraulaz, G.: The walking behaviour of pedestrian social groups and its impact on crowd dynamics. PLoS ONE 5(4), e10047 (2010)
3. Burstedde, C., Klauck, K., Schadschneider, A., Zittartz, J.: Simulation of pedestrian dynamics using a two-dimensional cellular automaton. Physica A: Statistical Mechanics and its Applications 295(3-4), 507–525 (2001)
4. Kirchner, A., Nishinari, K., Schadschneider, A.: Friction effects and clogging in a cellular automaton model for pedestrian dynamics. Phys. Rev. E 67, 056122 (2003)
5. Henein, C.M., White, T.: Agent-based modelling of forces in crowds. In: Davidsson, P., Logan, B., Takadama, K. (eds.) MABS 2004. LNCS (LNAI), vol. 3415, pp. 173–184. Springer, Heidelberg (2005)
6. Shao, W., Terzopoulos, D.: Autonomous pedestrians. Graphical Models 69(5-6), 246–274 (2007)
7. Willis, A., Gjersoe, N., Havard, C., Kerridge, J., Kukla, R.: Human movement behaviour in urban spaces: Implications for the design and modelling of effective pedestrian environments. Environment and Planning B: Planning and Design 31(6), 805–828 (2004)
8. Schultz, M., Schulz, C., Fricke, H.: Passenger dynamics at airport terminal environment. In: Klingsch, W.W.F., Rogsch, C., Schadschneider, A., Schreckenberg, M. (eds.) Pedestrian and Evacuation Dynamics 2008, pp. 381–396. Springer, Heidelberg (2010)
9. Bandini, S., Gorrini, A., Vizzari, G.: Towards an integrated approach to crowd analysis and crowd synthesis: A case study and first results. arXiv preprint arXiv:1303.5029 (2013)
10. Kostavelis, I., Boukas, E., Nalpantidis, L., Gasteratos, A.: A mechatronic platform for robotic educational activities. Interdisciplinary Mechatronics, 543–568

11. Kostavelis, I., Boukas, E., Nalpantidis, L., Gasteratos, A., Rodrigalvarez, M.A.: Spartan system: Towards a low-cost and high-performance vision architecture for space exploratory rovers. In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), pp. 1994–2001. IEEE (2011)
12. Konolige, K.: Small vision systems: Hardware and implementation. In: International Symposium on Robotics Research, pp. 111–116 (1997)
13. Nalpantidis, L., Sirakoulis, G.C., Gasteratos, A.: Review of stereo vision algorithms: From software to hardware. International Journal of Optomechatronics 2(4), 435–462 (2008)
14. Kostavelis, I., Nalpantidis, L., Gasteratos, A.: Supervised traversability learning for robot navigation. In: Groß, R., Alboul, L., Melhuish, C., Witkowski, M., Prescott, T.J., Penders, J. (eds.) TAROS 2011. LNCS, vol. 6856, pp. 289–298. Springer, Heidelberg (2011)
15. De Cubber, G., Doroftei, D., Nalpantidis, L., Sirakoulis, G.C., Gasteratos, A.: Stereo-based terrain traversability analysis for robot navigation. In: IARP/EURON Workshop on Robotics for Risky Interventions and Environmental Surveillance, Brussels, Belgium (2009)
16. Vannoorenberghe, P., Motamed, C., Blosseville, J.M., Postaire, J.G.: Monitoring pedestrians in a uncontrolled urban environment by matching low-level features. In: IEEE International Conference on Systems, Man, and Cybernetics, vol. 3, pp. 2259–2264. IEEE (1996)
17. Viola, P., Jones, M.J., Snow, D.: Detecting pedestrians using patterns of motion and appearance. International Journal of Computer Vision 63(2), 153–161 (2005)
18. Dollar, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: An evaluation of the state of the art. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(4), 743–761 (2012)
19. Cheng, Y.: Mean shift, mode seeking, and clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence 17(8), 790–799 (1995)
20. Vizzari, G., Manenti, L., Crociani, L.: Adaptive pedestrian behaviour for the preservation of group cohesion. Complex Adaptive Systems Modeling 1(7) (2013)
21. Weidmann, U.: Transporttechnik der fussgänger - transporttechnische eigenschaftendes fussgängerverkehrs (literaturstudie). Literature Research 90, Institut füer Verkehrsplanung, Transporttechnik, Strassen- und Eisenbahnbau IVT an der ETH Zürich (1993)
22. Kretz, T., Bönisch, C., Vortisch, P.: Comparison of various methods for the calculation of the distance potential field. In: Klingsch, W.W.F., Rogsch, C., Schadschneider, A., Schreckenberg, M. (eds.) Pedestrian and Evacuation Dynamics 2008, pp. 335–346. Springer, Heidelberg (2010)
23. Klüpfel, H.: A Cellular Automaton Model for Crowd Movement and Egress Simulation. PhD thesis, University Duisburg-Essen (2003)
24. Hall, E.T.: The Hidden Dimension. Anchor Books (1966)
25. Suma, Y., Yanagisawa, D., Nishinari, K.: Anticipation effect in pedestrian dynamics: Modeling and experiments. Physica A: Statistical Mechanics and its Applications 391(1), 248–263 (2012)