Data-driven simulation of pedestrian collision avoidance with a nonparametric neural network

Rafael F. Martin[a,*], Daniel R. Parisi[b]

[a] *Instituto Tecnológico de Buenos Aires. Lavarden 315, (C1437FBG) C. A. de Buenos Aires, Argentina.*
[b] *Instituto Tecnológico de Buenos Aires. CONICET. Lavarden 315, (C1437FBG) C. A. de Buenos Aires, Argentina.*

Abstract

Data-driven simulation of pedestrian dynamics is an incipient and promising approach for building reliable microscopic pedestrian models. We propose a methodology based on generalized regression neural networks, which does not have to deal with a huge number of free parameters as in the case of multilayer neural networks. Although the method is general, we focus on the one pedestrian - one obstacle problem. Experimental data were collected in a motion capture laboratory providing high-precision trajectories. The proposed model allows us to simulate the trajectory of a pedestrian avoiding an obstacle from any direction. Together with the methodology specifications, we provide the data set needed for performing the simulations of this kind of pedestrian dynamic system.

Keywords: pedestrian dynamics, data-driven simulation, navigation, steering, generalized regression neural network, artificial intelligence.

1. Introduction

The common practice when developing models for any nonequilibrium and many-particle system consists of proposing a model with a certain number of parameters and then, adjust them such that it can reproduces some observables

*Corresponding author:ramartin@itba.edu.ar

from the given system. Because, in principle, a huge number of models could be postulated, it is not easy to guarantee if there is a correct one. Particularly this is the case in pedestrian dynamics modeling: one can find a large number of models in the literature (see, for example, the reviews in [1, 2]) and none of them can simulate the wide diversity of pedestrian system configurations [2].

One possible way to overcome this limitation could be to skip the system modeling stage in the typical process described by: 1)Data Extraction; 2)Modeling; 3)Simulation. We could develop reliable simulation directly from the data, simplifying the above process as: 1)Data Extraction; 2)Simulation. We call this second scheme "data-driven simulation". One of the beauties of this approach is that simulations will be validated with experimental data by definition. Also, it could be computational efficient, and thus becomes a new paradigm to be transferred to commercial simulation software, improving the operational design and safety of pedestrian facilities.

In the present work we explore a possible way of putting together a particular data-driven formulation.

Recently, we proposed a general framework of pedestrian simulation [3] in which the surroundings of a virtual pedestrian, i.e., obstacles and other non-contacting particles, can only influence its trajectory by modifying its desired velocity.

The basic assumption is that the avoidance behavior can be exerted only by the self-propelled mechanism of the particle itself (usually modeled by the desired velocity).

This framework is independent of the type of low-level model being force-based, rule-based or other. For example, it could be implemented on the Social Force Model [4, 5], by replacing the social force, with a variable desired velocity that takes into account the possible future collisions [6].

Of course, this framework can also be implemented on a first-order model, in which the position ($\mathbf{r}$) of any particle can be updated by [$\mathbf{r}(t + \Delta t) = \mathbf{r}(t) + \mathbf{v}(t)\Delta t$] by dynamically adjusting the desired velocity [$\mathbf{v}(t)$].

Under this approach, the problem lies in postulating the heuristics required

for computing the variable desired velocity depending on the environment. As in traditional pedestrian theoretical models, any arbitrary heuristic can be proposed (for example, [7], [8]) and then the free parameters could be tuned in order to obtain simulated trajectories that approach experimental micro or macroscopic data.

Instead of this traditional methodology, we can directly use the experimental data so as to compute the desired velocity at each time step. More precisely, we postulate that a minimum set of real trajectories exist, which could have the complete information for providing a desired velocity to the simulated agent, considering the state of the agent and its surroundings in the simulated and experimental environment. This idea of using experimental data in a simulated environment is at the core of our data-driven simulation scheme.

Alternatively, the problem of simulating pedestrian dynamics could be seen in another dimension. The purpose of any model is to map current positions of particles [$\mathbf{r}(t)$, sometimes called 'state' or 'input'] into the positions at the next time step [$\mathbf{r}(t + \Delta t)$, or 'action' or 'output']. Again, this mapping could be achieved by the large number of traditional pedestrian models but also by using available experimental data.

In this sense, data-driven simulations have the benefit of avoiding the proposal of explicit models along with their parameters, which can be related or not to reality. Instead, the experimental data can be considered directly for mapping the past positions into the future ones. No *a priori* model assumption or guess needs to be made when simulating the pedestrian system. All the necessary information would be provided by the data from real scenarios.

Previous research papers using data-driven simulation have exploited the data directly [9, 10, 11] or through artificial neural networks (NN) [12, 13, 14, 15, 16]. This computation paradigm is a natural choice, because the NN can be "trained" with the experimental data and then it could be applied in the simulation when mapping old positions [$\mathbf{r}(t)$] into the future ones [$\mathbf{r}(t + \Delta t)$].

In general, these previous papers consider particular data sets corresponding to specific configurations (geometry and pedestrian flow) and then use these data

for simulating the same system configuration. How the data-driven methodology could be generalized for simulating arbitrary (previously unseen) geometries and flows remains an open question.

Artificial intelligence tools were also applied for crowd characterization and people counting from images and video [17, 18].

In the case of using neural networks, previous papers implement Multilayer Perceptron [19] (MLP, also known as Feed-Forward or Back-Propagation Neural Networks). These kinds of networks are very popular but their architecture presents an arbitrary number of hidden layers, each one with an arbitrary number of neurons. This leads to an also arbitrary number of free parameters ('weights') that should be determined via the training process using 'input/output' pairs ('patterns' or examples) from experimental data of the real system. For this reason, the number of patterns has to be much greater than the number of free parameters in order to find a reliable set of weights. In any case, by using an MLP the data are interpolated with a model having a huge number of parameters, which is of course, an undesirable property for any model.

Here we propose a data-driven approach using a nonparametric universal interpolator: the generalized regression neural network (GRNN) [20] (Sec. 2.3.3). The GRNN needs to have access to the data examples (patterns) when predicting a new output. However, because it has only one degree of freedom (only one free parameter), the number of (input/output) patterns can be relatively low. And this brings us to the second novelty. We postulate that a complete set of (input/output) examples, extracted from a limited number of experimental trajectories, could be sufficient for simulating and reproducing any arbitrary pedestrian dynamic configuration. As a starting point, here we present this methodology in the case of one pedestrian avoiding a fixed obstacle. As the methodology is general, it will be implemented in more general scenarios of pedestrian dynamics in future work.

2. The data-driven model

In this section, the proposed data-driven model is explained in detail.

2.1. General framework

Our general framework [3] postulates that a particle i, with position $\mathbf{r}_i$, has a temporary and short-range goal $\mathbf{T}_i^t(t)$ that is dynamically placed depending on the environment. $\mathbf{T}_i^t$ will produce a detour in the trajectory in order to avoid any collision. The environment is defined by the fixed long-distance goal ($\mathbf{T}_i$), the positions ($\mathbf{r}_j(t)$) and the relative velocities ($\mathbf{v}_{ij}$) of the nearest neighbors and obstacles. A graphical representation of these variables is presented in Fig.1.

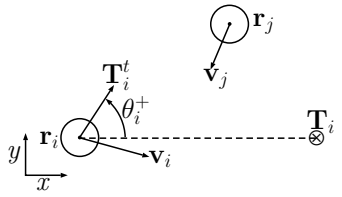


Figure 1: Basic quantities of the general framework defining the environment of particle i and the placement of the temporary local goal.

We denote the general function that receives these relevant variables and returns the temporary and short-range goal as $\mathbf{H}$:

$$\mathbf{T}_i^t(t) = \mathbf{H}(\mathbf{r}_i(t), \mathbf{r}_j(t), \mathbf{v}_{ij}(t), \mathbf{T}_i) \tag{1}$$

The vector $\mathbf{T}_i^t$ determines the avoidance direction, but it also has a magnitude that allows us to adjust the speed of the agent. The function $\mathbf{H}$ is completely general and, of course, it can take any form. Again, we remark that this formulation does not depend on the type of low-level operational model. Thus, we choose a first-order model for describing the evolution of the particle, because we are not considering any forces, and besides, it presents a higher computation speed than a second-order model.

The simulated particle has position $\mathbf{r}^s(t)$ and velocity $\mathbf{v}^s(t)$ at time t with a fixed long-distance goal $\mathbf{T}$.

$$\mathbf{r}^s(t + \Delta t) = \mathbf{r}^s(t) + \mathbf{v}^s(t)\Delta t, \tag{2}$$

where we can directly identify the dynamic target with the desired velocity $\mathbf{v}^s(t) = \mathbf{T}^t(t)$.

2.2. Experimental trajectories

Because this is a data-driven model, the experimental data are the first ingredient needed for obtaining the velocity ($\mathbf{v}^s(t)$) in eq. 2.

As a case study of the proposed method, we will focus on a simple configuration, considering one pedestrian and one fixed obstacle.

The experimental setup consists of a 6 m circumference where we locate the starting points ($\mathbf{S}_p$) and the final target $\mathbf{T}$ in a relative opposite location, as shown in Fig. 2. At the center of the circumference a fixed obstacle is placed, which has the approximate size of a pedestrian.

(a) (b)

Figure 2: Experimental setting. (a) Snapshot of the experiment. (b) Schematic representation of starting points ($\mathbf{S}_p$) and final target ($\mathbf{T}$) for the recorded trajectories.

The experiments were performed with the participation of four volunteers in the Motion Capture Laboratory located at the 'Instituto Tecnológico de Buenos Aires'.

The volunteers were instructed to walk normally from the starting points to the final target. Each volunteer wore a cap with three markers. Throughout

the experiment, each pedestrian had to walk less than 500 m (not continuously) inside the measurement area at normal speed and with no physical contact. Under these conditions, the experimental protocol did not involve any risks, protecting the integrity, privacy, and confidentiality of the research subjects.

The position of the markers was recorded using the commercially available technology from Optitrack ®. Each marker position was captured by 16 Flex3® cameras located throughout the recording environment at 33 frames per second and then processed using the Motive: Body® software. Pedestrians were tracked using the position of the three markers, but we considered the position of the pedestrian as that of the highest (central) one and the others were used only for reconstruction when, for very short periods, the acquisition system failed to record the position of the central one. The precision of the technology locating a marker in the 3-D space was 1 cm.

The 3-D spatial trajectories obtained were further processed. First, only the two components belonging to the horizontal plane were kept and the height was ignored. Then, in order to neglect the natural swaying of human walking, a Fourier filter was applied to each of the two horizontal components of the trajectories $\mathbf{r}(t) = [x(t),\ y(t)]$.

Finally, $N_o = 10$ original trajectories were selected from all the pedestrian data, which are displayed as solid lines in Fig.3.

The starting points ($\mathbf{S}_p$) near $y \sim 0$ produced trajectories with detours needed for avoiding the obstacle. However, in the extreme starting locations ($\mathbf{r}_1 and \mathbf{r}_1 0$), the trajectories were almost straight because a direct trajectory toward the final target would not intersect the obstacle, and thus it would not be necessary to dodge it.

In order to have a complete set of trajectories, we replicate these extreme trajectories by rotating them around the final target, spanning the rest of the angular positions, from which the simulated pedestrian can move directly toward the final target without performing any avoidance but keeping the natural variations in speed as its distance to the final target changes. We choose $N_r = 8$ replicated and rotated trajectories in strategic positions that can be seen as

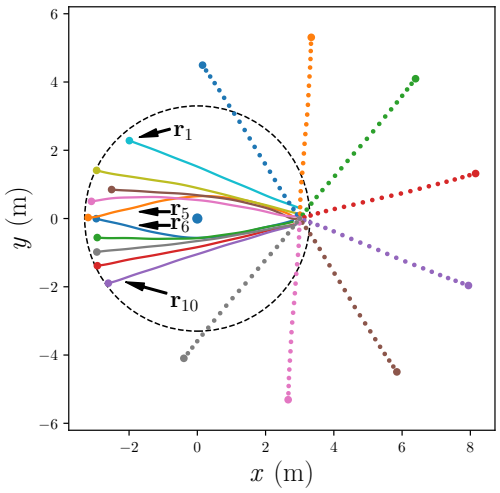


Figure 3: Experimental trajectories (solid lines) and rotated trajectories (dashed lines). The points indicate the initial position of trajectories. The final target for all trajectories is located at $(x,\ y) = (0,\ 3\ \text{m})$ coordinates.

dashed lines in Fig. 3.

The decision about the number of trajectories considered (N_o and N_r) will be explained in Sec. (3.3).

Summarizing, the $N = N_o + N_r = 18$ experimental trajectories in Fig. 3 contain the information for a pedestrian approaching a target from any angular position, having or not an obstacle to avoid. Note that this information can be coded in the particle's system of reference (see Sec. 2.2) and thus it is general, in the sense that the relative position of the obstacle can be arbitrary. The described trajectory data can be found in the supplementary material.

2.3. A neural network instance of $\boldsymbol{H}$

We define the input state seen from the particle that will allow us to compute the output action ($\mathbf{v}^s$ in eq. 2) as the temporary target from eq. (1).

Input / output pairs will be obtained from the experimental set of trajec-

tories. Then, a general regression neural network will take these examples for predicting new outputs as from the simulated environment (inputs).

2.3.1. Input

In a two-body problem, we can consider the pedestrian i, who has position $\mathbf{r}_i(t)$ at time t whose goal is $\mathbf{T}_i$, and any other arbitrary pedestrian or obstacle j with position $\mathbf{r}_j(t)$. We postulate a continuous input state given by the vector $\boldsymbol{\xi}_{ij}(t)$ (eq. 3). In the case of two particles the dimension of the input space is 6 (2 for particle i and 4 for the other particle). Of course, the input vector can be generalized; if there were more particles, its dimension would increase by the amount of 4 for each extra particle.

$$\boldsymbol{\xi}_{ij}(t) = [\ \hat{v}_i,\ \hat{\theta}_{ij},\ \hat{d}_{iT},\ \hat{v}_{ij},\ \hat{\theta}^v_{ij},\ \hat{d_{ij}}\ ] \tag{3}$$

In what follows we describe the variables of the input space. First, in order to make all the variables compatible, because of their different units, and spanning over different ranges, we define them as dimensionless by rescaling to values $\lesssim 2$.

- $\hat{v}_i = \dfrac{|\mathbf{v}_i(t^-)|}{1.8\ \mathrm{m/s}}$, where 1.8 m/s is the mean speed observed in our experiments, $\mathbf{v}_i(t^-)$ is the past velocity of pedestrian i at time t calculated as $\mathbf{v}_i(t^-) = [\mathbf{r}_i(t) - \mathbf{r}_i(t-k)]/(k\Delta t)$ (Fig. 4 (a)).

- $\hat{\theta}_{ij} = \begin{cases} 1 & \text{if } \theta_{ij} \geq \pi/2 \\ -1 & \text{if } \theta_{ij} \leq -\pi/2 \\ \theta_{ij}\ 2/\pi & \text{otherwise} \end{cases}$

  where θ_{ij} is the angle defined between the vectors $(\mathbf{T}_i - \mathbf{r}_i(t))$ and $(\mathbf{r}_j(t) - \mathbf{r}_i(t))$ as shown in Fig. 4 (b) and it lies between the interval $[-\pi, \pi]$. However, the input angle $\hat{\theta}_{ij}$ saturates when $|\hat{\theta}_{ij}| \geq \pi/2$, which makes the particle ignore the obstacles behind it. Also note that this variable takes

positive and negative values aiming to distinguish between right and left from the particle looking toward the final target.

- $\hat{d}_{iT} = \begin{cases} d_{iT}/4\text{m} & \text{if } d_T \leq 8\text{m} \\ 2 & \text{otherwise} \end{cases}$

  d_{iT} being the distance between the particle and its final target ($d_{iT} = |\mathbf{T}_i - \mathbf{r}_i(t)|$). The saturation value (8 m) causes obstacles beyond that distance to be neglected.

- $\hat{v}_{ij} = \dfrac{|\mathbf{v}_{ij}|}{1.8\, m/s}$, where $\mathbf{v}_{ij}$ is the relative velocity of j seen from particle i ($\mathbf{v}_{ij} = \mathbf{v}_j - \mathbf{v}_i$).

- $\hat{\theta}^v_{ij} = \begin{cases} -2(\theta^v_{ij} + \pi)/\pi & if \quad \theta^v_{ij} < -\pi/2 \\ -1 & if \quad -\pi/2 < \theta^v_{ij} < 0 \\ +1 & if \quad 0 < \theta^v_{ij} \leq \pi/2 \\ -2(\theta^v_{ij} - \pi)/\pi & if \quad \theta^v_{ij} > \pi/2 \end{cases}$

  where θ^v_{ij} is the angle between the vectors ($\mathbf{T}_i - \mathbf{r}_i(t)$) and the relative velocity ($\mathbf{v}_j - \mathbf{v}_i$) as shown in Fig. 4 (c). This function saturates for values in the range $|\theta^v_{ij}| < \pi/2$ because in this case the particle j would be moving away from particle i and as a result, there cannot be any collision.

- $\hat{d}_{ij} = \begin{cases} d_{ij}/4\text{m} & \text{if } d_{ij} \leq 8\text{m} \\ 2 & \text{otherwise} \end{cases}$

  where d_{ij} is the Euclidean distance between the particles ($|\mathbf{r}_i(t) - \mathbf{r}_j(t)|$).

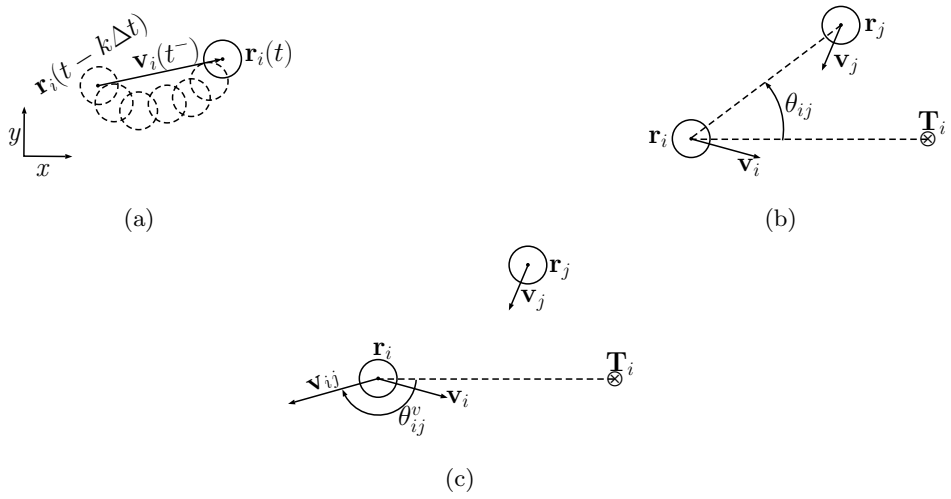


Figure 4: Basic quantities needed for defining the input vector ($\boldsymbol{\xi}$). (a) Past velocity $\mathbf{v}_i(t^-)$ of pedestrian i. (b) Relative angle θ_{ij} between both pedestrians. (c) Relative velocity $\mathbf{v}_{ij}$ with angle θ^v_{ij}.

2.3.2. Output

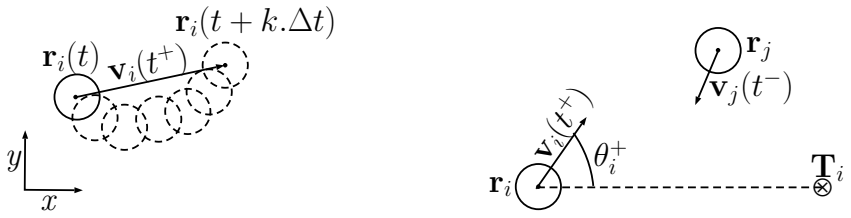
The reaction of the agent will be modeled as its velocity, which in our first-order model will allows us to move it toward its future position (eq. 2). This velocity will be defined in polar coordinates relative to the direction between the particle and its final target. We call this angle θ_i^+, which is defined between $\mathbf{T}_i$ and $\mathbf{T}_i^t$ [see Fig. 1 or 5 (b)]; this definition allows us to have a rotation-invariant data set. In consequence, the output vector has only two dimensions regardless of the number of particles in the system.

$$\boldsymbol{\zeta}_i(t) = [\ {}^1\zeta(t),\ {}^2\zeta(t)\ ] = [\ v_i^+,\ \theta_i^+\ ]. \tag{4}$$

Here, v_i^+ is the speed of particle i for the next time step calculated as the magnitude $v_i^+ = |\mathbf{v}_i| = |\mathbf{r}_i(t+k\Delta t) - \mathbf{r}_i(t)|/(k\Delta t)$ (Fig. 5 (a)). And θ_i^+ is the angle of the velocity with respect to the direction defined by $(\mathbf{T}_i - \mathbf{r}_i(t))$ (Fig. 5 (b)).

If there are no data at time $(t + k\Delta t)$, the output is not calculated and

therefore, the corresponding input is not considered.



(a) (b)

Figure 5: Definition of the output vector ($\boldsymbol{\zeta}$). (a) Future velocity $\mathbf{v}_i(t^+)$ of pedestrian i. (b) Angle of the future velocity relative to the direction to the final target (θ_i^+).

2.3.3. The nonparametric neural network

The above definitions of input/output pairs (eqs. 3 and 4) can be used for an arbitrary number of interacting particles. However, as stated above, we will consider a simple experimental configuration with one moving particle and one obstacle.

We call $\mathbb{E} = \{\boldsymbol{\xi}(t), \boldsymbol{\zeta}(t)\}_t$ the experimental set of state/action examples obtained from the data points for each time step t.

Each one of the two components of the output vector $\boldsymbol{\zeta}(t)$ (eq. 4) will be approximated by one neural network with output ${}^{\mu}O : \mathbb{R}^6 \rightarrow \mathbb{R}$, where $\mu = 1, 2$ indicates its polar components, i.e., the speed (v_i^+) and the angle (θ_i^+) respectively.

The neural network we choose is the generalized regression neural network (GRNN) [20], which is a type of radial basis function network [21].

The GRNN is a universal interpolator based on nonparametric regression. The basic idea is that when trying to predict the output for a new input, the data examples are used in the following way: first, the distance between the new input and the data inputs is calculated, then the corresponding data outputs are weighed with a kernel function, depending on that distance, and averaged. In other words, the data outputs of closer data inputs are used for interpolating the new output.

In what follows we explain this concept explicitly for a network with one dimensional output (O). Suppose a training family of ordered pairs $\{\boldsymbol{\xi}_n, \zeta_n\}_{n \leq N}$, then:

$$O(\boldsymbol{\xi}) = \frac{\sum_{n=1}^{N} \zeta_n K(\boldsymbol{\xi}, \boldsymbol{\xi}_n)}{\sum_{n=1}^{N} K(\boldsymbol{\xi}, \boldsymbol{\xi}_n)} \tag{5}$$

where

- $O(\boldsymbol{\xi})$ is the prediction value of an arbitrary input vector $\boldsymbol{\xi}$.
- ζ_n is the output of example n corresponding to the input vector $\boldsymbol{\xi}_n$.
- $K(\boldsymbol{\xi}, \boldsymbol{\xi}_n) = e^{-l_n/2\sigma^2}$ is the radial basis function kernel that weighs the contribution of the *nth* output example in order to predict the new output.

Where $l_n = (\boldsymbol{\xi} - \boldsymbol{\xi}_n)^T(\boldsymbol{\xi} - \boldsymbol{\xi}_n)$ is the square distance between data examples $\boldsymbol{\xi}_n$ and the input vector $\boldsymbol{\xi}$.

Once we have a proper set of N patterns, the only degree of freedom in this neural network is the so-called spread (σ), which can be taken as a scalar value for all examples and variables of the input vector.

3. Simulations

In this section we describe how the spread (σ) of both GRNN's was calibrated and we present results showing that with the proposed approach we can simulate several configurations of a pedestrian avoiding an obstacle, within and beyond the boundaries of the experimental data.

3.1. Data-driven simulation scheme

The proposed data-driven simulation method is shown in schematic form in Fig. 6.

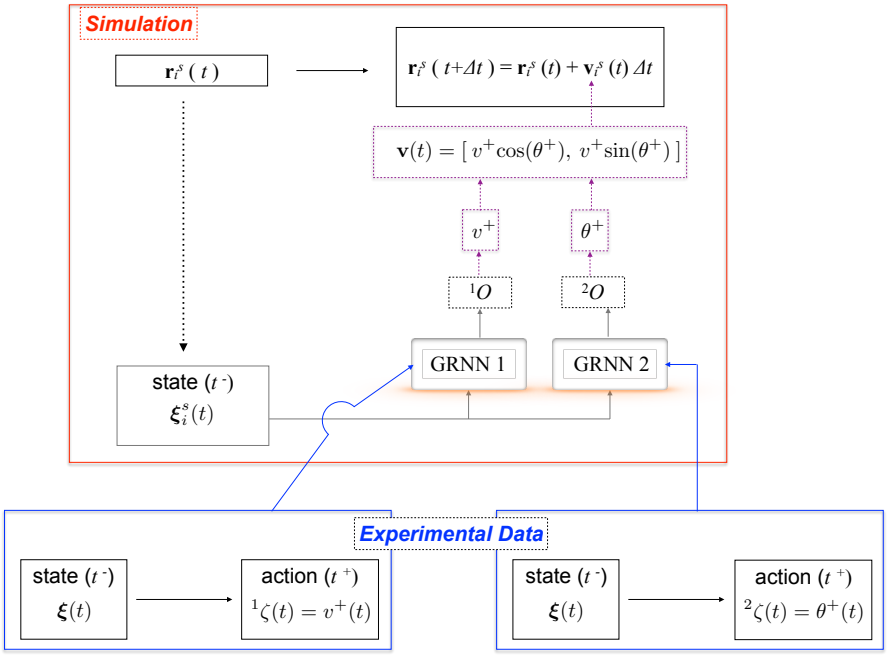


Figure 6: Flow diagram of the simulation procedure by estimating the future velocity of the simulated particle by means of two GRNN's that use data extracted from the same experimental set of trajectories. All symbols were defined in previous sections. Superscript s stands for *simulated* positions and velocities.

At each time step of the simulation, both GRNN's will provide the speed and angle of the velocity for computing the future position of the particle. In order to do this, the GRNN's will use the experimental data set $\mathbb{E} = \{\boldsymbol{\xi}(t), \boldsymbol{\zeta}(t)\}$, which is provided in the supplementary material. As stated in Sec. 2.3.3, there is only one free parameter for each GRNN: the spread (σ). In the next section we specify how this parameter was determined.

3.2. Calibrating the GRNN

In order to synchronize the simulation with the experimental data, we consider a time step $\Delta t = 1/33$ s corresponding to the maximum time resolution of the acquisition system ($k = 1$ in Fig. 4 (a) and 5 (a)). Considering that the training data set is composed of 18 trajectories (shown in Fig. 3)

each having 95 positions on average, the number n of input/output patterns is $n \sim 18 \times 95 = 1710$.

We can write the set of all data points as a collection of 18 subsets corresponding to each trajectory ${}^{18}\boldsymbol{\xi}_n = \{\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_{18}\}$, where $\mathbf{r}_i = [\mathbf{r}_i(t)]_{t \leq t_i^f}$ is the succession of pedestrian positions at discrete time steps Δt and t_i^f is the final time of $\mathbf{r}_i$. The first 10 trajectories correspond to original data, and trajectories from 11 to 18 are replications of the extreme trajectory $\mathbf{r}_1$ rotated as explained in Sec. 2.2.

Now, for the determination of the spread in each GRNN, we consider data patterns from the 10 original trajectories ${}^{10}\boldsymbol{\xi}_n = \{\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_{10}\}$ from which we will try to reproduce the 6 trajectories having data trajectories at both sizes, i.e., the 6 original trajectories different than $\mathbf{r}_1$, $\mathbf{r}_5$, $\mathbf{r}_6$ and $\mathbf{r}_{10}$, and proceed with a leave-one-out cross-validation. In other words, we take out each trajectory $\mathbf{r}_i$ with $i = 2, 3, 4, 7, 8, 9$ from the set of patterns $({}^{10}\boldsymbol{\xi}_n)$ to reconstruct the same trajectory $(\mathbf{r}_i^s)$ by simulating it with the methodology described in Fig. 6 considering the nine remaining trajectories as data examples for the GRNN's. Because each simulated trajectory needs two neural networks (one for each polar coordinate) the global error will be a function of both of them. We call $E_i(\sigma_1, \sigma_2)$ the error when comparing $\mathbf{r}_i$ with $\mathbf{r}_i^s$. Then, $E(\sigma_1, \sigma_2) = \frac{\sum_{i=2,3,4,7,8,9} E_i(\sigma_1, \sigma_2)}{6}$ is the global error for these spread values.

As boundary conditions for each simulated trajectory $\mathbf{r}_i^s$ we take the initial position $[\mathbf{r}_i^s(0) = \mathbf{r}_i(0)]$ and velocity $\mathbf{v}_i^s(0) = [\mathbf{r}_i(\Delta t) - \mathbf{r}_i(0)]/\Delta t$ equal to the experimental ones. The final target is selected as the last position of the experimental trajectory $[\mathbf{T}_i = \mathbf{r}_i(t_i^f)]$.

We define two different average error functions between simulated and experimental trajectories, one based on the minimum distance to the obstacle, located in $\mathbf{r}_{obs} = (0, 0)$, which we call $E_d(\sigma_1, \sigma_2)$ defined by eq. (6), and the other is the mean of the absolute value of the difference of position at the same

time step and we call it $E_t(\sigma_1, \sigma_2)$ defined by eq. (7).

$$E_d(\sigma_1, \sigma_2) = \frac{1}{6} \sum_{i=2,3,4,7,8,9} |\min_t(|\mathbf{r}_i^s(t) - \mathbf{r}_{obs}|) - \min_t(|\mathbf{r}_i(t) - \mathbf{r}_{obs}|)|, \quad (6)$$

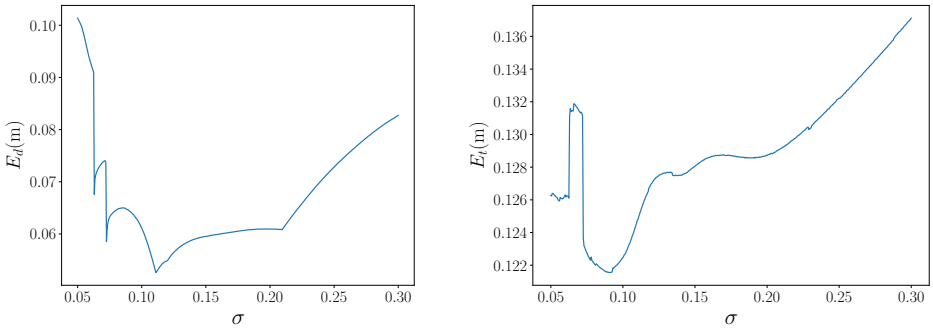
$$E_t(\sigma_1, \sigma_2) = \frac{1}{6} \sum_{i=2,3,4,7,8,9} \sum_{t \leq t_i^f} \frac{|\mathbf{r}_i^s(t) - \mathbf{r}_i(t)|}{t_i^f}. \quad (7)$$

First, we consider the simplified case in which the same spread value is used for both neural networks ($\sigma_1 = \sigma_2 \equiv \sigma$). This assumption can be made because the input space is the same for both networks and the spread is a measure of how many patterns are considered for estimating the output.

Figure 7 shows the results. The optimum spreads found were: $\sigma = 0.11$ for the distance-to-obstacle error $E_d = 0.05$ m, and $\sigma = 0.09$ for the frame-to-frame error $E_t = 0.122$ m.

These spread values provide us with a range of usable values in the proposed network. Interestingly, it contains $\sigma = < d_{fn} > = 0.09 \pm 0.06$, which is the mean distance between first neighbors (excluding points from the same trajectory) in the input space. Then, we can state that the GRNN's having better performance on the data are those that take into account the closest data points in the input space, with respect to the new input to be predicted.

Next, we relax the $\sigma_1 = \sigma_2$ constraint and explore this error as a function of the two variables. The heat map plot in Fig. 8 displays the minimum value of the error $E_t = 0.120$ m at $\sigma_1 = 0.08$ and $\sigma_2 = 0.11$. We can see that the decoupling of both GRNN's leads to a similar approximation of our experimental data. Again, the spread values obtained are comparable with the mean distance between first neighbors in the input space.



(a) (b)

Figure 7: Measures of the error for the simulated trajectories compared with the experimental ones as a function of the GRNN parameter σ. (a) Minimum distance to obstacle error: E_d (eq. 6). (b) Average microscopic difference between trajectories: E_t (eq. 7).

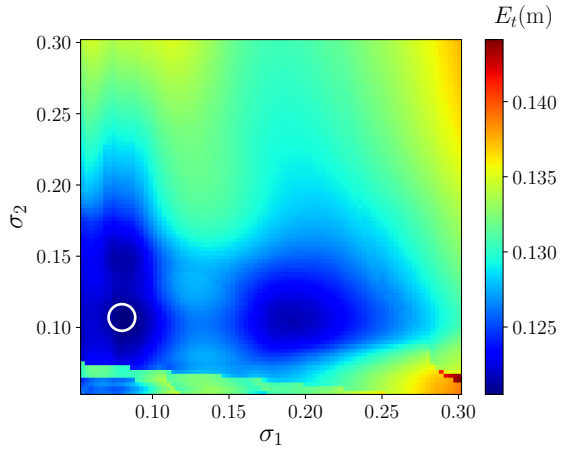


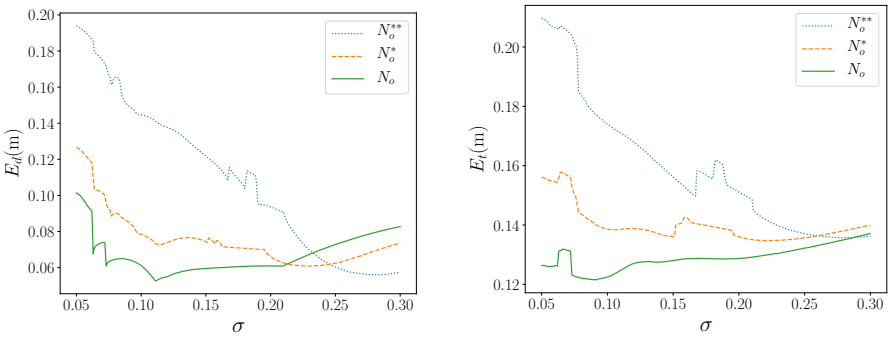
Figure 8: Average microscopic error: E_t (eq. 7) as a function of the spreads of both GRNN's. The white circle indicates the position of the minimum.

Because the simplification $\sigma_1 = \sigma_2 = \sigma$ works with an error similar to $\sigma_1 \neq \sigma_2$, in what follows we consider the case with only one free parameter of the GRNN.

3.3. A minimum data set

In this section we justify the number of trajectories chosen in previous sections for solving the data-driven simulation of the studied system.

First, we consider the $N_o = 10$ original trajectories and repeat the study performed in the previous section (Fig. 7), but considering subsets with a lower number of trajectories, $N_o^* = 8$ and $N_o^{**} = 6$. Here, we also consider as patterns but do not simulate the trajectories $\mathbf{r}_1$, $\mathbf{r}_5$, $\mathbf{r}_6$ and $\mathbf{r}_{10}$, thus from the remaining six trajectories we subtract two and four of them in order to get N_o^* and N_o^{**} correspondingly. In both cases, there are 15 ways of doing it: $\binom{6}{2} = \binom{6}{4} = 15$. So, we compute the corresponding 15 errors and their averages are shown in Fig.9.



(a) (b)

Figure 9: Measures of the error for the simulated trajectories as a function of the GRNN parameter σ for subsets experimental trajectories of different sizes. (a) Minimum distance to obstacle error: E_d. (b) Average microscopic difference between trajectories: E_t.

It can be seen that for smaller subsets of data trajectories, the error increases for a wide range of spread values. Thus, it is necessary to consider the set $N_o = 10$ that allows the best possible sampling in the input/output spaces.

Second, taking $\sigma = 0.11$, we study the error E_t in the rotated trajectories described in Sec. 2.2 and shown as dashed lines in Fig. 3. Here, the objective is to find the minimum number of rotated trajectories (N_r) in order to have an

error compatible with the original trajectories.

In order to do this, as data patterns for calibration and simulation, we take the two extreme trajectories $\mathbf{r}_1$ and $\mathbf{r}_{10}$ shown in Fig. 3 and N_r rotated copies of $\mathbf{r}_1$, thus conforming a set of $N_r + 2$ trajectories. The rotated trajectories were added one by one at equidistant angles from $\mathbf{r}_1$ and $\mathbf{r}_{10}$. Then the error E_t is computed with the leave-one-out cross-validation over these $N_r + 2$ trajectories, and it is shown in (Fig. 10).

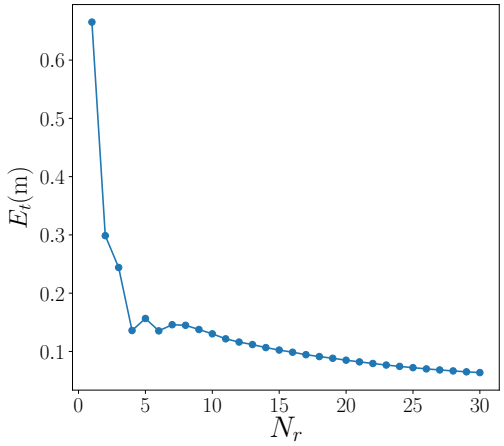


Figure 10: Average microscopic error: E_t as a function of the number of rotated trajectories.

It is natural to expect that the error will decrease when the number of trajectories increases. However, we can appreciate that for small N_r the error drops dramatically and then stabilizes presenting a slow decay. We choose a value of $N_r = 8$ in the transition zone, taking also into account that the error E_t is compatible with the error of original trajectories N_o.

3.4. Results

With the spreads found in Sec. 3.2 and the trajectory data described in Sec. 2.2 and 2.3, we analyze the performance of the simulations using the proposed method described in Sec. 3.1.

To this end, we simulate new trajectories for one particle having forty-eight different initial positions, shown as points in Fig. 11, around the final target at $\mathbf{T}_i = (3\text{ m}, 0)$. Also, a fixed obstacle is placed at coordinates $(x, y) = (0, 0)$ in a configuration similar to the experiments. However, neither of these trajectories is equal to the experimental ones. In all cases, the initial velocity points toward $\mathbf{T}_i$ and has an initial speed of $v_i = 1.17$ m/s, which is the arithmetic mean of the initial speeds of all the experimental trajectories.

First, we present the results corresponding to the case in which both spreads are considered equal to $\sigma = 0.11$. In Fig. 11 the smoothness and continuity of the trajectories with respect to the initial positions can be seen. Also note that the minimum distances of the simulated trajectories from the obstacle ($< d_{min} > \sim 0.5\ m$) are similar to those from our experiments ($< d_{min} > \sim 0.6\ m$) and from other papers [22] ($< d_{min} > \sim 0.6\ m$), [23] ($< d_{min} > \sim 0.7\ m$, in this case the obstacle is an standing human).

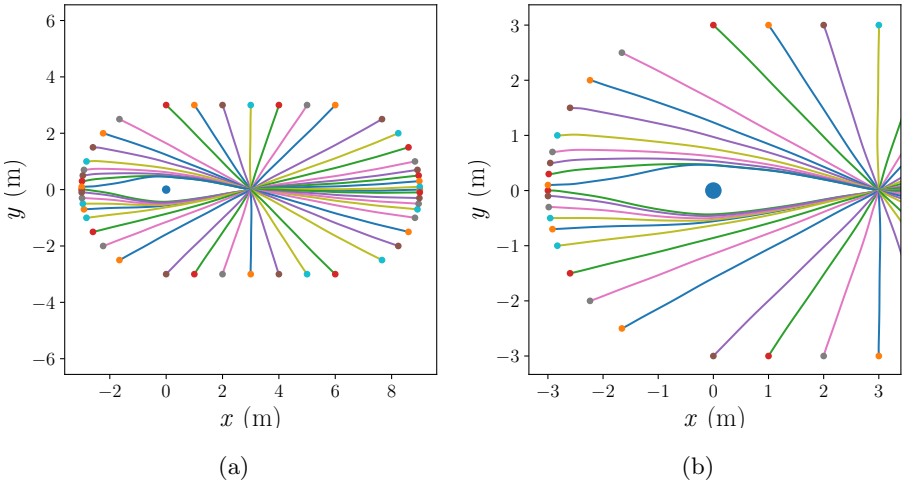


Figure 11: Simulated trajectories with $\sigma_1 = \sigma_2 = \sigma = 0.11$. (a) Complete view of the simulated system. (b) Zoom over the avoidance region.

It should be noted that only potentially colliding trajectories produce a detour for avoiding the obstacle, while the rest of the particles describe straight

trajectories toward the target. Also, if the obstacle were located in another position (at similar distance from the target), the trajectory patterns would rotate accordingly, because the input state and output action are defined in a coordinate system relative to the particle, i.e., polar coordinates taking the zero angle axes as the direction from the particle toward the target ($\mathbf{T}_i - \mathbf{r}_i(t)$). Thus, the results do not depend on the absolute position of the obstacle.

Second, we explore the same configuration, also simulating one particle at a time, but using a different spread value ($\sigma = 0.09$) corresponding to the minimum of the error E_t (Eq.7) as shown in Fig.7 (b). The results are presented in Fig. 12.

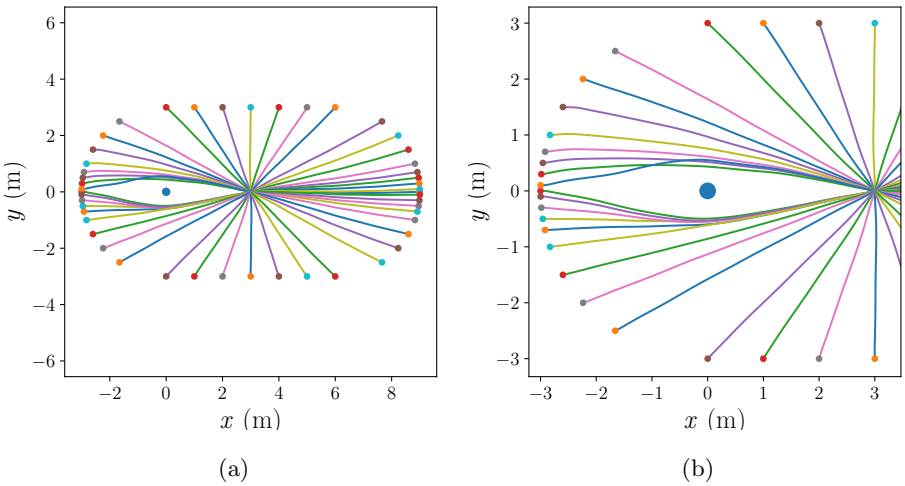


Figure 12: Simulated trajectories with $\sigma = 0.09$. (a) Complete view. (b) Zoom over the avoidance zone.

Also in this case, the simulated trajectories correctly describe the avoidance behavior. However, some differences can be observed; for example, there is one trajectory that slightly crosses over other neighbors' trajectories. This crossing is also observed in the experiments as is shown in Fig. 3.

An important consequence is that different values of σ can lead to different avoidance behaviors, which can be used for simulating a heterogeneous popu-

lation of virtual pedestrians. Of course, another way of doing this would be to use a different data set of trajectories for the GRNN of each simulated agent.

Animations of selected trajectories reported in this section can be seen in the supplementary material.

3.5. Extrapolation to more complex obstacles

In order to see whether our model has any prediction capacity, we stress the proposed methodology by simulating some configurations different from those of the experimental setup. In particular, we consider more complex obstacles composed of several simple ones. For these simulations we choose the variant of using $\sigma = 0.11$.

In Fig. 13 (a) a wall-like obstacle of length 1.4 m is avoided by the simulated particles. The wall is oriented along the direction of trajectories. And the simulated pedestrian considers the closest point over the wall in its field of view as the obstacle to be avoided.

Another configuration of a larger obstacle composed of three basic obstacles is presented in Fig. 13 (b). In this case, the simulated agent considers the closest obstacle and reacts in consequence, following the proposed method. Also here, it can be observed that the particles dodge the obstacle at reasonable distance.

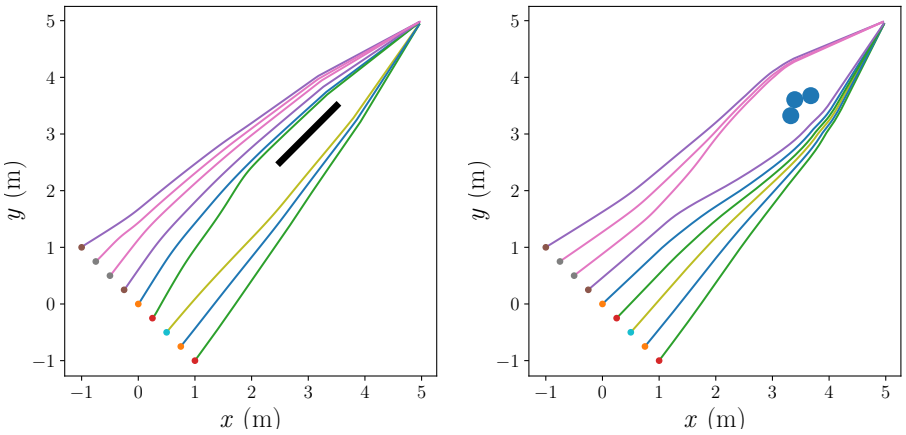


Figure 13: Simulated trajectories in previously unseen scenarios. (a) A wall-like obstacle. (b) A larger obstacle built from four basic obstacles

In both cases, it can be seen that the simulated particles describe natural trajectories when avoiding larger obstacles. It is also demonstrated that arbitrary rotation of the system does not affect the performance of the method.

3.6. Comparison with traditional modeling method

Finally, in this section we compare our data-driven approach with the traditional modeling procedure. To this end, we choose the predictive collision avoidance model [24], which is a variation of the social force model conceived for producing better collision avoidance maneuvers. The direction of the "social force" term is given by the repulsion that would produce the future positions of particles at collision time. The benefits and details of this model can be seen in [24]. However, as the magnitude of the modified "social force" is described qualitatively in the cited paper, a particular form we implemented is presented in Fig. 14.

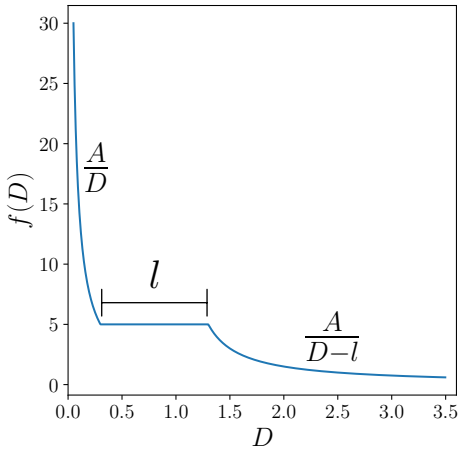


Figure 14: Magnitude of the avoidance force in the Karamouzas et al. 's predictive model [24].

This function has two free parameters that we used for minimizing the error at which the model can adjust the $N_o = 10$ original trajectories. The desired speed of the driving force was set as $v_d = 1.8\ m/s$, which is the mean speed of the experimental trajectories. The rest of the model parameters and specifications can be seen in the original work [24].

We calculated the frame-to-frame error defined in Eq. 7 considering each experimental trajectory and the corresponding simulated one having the same initial position and final target. The error E_t surface is displayed in Fig. 15.

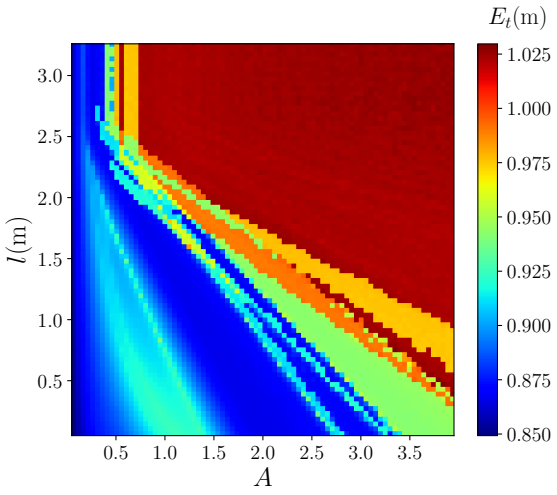


Figure 15: Error E_t as a function of both parameters of the repulsion force shown in 14 implemented in the avoidance model of Karamouzas et al. [24].

We can observe that the minimum error is near $E_t^{model} = 0.90\ m$, while in our approach it is $E_t^{Data-driven} = 0.12\ m$. One possible explanation is that the model cannot adjust the variation in the speed of the simulated particles according to the experimental ones, at least in the present experiment. If one wants to improve this, a new modification to the model could be proposed, which of course would bring new parameters. Also, it should be noted that the model already has more parameters than the two parameters used for computing the error in Fig. 15.

On the other hand, the proposed data-driven approach has a maximum of two parameters, and it also works fine with only one value of the parameter σ. Even if this traditional model could be modified or adjusted in order to reproduce the experimental data, we remark that the data-driven approach would save any effort in the modeling procedure and will provide (by definition) experimentally validated trajectories through efficient simulations.

4. Conclusions and perspective

The collision avoidance problem considering one moving pedestrian and one obstacle was studied with the technology from a motion capture laboratory allowing high-precision tracking of trajectories.

A data-driven model that uses a generalized regression neural network (GRNN) was proposed as a general method for simulating pedestrian dynamics. As a first approach, it was implemented with a simple configuration studied experimentally, consisting of one pedestrian avoiding a fixed obstacle.

The advantage of the GRNN is that it only has one free parameter and no training phase is needed, because the input/output patterns are used directly by this neural network.

The proposed simulation scheme allows us to reproduce the experimental data and generalize to other scenarios not explicitly contained in these data used to feed the GRNN. In this sense, the methodology proposed is invariant under rotations of the relative particle - obstacle positions. Thus we can claim that the data-driven simulation of the general problem of avoiding one narrow obstacle at large and medium distances has been solved.

We provide the model along with a range of spread values (the free parameter of the GRNN) and with the experimental data (input/output patterns, available in the supplementary material) ready to use by the community.

Our model is ready for considering more particles. Thus, in future work we will present results of navigation in more populated environments at medium and also at high densities, where contact and competitiveness could be present.

Acknowledgements

The authors acknowledge financial support via project PID2015-003 (Agencia Nacional de Promoción Científica y Tecnológica, Argentina; Instituto Tecnológico de Buenos Aires; Urbix Technologies S.A.) and from ITBACyT-2018-42 (Instituto Tecnológico de Buenos Aires). Also, we are grateful to Marcela

Guerrero for her invaluable collaboration in setting up the Motion Capture system and to the four volunteers of the experiments.

References

[1] A. Schadschneider, W. Klingsch, H. Klüpfel, T. Kretz, C. Rogsch, A. Seyfried, Evacuation dynamics: Empirical results, modeling and applications, Encyclopedia of complexity and systems science (2009) 3142–3176.

[2] D. C. Duives, W. Daamen, S. P. Hoogendoorn, State-of-the-art crowd motion simulation models, Transportation research part C: emerging technologies 37 (2013) 193–209.

[3] R. F. Martin, D. R. Parisi, Pedestrian collision avoidance with a local dynamic goal, in: Proceedings of the PED2018 Conference (in progress), 2018.

[4] D. Helbing, P. Molnar, Social force model for pedestrian dynamics, Physical review E 51 (5) (1995) 4282.

[5] D. Helbing, I. Farkas, T. Vicsek, Simulating dynamical features of escape panic, Nature 407 (6803) (2000) 487.

[6] W. Qian-Ling, C. Yao, D. Hai-Rong, Z. Min, N. Bin, A new collision avoidance model for pedestrian dynamics, Chinese Physics B 24 (3) (2015) 038901.

[7] M. Moussaïd, D. Helbing, G. Theraulaz, How simple rules determine pedestrian behavior and crowd disasters, Proceedings of the National Academy of Sciences 108 (17) (2011) 6884–6888.

[8] M. J. Seitz, N. W. Bode, G. Köster, How cognitive heuristics can explain social interactions in spatial movement, Journal of The Royal Society Interface 13 (121) (2016) 20160439.

[9] S. Kim, A. Bera, A. Best, R. Chabra, D. Manocha, Interactive and adaptive data-driven crowd simulation, in: 2016 IEEE Virtual Reality (VR), IEEE, 2016, pp. 29–38.

[10] M. Zhao, A clustering based approach for realistic and efficient data-driven crowd simulation, arXiv preprint arXiv:1506.04480.

[11] G. Kouskoulis, I. Spyropoulou, C. Antoniou, Pedestrian simulation: Theoretical models vs. data driven techniques, International Journal of Transportation Science and Technology 7 (4) (2018) 241–253.

[12] X. Song, D. Han, J. Sun, Z. Zhang, A data-driven neural network approach to simulate pedestrian movement, Physica A: Statistical Mechanics and its Applications 509 (2018) 827–844.

[13] X. Wei, W. Lu, L. Zhu, W. Xing, Learning motion rules from real data: Neural network for crowd simulation, Neurocomputing 310 (2018) 125–134.

[14] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, S. Savarese, Social lstm: Human trajectory prediction in crowded spaces, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 961–971.

[15] A. Tordeux, M. Chraibi, A. Seyfried, A. Schadschneider, Prediction of pedestrian speed with artificial neural networks, arXiv preprint arXiv:1801.09782.

[16] Y. Ma, E. W. M. Lee, R. K. K. Yuen, An artificial intelligence-based approach for simulating pedestrian movement, IEEE Transactions on Intelligent Transportation Systems 17 (11) (2016) 3159–3170.

[17] Q. Wang, M. Chen, F. Nie, X. Li, Detecting coherent groups in crowd scenes by multiview clustering, IEEE transactions on pattern analysis and machine intelligence.

[18] Q. Wang, J. Gao, W. Lin, Y. Yuan, Learning from synthetic data for crowd counting in the wild, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 8198–8207.

[19] S. Haykin, Neural networks: a comprehensive foundation, Prentice Hall PTR, 1994.

[20] D. F. Specht, A general regression neural network, IEEE transactions on neural networks 2 (6) (1991) 568–576.

[21] J. Park, I. W. Sandberg, Universal approximation using radial-basis-function networks, Neural computation 3 (2) (1991) 246–257.

[22] X. Jia, C. Feliciani, D. Yanagisawa, K. Nishinari, Experimental study on the evading behavior of individual pedestrians when confronting with an obstacle in a corridor, arXiv preprint arXiv:1905.06173.

[23] D. R. Parisi, P. A. Negri, L. Bruno, Experimental characterization of collision avoidance in pedestrian dynamics, Physical Review E 94 (2) (2016) 022318.

[24] I. Karamouzas, P. Heil, P. Van Beek, M. H. Overmars, A predictive collision avoidance model for pedestrian simulation, in: International Workshop on Motion in Games, Springer, 2009, pp. 41–52.