Journal of Statistical Mechanics:
Theory and Experiment

PAPER

How update schemes influence crowd simulations

To cite this article: Michael J Seitz and Gerta Köster *J. Stat. Mech.* (2014) P07002

Journal of Statistical Mechanics: Theory and Experiment
An IOP and SISSA journal

How update schemes influence crowd simulations

Michael J Seitz[1,2] **and Gerta Köster**[1]

[1] Department of Computer Science and Mathematics, Munich University of Applied Sciences, Lothstr. 4, 80335 München, Germany
[2] Department of Informatics, Technische Universität München, Boltzmannstr. 3, 85748 Garching, Germany

E-mail: m.seitz@hm.edu and koester@hm.edu

Received 20 January 2014
Accepted for publication 7 May 2014
Published 2 July 2014

Online at stacks.iop.org/JSTAT/2014/P07002
doi:10.1088/1742-5468/2014/07/P07002

Abstract. Time discretization is a key modeling aspect of dynamic computer simulations. In current pedestrian motion models based on discrete events, e.g. cellular automata and the Optimal Steps Model, fixed-order sequential updates and shuffle updates are prevalent. We propose to use event-driven updates that process events in the order they occur, and thus better match natural movement. In addition, we present a parallel update with collision detection and resolution for situations where computational speed is crucial. Two simulation studies serve to demonstrate the practical impact of the choice of update scheme. Not only do density-speed relations differ, but there is a statistically significant effect on evacuation times. Fixed-order sequential and random shuffle updates with a short update period come close to event-driven updates. The parallel update scheme overestimates evacuation times. All schemes can be employed for arbitrary simulation models with discrete events, such as car traffic or animal behavior.

Keywords: interacting agent models, traffic and crowd dynamics, discrete fluid models

J. Stat. Mech. (2014) P07002

1742-5468/14/P07002+17$33.00

Contents

1. Introduction

Discretization of time and space is an issue that has to be coped with at some level in every pedestrian simulation model. In cellular automata [4, 7, 15, 20, 25] the spacial discretization is determined by a rigid grid of cells and pedestrians are usually represented by occupied cells. Therefore, the maximum density, the shape of pedestrians and the number of movement directions are fixed. This is an advantage in terms of computational complexity but a disadvantage in terms of modeling flexibility. On the other hand, the simple spacial structure can also facilitate the calibration of parameters.

In force based models [10, 11, 17, 18, 39] pedestrians move in continuous space. Adding another behavioral aspect in these models means adding another force and/or manipulating the existing ones. The forces accelerate the pedestrians similar to molecular dynamics. This is mathematically described as a system of differential equations that has to be solved. Thus, time and space discretization in these models is a problem of numerical mathematics: One has to choose a simulation time step that ensures a sufficiently accurate solution, while different numerical schemes allow for different simulation time steps [21]. This also holds for other more novel models based on differential equations that describe movement in continuous time, like the Gradient Navigation Model [13].

Update schemes schedule how pedestrians are moved and thus they are a crucial issue with great practical importance. Technically speaking, it is the principal algorithm that governs the simulation process. In many cases the modification of the update scheme causes the model to behave differently and it has to be recalibrated. A parallel update scheme, for instance, has the objective of improving computational

doi:10.1088/1742-5468/2014/07/P07002

J. Stat. Mech. (2014) P07002

performance and should not influence the model's behavior and macroscopic simulation outcome, such as evacuation times.

In the following we describe update schemes for models, such as cellular automata, that update the state of the system by processing discrete events. That means movement is not simulated as a continuous process but as discrete steps. This can generally be achieved in two ways: the *time-slicing approach* moves forward in time by Δt and then updates all elements in the system; the *discrete-event simulation approach* processes the events at the simulation time they actually occur (see, e.g., [28]). We address the former with the fixed-order sequential update, the random shuffle update and parallel update and call the latter event-driven. In current pedestrian motion models based on discrete events, fixed-order sequential updates and shuffle updates are prevalent.

Our first objective is to demonstrate the influence of update schemes on the simulation outcome. In particular, we want to lay open the very undesirable dependency of the solution on the time-step size Δt in sequential update schemes. Secondly, we propose that the solution is the natural event-driven approach to simulation where time updates are dictated by the occurrence of events, rather than imposing an update time Δt. Thirdly, for the sake of computational speed, we propose a parallel update scheme that is robust and comes sufficiently close to the results of the event-driven update to be useful for practical applications with a very large number of pedestrians.

In the case of the parallel update, the scheme is not intended to be superior to the event-driven scheme in terms of model behavior, but to facilitate computation. The parallel update we discuss serves two purposes. First, it allows us to develop our argument on how update schemes alter models. Second, we investigate whether this update scheme will nonetheless allow us to compute acceptable results. For this, we compare it to the model that we argue to be the best: the event-driven update.

The fixed-order sequential update does not represent a meaningful approach to scheduling movement. However, it is the most straightforward one. Thus, it serves particularly well to stress the argument that the choice of update is a major model choice. The same holds true for the random shuffle update, which may prevent a systematic bias due to randomness, but does not represent realistic pedestrian behavior.

With regard to interpretation, every update scheme can be questioned in two ways: Does it result in natural pedestrian movement and does it represent actual natural processes? Here we argue that the event-driven update scheme represents behavior that is closest to the natural process. While event-driven simulation approaches are well established in general simulation theory [12, 28], and can also be found in traffic simulation [8], pedestrian models usually are not based on this concept.

We do not discuss which scheme fits experiments best, but which one best logically represents natural movement. We think that this is the most important part of the decision about which update scheme to select for a crowd simulation model. It motivated our investigation of the event-driven update scheme and the investigation in general. One has to decide which update scheme to use and then calibrate other parameters to match experimental data.

In section 2 we describe the two-dimensional pedestrian movement model, which we use to investigate the update schemes. In sections 3–5 we describe a fixed-order sequential update, the random shuffle update, the event-driven update and a parallel update with collision detection and resolution. In section 6 we investigate and compare

J. Stat. Mech. (2014) P07002

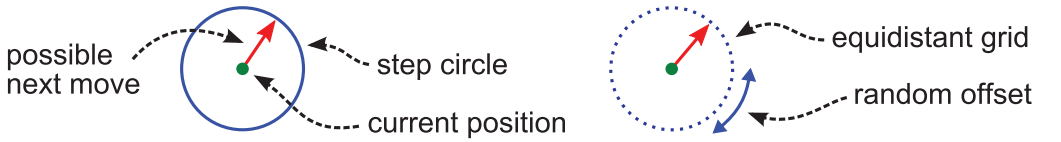

Figure 1. Schematic depiction of pedestrian movement in the Optimal Steps Model. On the left side the step circle and one possible movement is shown: pedestrians can move to any point on a circle with the step length as radius. On the right an equidistant grid is used to solve the optimization problem of finding the most attractive point on the circle: that is, the position with the highest utility. To prevent systematic errors in the movement, the grid is rotated by a random offset.

the update schemes quantitatively with two simulation scenarios and finally evaluate the results in section 7.

2. Pedestrian movement model

This section gives an outline of the Optimal Steps Model [31], the pedestrian movement model we use. The guiding idea is that natural stepwise movement of pedestrians is also the spacial discretization within the simulation. Although in reality the center of mass of pedestrians' bodies moves in a continuous manner, the movement decision is *where to make the next step*. We assume that pedestrians can step forward in any direction using a fixed step length. Thus, we obtain a step circle (see figures 1).

The stride length λ strongly depends on the speed of movement v [22, 31]. This can be taken into account by adapting the step length to the speed with a regression model. For normal walking speed we use the classical linear model

$$\lambda = \beta_0 + \beta_1 \cdot v + \epsilon \tag{1}$$

with regression coefficients β_0, β_1 and normally distributed error term $\epsilon \sim N(0, \sigma)$ depending on the standard deviation σ. The estimated coefficients are set to $\widehat{\beta_0} = 0.462$, $\widehat{\beta_1} = 0.235$ and the estimated standard deviation to $\hat{\sigma} = 0.036$, which is the result of empirical investigation [31].

The decision on where to make the next step on the circle is obtained by assigning a utility to every position in the plane. We employ two-dimensional functions that represent attraction or repulsion. High values represent high utility and therefore attraction. Low values represent low utility and are repulsive. Note that in the first publication of the model the term *potential* is used and higher values are more repulsive than lower values. Therefore, utility functions are the negative potentials. The former terminology can be misleading in our case, since the functions are not actually interpreted as potentials but rather as a measure of attractiveness for each position. Thus, we choose the concept of utility in this publication. However, the model is independent of the terminology discussion.

The attraction to the target is represented by the shortest travel time from the target to any point in the plane. Since the travel time has a positive value, we negate it to obtain higher utility for points closer to the target. The travel time is given by the

solution to the eikonal equation. An efficient way to obtain this solution is Sethian's fast marching algorithm [16, 32, 33]. Repulsive functions describe the distance pedestrians want to keep from each other or from obstacles. A constant, very low utility value is assigned to pedestrians' torsos and to obstacles. This ensures that pedestrians neither step on each other nor on obstacles. Pedestrian torsos are circular for simplicity.

With utility values at each position we can determine the point on the step circle with the highest utility. Solving this problem is a one-dimensional optimization task with periodic boundary conditions. An equidistant grid on the circle yields an approximate solution (see figure 1). In order to prevent systematic discretization errors, we add a random offset and thereby rotate the equidistant grid on the step circle. This way for every time step t we obtain angles

$$\varphi_{i,t}(k) = \frac{2\pi}{q}(k + u_{i,t}) \tag{2}$$

for every grid point $k = 1, \cdots, q$ on the circle around pedestrian $i = 1, \cdots, n$ given the uniformly distributed random offset $u_{i,t} \sim U(0, 1)$.

An extension to this concept is to not only search for the best position on the circle, but also within it. Then a two-dimensional optimization problem with boundary conditions has to be solved [36]. In this study we optimize on the circle with an equidistant grid that is randomly rotated for simplicity. Note that although we use the Optimal Steps Model for all simulation examples, the update schemes discussed are applicable for microscopic pedestrian simulation models with discrete events in general.

In force-based models space discretization is determined by time discretization: The position and the velocity are the solution of a differential equation. Thus, the accuracy of the numerical discretization determines the spatial resolution. Accuracy is achieved by sufficiently small—often very small—time steps. Therefore, the ideal is to have a time step as small as possible with a consequently also small spacial discretization. In cellular automata the spacial discretization is given by the grid and cannot be changed directly. In contrast to the Optimal Steps Model neither of the two other approaches to discretization match natural movement [31].

3. Fixed-order sequential and random shuffle update

The update scheme used for the original pedestrian simulation model [31] is a fixed-order sequential update. This approach has several advantages: It does not consume additional computation time for ordering pedestrians; collisions can be prevented; the order can be chosen to be suitable for some initial situation. The random shuffle update is only different to the fixed-order sequential update in one respect: the order is randomly permuted every time step. Therefore, it is a sequential update scheme with fixed time-step length as well.

For the fixed-order sequential update, the order is chosen according to the time of creation: pedestrians created the first move first. Therefore, a list data structure can be used in which newly created pedestrians are simply appended to the end of the list. In many scenarios pedestrians are created over time, rather than at once. Thus, pedestrians

J. Stat. Mech. (2014) P07002

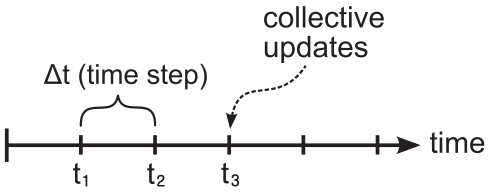

Figure 2. Unit clock for fixed-order sequential, random shuffle and parallel updates. All pedestrians are updated after a time step with fixed length Δt.

who have advanced furthest towards the target naturally move first. This behavior can be useful to avoid extensive blocking of pedestrians due to the update: Pedestrians at the front have already had the chance to move when pedestrians moving behind them make their steps. On the other hand this reasoning has its limitations, especially when it comes to scenarios with several sources at different places. Also, faster pedestrians overtaking others do not benefit from this order. However, in section 6.1 the fixed-order sequential update seems to have a clear advantage over the random shuffle update because the former seems to facilitate movement.

Another approach for fixed-order sequential updates could be the *frozen shuffle update* [2, 3]. With this update the order is randomly chosen once and not changed during the simulation. If the update order is not fixed, one can explicitly impose a spacial order [5, 6, 26], which is usually called a *backward-ordered updating scheme* [14]. In the following *fixed-order sequential update* means that pedestrians move in the order of their creation.

Contrary to fixed-order sequential updates, the random shuffle update randomly permutes the order each time step. This can be seen as a sequential update as well, but with random order. One advantage of this scheme could be that no systematic effects are introduced through the order of movement. The disadvantage, on the other hand, is that a random order does not provide a meaningful interpretation of real pedestrians' movements.

In both cases, the fixed-order sequential and the random shuffle scheme, the update itself is triggered by a unit time clock (see figure 2). Therefore, pedestrians are updated at equidistant points in time. The time step between two updates has length Δt and must be given as simulation parameter. After each time step all pedestrians are updated sequentially, given the fixed-order described above, or in random order for the random shuffle update. In terms of simulation time all pedestrians are updated at the same moment.

To generate different speeds of individuals, pedestrians only move when they are allowed to according to their desired speeds. For this, each pedestrian has an individual time credit τ, which is increased by the length of the simulation unit time step Δt at each update. The pedestrian is only moved when the time credit allows a whole movement step according to the desired speed and stride length. The time necessary for this step is discounted from the overall credit (see algorithm 1).

In the following, we use the term *event* for the moment in which a new movement begins. In the language of events, all events that have occurred in one unit time step are processed at the exact same simulation time after that time step (see figure 3). There is a certain similarity among update schemes: The fixed-order sequential update, the random shuffle update, the frozen shuffle update, as well as the parallel update have the same event handling in this respect.

J. Stat. Mech. (2014) P07002

Algorithm 1. Update algorithm for pedestrians with unit time step length Δt and individual time credit τ. Every pedestrian has an individual time credit τ, movement step length λ and desired speed v. This algorithm is applied for all pedestrians after each time step.

1: $\tau = \tau + \Delta t$
2: **if** $\tau \geqslant \lambda / v$ **then**
3: make movement step
4: $\tau = \tau - \lambda / v$
5: **end if**

4. Event-driven update

In this section we no longer process the events after a unit time clock step according to some fixed or random order. Instead we process the events in their natural order, that is, the way they occur [12]. Then the updates are not aligned to a fixed time grid in simulated time, but are processed at the point in the simulated time, when they happen. This way we obtain a natural order of events (see figure 4). In simulation theory this approach is also referred to as *discrete-event simulation* (see, e.g., [28], chapter 2).

Note that the order corresponds to the order of a fixed-order sequential or random shuffle update with time step length $\Delta t \to 0$ if algorithm 1 is used. This is what approximates natural behavior best in terms of updates: in a conflicting situation (see figure 5), the conflict is resolved by moving the pedestrian first who is allowed to move first according to his or her speed.

With this scheme no unit time clock or collective update is necessary. The implementation can be realized with a priority queue: For every pedestrian the time of the next movement step has to be computed and pedestrians are arranged in the order of increasing time of this event. Additional computation time in comparison to a fixed-order scheme is limited to the insertion of events into the chosen data structure representing the priority queue.

This scheme represents natural behavior: pedestrians are allowed to move as soon as they have finished the previous step. Due to this, conflicts are prevented in a natural way as well: the one who comes first also chooses the next position first. Others have to adapt when it is their turn to move. Therefore, we argue that this is the best approximation to real pedestrian conflict management for models without prediction of trajectories.

Given the natural behavior and the algorithmic simplicity of this scheme, we propose it as reference method. Model calibration and validation should also be done using the event-driven update scheme, while other schemes should be evaluated on the basis of their deviation from it. Hence, in the section on simulation experiments we investigate the parallel, sequential and shuffle update accordingly.

J. Stat. Mech. (2014) P07002

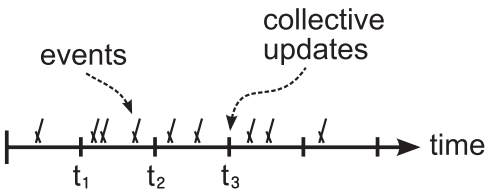

Figure 3. With fixed-order sequential, random shuffle and parallel updates all events that take place within one time step are processed at the same time, after the time step has finished. The order of how these events are processed is not the order of occurrence during the time step. The order is either fixed (fixed-order sequential update), randomly permuted (random shuffle update) or there is no order (parallel update) if they are processed simultaneously.

5. Parallel update

In this section we regard the event-driven update of the previous section as reference method. With the parallel update scheme itself we do not try to improve simulation results but reproduce as well as possible the results from the event-driven update scheme. This is because, in general, stepping behavior is asynchronous in the model as well as in reality. Nonetheless, a parallel update scheme can extensively improve computational performance and thus scalability of the system. If a simulation can be computed on parallel systems, real time requirements can be met more easily and large-scale case studies are facilitated or even made possible for the first time.

In social force models, or other models based on differential equations, the update is naturally parallel, because they are based on motion in continuous time. Therefore, actual parallelization is only a matter of implementation and not of model development. In simulations with discrete events, on the other hand, parallel processing is not possible without carefully designed algorithms that update pedestrians at the same time without changing the model's behavior.

Only non-conflicting events can be processed at the same time, which are events that do not depend on each other. For instance, two pedestrians at a great distance may be moved at the same time, since they do not have any influence on each other. On the other hand, if there are other pedestrians between them, they might influence each other indirectly through a chain of movement. Therefore, they cannot be regarded as non-conflicting. Determining the non-conflicting events is complex and requires computational resources.

Another approach is to adjust the model in a way that allows a parallel update of all pedestrians at the same time. This changes the model, hence it is important to revalidate to make sure that the impact on results is sufficiently small for them to still be of practical use. For cellular automata traffic models some approaches for parallelization can be found [14, 24, 29, 37].

At first we present a scheme that may yield collisions: that is, overlapping pedestrian torsos. Two steps are necessary to ensure conflict-free access to the data structure that holds all pedestrian positions: (a) pedestrians have to determine the next position without actually moving, (b) pedestrians move to the next position. If the pedestrians

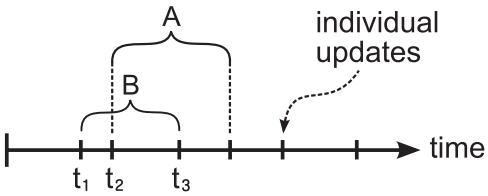

Figure 4. Event-driven updates. The events of making the next step are processed in their natural order of occurrence: pedestrians are moved, as soon as they are allowed to begin the next step. This way conflicting situations are resolved the way one can assume they would also be resolved in reality. In this figure pedestrian A needs more time for the movement step than pedestrian B.

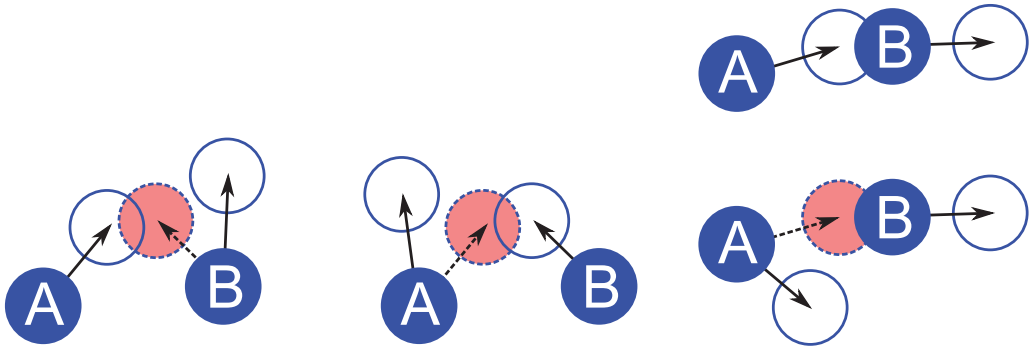

Figure 5. Two scenarios (1 on the left and 2 on the right) with conflict situations. Two pedestrians A and B in close proximity might choose different positions depending on who is updated first. In scenario 1 on the left, A moves first and occupies the position (the dashed circle filled with red) that would have been stepped on by B if he had moved first. On the other hand in the same scenario on the right if B moves first, A has to sidestep. In scenario 2 in the upper figure, B is allowed to move first and thereby gives way to the movement of A. But if A is allowed to move first, she has to sidestep, as is shown in the lower part.

moved to the next position immediately, they might perceive close pedestrians at their old position or new position based on arbitrary computer runtime factors. This is highly undesirable and therefore the positions have to be determined first, (a). After this step is completed, the pedestrians actually change positions, (b).

The approach discussed so far yields unstable results. For instance, two pedestrians might occupy very close positions with overlapping torsos. Due to the very low utility at their current positions they try to move away from each other in the next step. Subsequently they might again occupy the same position and so on (this behavior is elaborated in figure 6). The instability also hinders the pedestrian flow and thus we can expect slower crowd movement (see also section 6).

To resolve the collisions, we propose a conflict resolution mechanism based on the natural movement order. After the first two steps (a) of determining the next position and (b) moving to the next position, an analytic step (c) is performed. In this step, if pedestrians overlap the pedestrian who was supposed to move first according to the event-driven update remains at the new position. The other one is repositioned to where he was before (d). The previous position is always free, because other pedestrians

J. Stat. Mech. (2014) P07002

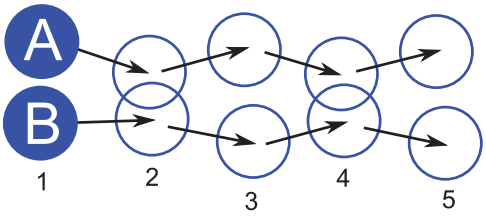

Figure 6. Possible movement using parallel updates without conflict resolution. In step one no collisions—that is, no overlapping torsos—are present. Because of independent movement parallel updates both pedestrians A and B choose a position in the middle of the path, which leads to overlapping. In the next step they both might choose positions further apart, because utility is very low whenever they are close together, and so on. This results in a rather unstable and unrealistic movement behavior.

Algorithm 2. Parallel update scheme with conflict resolution. Steps (a)–(e) must be computed in a sequential manner because of concurrency. Each of the steps is processed at the same time for all pedestrians, that is, with parallel computation. At first the next position for each pedestrian is determined if they are allowed to move according to their individual time credit τ (*a*). Then pedestrians are moved to this position (b). Following that, conflicts are diagnosed (c). In the next step conflicts are resolved by repositioning all but one of the conflicting pedestrians of each conflict group (d). Finally the time credit is updated (e).

```
1:  τ = τ + Δt
2:  if τ ⩾ λ/v then
3:      determine next position                          ▷ (a)
4:      move to next position                            ▷ (b)
5:      determine conflicts                              ▷ (c)
6:      reposition all conflicting pedestrians except for one   ▷ (d)
7:      if pedestrian has moved to a new position then
8:          τ = τ − λ/v                                  ▷ (e)
9:      end if
10: end if
```

perceived this position as occupied while determining the next position. Therefore, no overlapping pedestrians remain.

An important detail in this algorithm is the following: If pedestrians are repositioned, the time necessary for this step must not be discounted from their individual time credit τ. Therefore, another step (e) has to be performed: The final movement is determined and τ updated accordingly. Therefore, τ is reduced if and only if the pedestrian has moved to a new position after steps (a)–(d). The whole scheme is summarized in algorithm 2.

6. Comparative simulation experiments

In this section we compare simulation results for the three update schemes described previously. Parameters are chosen as in [31] and are not further discussed here. For the

Table 1. Summary of mean evacuation times (see also figures 8 and 9) in seconds and variance of evacuation times for the evacuation scenario investigated in section 6.1 (see figure 7). The scenario was run 500 times for each of the update schemes: sequential and shuffle update (see section 3) with $\Delta t = 0.5$ s and $\Delta t = 0.2$ s, event-driven update (see section 4) and parallel update with $\Delta t = 0.2$ s and collision detection and resolution (see section 5).

Update scheme	Mean (seconds)	Variance
sequential 500 ms	65.3	5.9
sequential 200 ms	61.8	4.7
event-driven	60.7	3.6
shuffle 500 ms	70.9	6.9
shuffle 200 ms	63.8	4.1
parallel	67.5	4.2

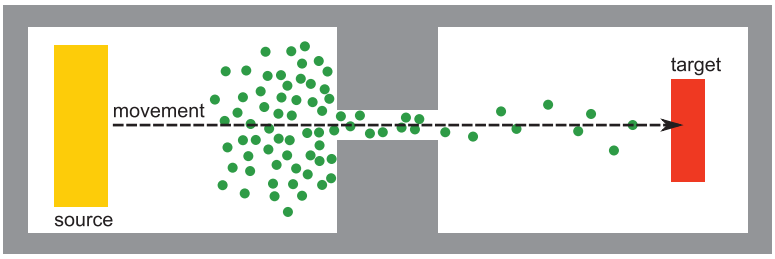

Figure 7. Evacuation scenario similar to the one investigated in [23]. Pedestrians move from the source (yellow) on the left to the right target (red) passing through a bottleneck. The bottleneck has width 1.2 m and length 3.9 m. The time measurement is taken when the last pedestrian reaches the target.

sequential and shuffle update scheme we use a unit clock update with $\Delta t = 0.5$ s and one with $\Delta t = 0.2$ s. This way differences stemming from the time step length can be observed. For the parallel update we use $\Delta t = 0.2$ s.

6.1. Evacuation of a room

The first simulation study is an evacuation of a room similar to a scenario that has been empirically investigated [23]. Here we focus on the comparison of evacuation times. Pedestrians are located in a room and have to pass through a bottleneck of width 1.2 m and length 3.9 m. After the bottleneck they have to pass through another room until they finally reach the target. The time measurement is taken as soon as all pedestrians have reached the target. The scenario is shown in figure 7.

The scenario was simulated 500 times for each update scheme. The resulting evacuation times are summarized in table 1 and figures 8 and 9. Welch's t-test yields very small p-values ($p < 0.0001$) comparing results across update schemes. This shows that all update schemes compared to each other yield significantly different evacuation times. Because of the small variance, the event-driven update could be seen as the most robust scheme.

J. Stat. Mech. (2014) P07002

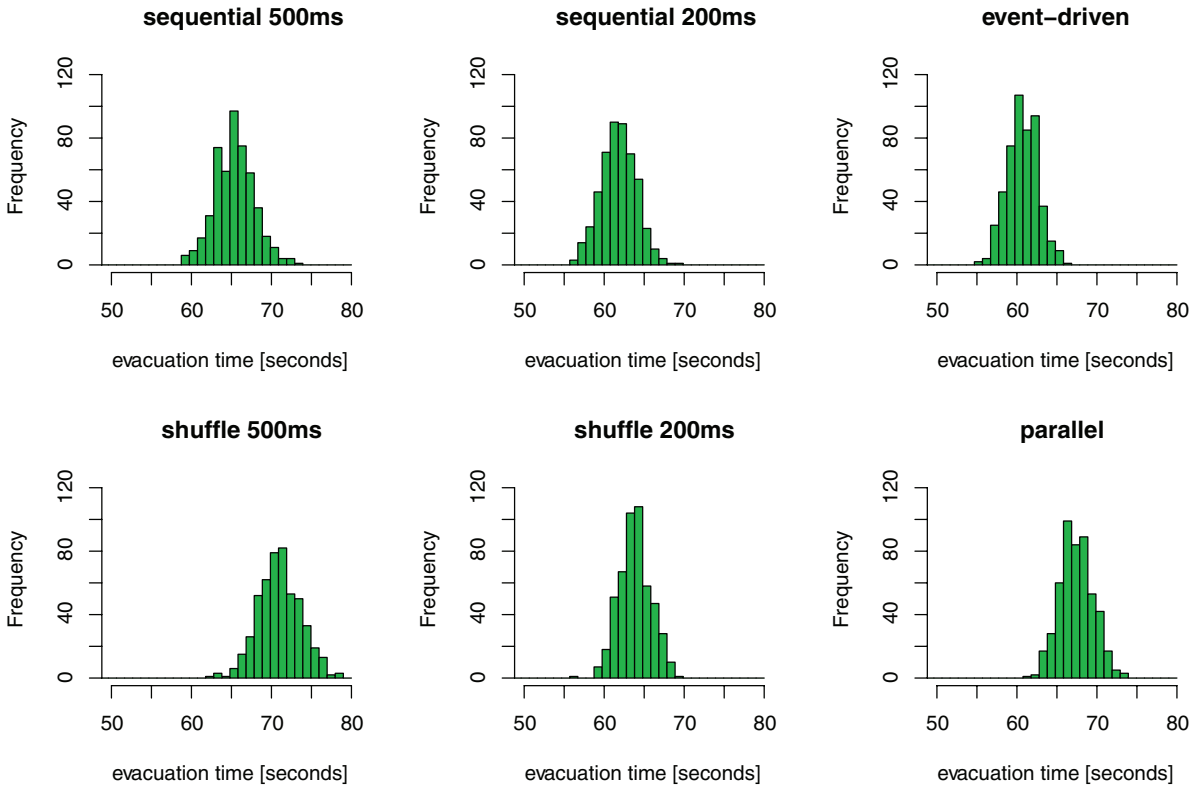

Figure 8. Mean evacuation times in a histogram for each update scheme (see table 1 and figure 9).

Both the fixed-order sequential update and the random shuffle update yield evacuation times close to the event-driven scheme, with smaller time steps Δt. The random shuffle update generally shows slower evacuations than the fixed-order sequential update, but the difference becomes less pronounced with $\Delta t = 0.2$.

We note that evacuation times differ significantly between update schemes. Therefore, update schemes change the model in a way that has an impact on macroscopic measures. The sequential and shuffle update yield results closer to the event-driven update if the time step Δt is reduced. This is as expected and should finally lead to almost identical results for very small unit clock time steps Δt.

6.2. Density-speed relationship

One of the most important properties in pedestrian streams is the density-speed relationship, or the equivalent density-flow relationship. Its basic properties should be matched by every pedestrian simulation. The most important one might be decreasing speed with higher densities and finally stagnation with very high densities. There are several empirical investigations [19, 34, 40, 41] but it seems evident that many different relationships exist, depending on the situation, location and culture of pedestrians [9].

Our objective in this investigation is to compare the update schemes rather than to calibrate the model according to empirical findings. A general method for calibrating to a given density-speed relationship has been presented [31]. Here we use the relationship given by Weidmann [38] as a point of reference. The random shuffle update is not

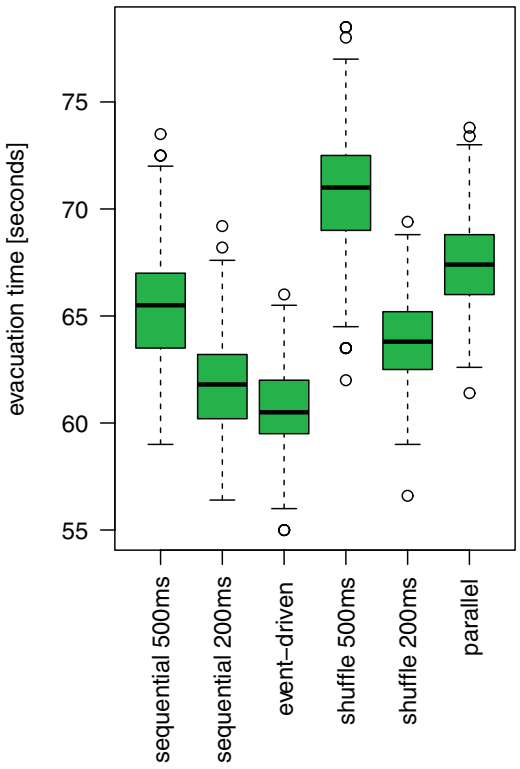

Figure 9. A box-plot (with standard parameters from the R-function boxplot [27]) summarizes the results for all update schemes. The plot is based on the same data as in figure 8. The event-driven update scheme yields the smallest evacuation times and the shuffle update scheme with $\Delta t = 0.5\ s$ the largest (see also figure 8 and table 1).

investigated any further in this part, since it shows similar behavior as the fixed-order sequential update scheme and yields evacuation times further away from the desired results of the event-driven update in section 6.1.

In the experimental setup for measuring the density and speed, pedestrians are located in a 4 m wide and 50 m long corridor. They walk from left to right. When they reach the end of the corridor they are relocated to the beginning. This gives us periodic boundary conditions[3]. The measurement is taken in the middle of the corridor at a 4 m wide and 10 m long measurement area. The current speed of each pedestrian is the length of the last movement step divided by the unit clock time step length Δt. Note that technically the event-driven update does not have a unit clock, but it is used for the measurement here. The density is measured with a Voronoi diagram [30, 35]: Each pedestrian has a personal space A_i, which is the Voronoi cell defined by the pedestrian and her neighbors. The individual density for pedestrian i is $1/A_i$.

Since all updates are based on discrete events, individual speeds are 0 every time the pedestrian is not allowed to move. On the other hand, measured speeds are overestimated in the next step when the pedestrian is allowed to move again (see figure 10, left). This is because, for the measurement, a full step of a pedestrian who is allowed to move is divided by a relatively small time step. Thus, averaging is necessary. Here

[3] Note that some additional measures must be taken in order to obtain truly periodic boundary conditions, which was done for this study.

J. Stat. Mech. (2014) P07002

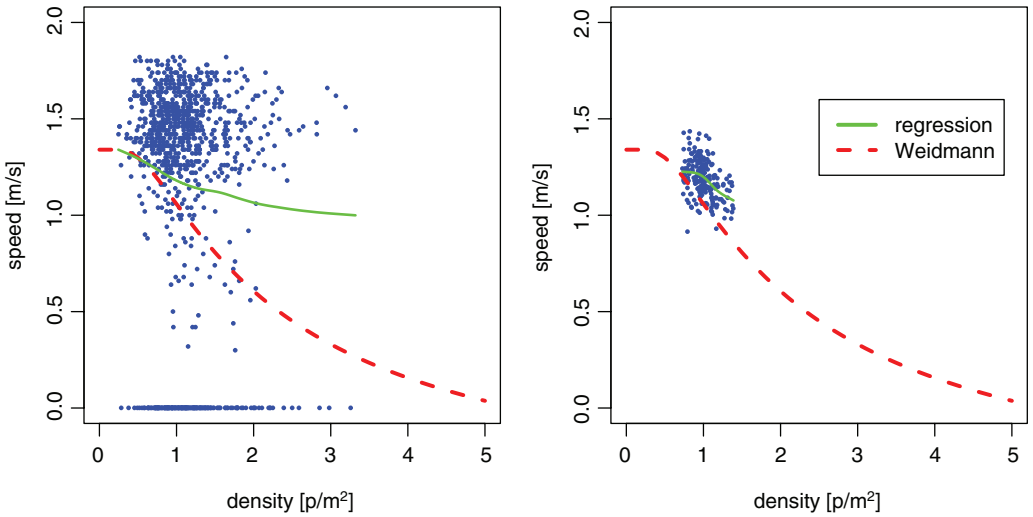

Figure 10. Illustration of the density-speed measurements taken for the event-driven update with 200 pedestrians in the corridor. Left: 1000 measurements of individual pedestrians within the measurement area. Independent of the density at many occasions the speed is 0 due to the discrete update. Pedestrians do not move because they do not have the necessary space or time credit τ. Right: all 8012 measurements averaged over the measurement area resulting in 200 points. Now the variation is much smaller and no measurements with speed 0 remain. The latter can then be used for a meaningful density-speed relationship (see also figure 11).

we average over space and *not* time: All individual speeds within the measurement area defined above are averaged (see figure 10, right). For the density within the measurement area we employ the formula [35]

$$D = \frac{N}{\sum_{i=1}^{N} A_i} \tag{3}$$

with N being the number of pedestrians within the area. To cover the whole range of possible densities, different numbers of pedestrians are created in the corridor (from 40 up to 700). Figure 11 on the left shows the relationship for the event-driven update with one point for each measurement and a spline regression.

In figure 11, on the right, the update schemes for all fundamental diagrams are compared. As before, the event-driven and the sequential update with $\Delta t = 0.2$ s produce very similar results. The parallel update again yields the smallest speed and the sequential update with $\Delta t = 0.5$ s lies in between. The difference is greatest with medium densities, that is at about 2 p/m^2. This can be explained by the number of conflicting situations. With low densities all pedestrians are free to move without having other pedestrians in their way. With other pedestrians nearby conflicting situations occur; that is, it matters who moves first. This is exactly what is determined by the update scheme and therefore the differences are most pronounced. Finally, very high densities prevent movement in general. Therefore, it does not matter who would have been allowed to move first, if no one is moving at all.

J. Stat. Mech. (2014) P07002

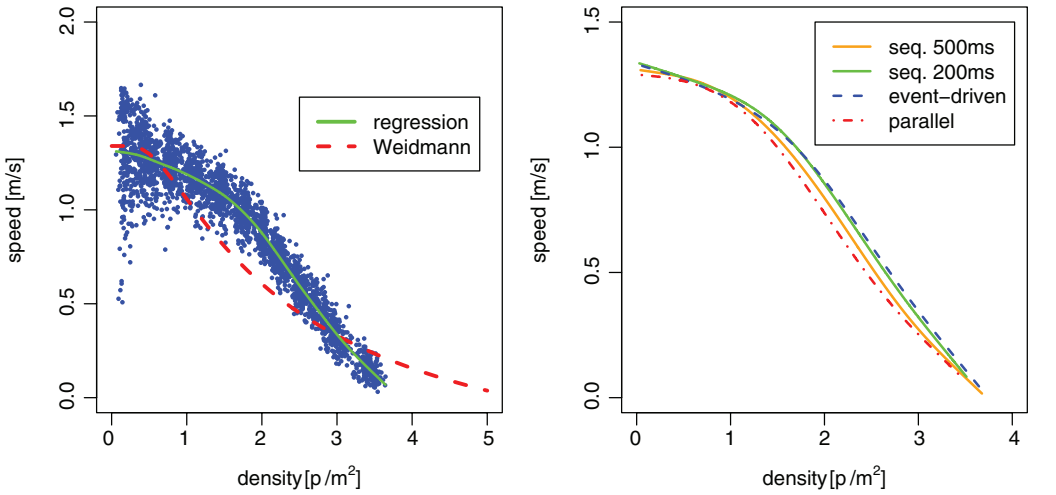

Figure 11. Density-speed relations for different update schemes. Left: relationship for the event-driven update compared to the curve reported by Weidmann [38]. Each point is one measurement averaged for all pedestrians within the measurement area (see also figure 10). At low densities scatter is always higher due to less pedestrians being located within the measurement area at the same time. Right: comparison of spline regressions for all four update schemes. The curves for the event-driven and sequential update with $\Delta t = 0.2$ s are almost identical. As in the evacuation scenario in section 6.1 the parallel update produces the slowest movement.

7. Conclusion

We presented update schemes for a simulation model with discrete events: the fixed-order sequential update, the random shuffle update, the event-driven update, and a parallel update with collision detection and resolution. We argued that the event-driven update should be regarded as reference method, since the order of movement captures the natural order. Real pedestrians perceive motion and not only positions [1], which is not captured by the model. But within the boundaries of the simulation model, the event-driven update represents the most natural way of treating concurrent movement.

The methods were investigated through two simulation studies, which demonstrated the influence of update schemes: a bottleneck scenario with evacuation times as measure, and a corridor with periodic boundary conditions for density-speed relationships. Both studies led to the conclusion that the event-driven update allows for the fastest movement of a crowd and can be approximated by a fixed-order sequential or random shuffle update with small Δt. The parallel update seems to handicap smooth movement and therefore causes slower movement of the crowd, especially at medium densities. This is because pedestrians choose positions they would not have chosen if they had anticipated the movement of another pedestrian. This is improved in the event-driven update, since pedestrians have all possible information available at the point in time when they are moving.

One can use the parallel update scheme for scenarios with high performance requirements. Because of the difference in results between the parallel update scheme and the event-driven update scheme, it might be necessary to further calibrate the model to

J. Stat. Mech. (2014) P07002

obtain the same results as for the event-driven update. For instance, one could increase the desired speed pedestrians want to walk in order to even out slower evacuation times. However, we propose to use the event-driven update whenever computation time is not critical.

The parallel algorithm yields an approximation for the reference method that is collision free. Since the egress time with the parallel update is systematically overestimated, it can also be used to determine a reasonable upper boundary. Furthermore, with smaller time steps Δt the parallel update approaches the behavior of the event-driven update, because conflicting situations become less with decreasing time steps. Here, the same argument holds as for the fixed-order sequential and random shuffle update.

We suggest to use either the event-driven scheme (for accuracy) or the parallel scheme (for computational efficiency) depending on the requirements. The frozen shuffle update, the random shuffle update and the fixed-order sequential update, on the other hand, should be dismissed for practical application in pedestrian simulations.

The update schemes we presented can be used for any pedestrian stream simulation that is based on discrete events, like cellular automata. Furthermore, simulations based on discrete events for other applications, like for car traffic or animal behavior, can make use of the presented mechanisms.

Acknowledgments

This work was partially funded by the German Federal Ministry of Education and Research through the project MEPKA on mathematical characteristics of pedestrian stream models (Grant No. 17PNT028) and the project MultikOSi on assistance systems for urban events—multi criteria integration for openness and safety (Grant No. 13N12824). The authors gratefully acknowledge the support by the Faculty Graduate Center CeDoSIA of TUM Graduate School at Technische Universität München, Germany.

References

[1] Albright T D and Stoner G R 1995 Visual motion perception *Proc. Natl Acad. Sci.* **92** 2433–40
[2] Appert-Rolland C, Cividini J and Hilhorst H J 2011 Frozen shuffle update for a deterministic totally asymmetric simple exclusion process with open boundaries *J. Stat. Mech.* P10013
[3] Appert-Rolland C, Cividini J and Hilhorst H J 2011 Frozen shuffle update for an asymmetric exclusion process on a ring *J. Stat. Mech.* P07009
[4] Blue V J and Adler J L 2001 Cellular automata microsimulation for modeling bi-directional pedestrian walkways *Transport. Res.* B **35** 293–312
[5] Brankov J G, Papoyan V V, Poghosyan V S and Priezzhev V B 2006 The totally asymmetric exclusion process on a ring: exact relaxation dynamics and associated model of clustering transition *Physica* A **368** 471–80
[6] Brankov J G, Priezzhev V B and Shelest R V 2004 Generalized determinant solution of the discrete-time totally asymmetric exclusion process and zero-range process *Phys. Rev.* E **69** 066136
[7] Burstedde C, Klauck K, Schadschneider A and Zittartz J 2001 Simulation of pedestrian dynamics using a two-dimensional cellular automaton *Physica* A **295** 507–25
[8] Charypar D, Axhausen K W and Nagel K 2007 Event-driven queue-based traffic flow microsimulation *Transport. Res. Rec.: J. Transport. Res. Board* **2007** 35–40
[9] Chattaraj U, Seyfried A and Chakroborty P 2009 Comparison of pedestrian fundamental diagram across cultures *Adv. Complex Syst.* **12** 393–405
[10] Chraibi M, Kemloh U, Schadschneider A and Seyfried A 2011 Force-based models of pedestrian dynamics *Netw. Heterog. Media* **6** 425–42

[11] Chraibi M, Seyfried A and Schadschneider A 2010 Generalized centrifugal-force model for pedestrian dynamics *Phys. Rev.* E **82** 046111
[12] Conway R W, Johnson B M and Maxwell W L 1959 Some problems of digital systems simulation *Manag. Sci.* **6** 92–110
[13] Dietrich F and Köster G 2014 Gradient navigation model for pedestrian dynamics *Phys. Rev.* E **89** 062801
[14] Evans M R 1997 Exact steady states of disordered hopping particle models with parallel and ordered sequential dynamics *J. Phys. A: Math. Gen.* **30** 5669
[15] Ezaki T, Yanagisawa D, Ohtsuka K and Nishinari K 2012 Simulation of space acquisition process of pedestrians using proxemic floor field model *Physica* A **391** 291–9
[16] Hartmann D 2010 Adaptive pedestrian dynamics based on geodesics *New J. Phys.* **12** 043032
[17] Helbing D, Farkas I and Vicsek T 2000 Simulating dynamical features of escape panic *Nature* **407** 487–90
[18] Helbing D and Molnár P 1995 Social force model for pedestrian dynamics *Phys. Rev.* E **51** 4282–6
[19] Jelić A, Appert-Rolland C, Lemercier S and Pettré J 2012 Properties of pedestrians walking in line: fundamental diagrams *Phys. Rev.* E **85** 036111
[20] Kirchner A, Klüpfel H, Nishinari K, Schadschneider A and Schreckenberg M 2003 Simulation of competitive egress behavior: comparison with aircraft evacuation data *Physica* A **324** 689–97
[21] Köster G, Treml F and Gödel M 2013 Avoiding numerical pitfalls in social force models *Phys. Rev.* E **87** 063305
[22] Kuo A D 2001 A simple model of bipedal walking predicts the preferred speed-step length relationship *J. Biomech. Eng.* **123** 264–9
[23] Liddle J, Seyfried A, Steffen B, Klingsch W, Rupprecht T, Winkens A and Boltes M 2011 Microscopic insights into pedestrian motion through a bottleneck, resolving spatial and temporal variations arXiv:1105.1532:v1
[24] Nagel K and Schleicher A 1994 Microscopic traffic modeling on parallel high performance computers *Parallel Comput.* **20** 125–46
[25] Nagel K, Wolf D E, Wagner P and Simon P 1998 Two-lane traffic rules for cellular automata: a systematic approach *Phys. Rev.* E **58** 1425–37
[26] Poghosyan V S and Priezzhev V B 2008 The relaxation dynamics of the tasep with particle-dependent hopping probabilities on a ring *Rep. Math. Phys.* **61** 239–46
[27] R Development Core Team 2013 R : a language and environment for statistical computing (www.r-project.org) version 3.0.2
[28] Robinson S 2004 *Simulation: the Practice of Model Developmentand Use* (New York: Wiley)
[29] Schadschneider A and Schreckenberg M 1993 Cellular automation models and traffic flow *J. Phys. A: Math. Gen.* **26** L679
[30] Schadschneider A and Seyfried A 2011 Empirical results for pedestrian dynamics and their implications for modeling *Netw. Heterog. Media* **6** 545–60
[31] Seitz M J and Köster G 2012 Natural discretization of pedestrian movement in continuous space *Phys. Rev.* E **86** 046108
[32] Sethian J A 1996 A fast marching level set method for monotonically advancing fronts *Proc. Natl Acad. Sci.* **93** 1591–5
[33] Sethian J A 1999 *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision and Materials Science* (Cambridge: Cambridge University Press)
[34] Seyfried A, Steffen B, Klingsch W and Boltes M 2005 The fundamental diagram of pedestrian movement revisited *J. Stat. Mech.* P10002
[35] Steffen B and Seyfried A 2010 Methods for measuring pedestrian density, flow, speed and direction with minimal scatter *Physica* A **389** 1902–10
[36] von Sivers I and Köster G 2014 How stride adaptation in pedestrian models improves navigation arXiv:1401.7838:v1
[37] Wahle J, Neubert L, Esser J and Schreckenberg M 2001 A cellular automaton traffic flow model for online simulation of traffic *Parallel Comput.* **27** 719–35
[38] Weidmann U 1992 Transporttechnik der Fussgänger *Schriftenreihe des IVT* vol 90) 2nd edn (Zürich Institut für Verkehrsplanung, Transporttechnik, Strassen- und Eisenbahnbau (IVT ETH)
[39] Yu W J, Chen R, Dong L Y and Dai S Q 2005 Centrifugal force model for pedestrian dynamics *Phys. Rev.* E **72** 026112
[40] Zhang J, Klingsch W, Schadschneider A and Seyfried A 2011 Transitions in pedestrian fundamental diagrams of straight corridors and t-junctions *J. Stat. Mech.* P06004
[41] Zhang J, Klingsch W, Schadschneider A and Seyfried A 2012 Ordering in bidirectional pedestrian flows and Its influence on the fundamental diagram *J. Stat. Mech.* P02002

J. Stat. Mech. (2014) P07002