The EvacSim Pedestrian Evacuation Agent Model: Development and Validation

Seán Óg Murphy[12], Kenneth N. Brown[2], Cormac Sreenan[1]

1. Mobile and Internet Systems Laboratory, 2. Cork Constraint Computation Centre,
Department of Computer Science, University College Cork, Ireland
jmm3@student.cs.ucc.ie, k.brown@cs.ucc.ie, cjs@cs.ucc.ie

Keywords: evacuation simulation, model evaluation, pedestrian, emergency, real time

Abstract

EvacSim is a multi-agent building evacuation simulation featuring pedestrian occupant agents occupying two-dimensional continuous free-space to simulate pedestrian egress for building evaluation and evacuation planning support. We detail pedestrian model elements that govern microscopic agent movement such as personal space preservation, obstacle avoidance and moving together as a crowd. We validate the EvacSim pedestrian model against real-world pedestrian data, comparing flow rates, density and velocity for corridor entry and for merging crowds, and we demonstrate that EvacSim simulations are consistent with the real-world data.

1. INTRODUCTION

EvacSim (Figure 1) is a simulation tool developed in Java that models pedestrian evacuation of buildings for evaluation of building designs, evacuation planners and emerging emergency response techniques [1,2]. EvacSim models space in two dimensions with agents occupying continuous free space, updating each agent periodically (once per "tick" of simulation time). In this paper we describe the requirements of simulation, the model design choices taken to accommodate these requirements, and the validation of this model against real-world pedestrian movement data.

An important feature of the EvacSim model is that complexity is kept low by performing relatively simple atomic actions. For example, rather than re-computing agent positions to maximize agent personal space in each tick, our agents make a series of small positional adjustments over the course of multiple ticks, with stable positioning arising naturally as a consequence.

Agents preserve personal space, avoid obstacles collisions, and produce queuing and congestion behaviour which is a key requirement for building and planner evaluation. We describe experiments performed to evaluate the model in terms of the pedestrian throughput in various bottleneck widths and compare these results with real-world pedestrian experiments from the literature, demonstrating a close correspondence between EvacSim pedestrian movement and real behaviour. We also investigate the behaviour of crowds merging in building evacuation, comparing EvacSim's model with real-world t-junction results and we show how an improvement to the braking behaviour of the agents improves the model realism.

This work is funded by HEA as part of the PRTLI-4 NEMBES project, and by Science Foundation Ireland as part of the ITOBO project under grant No. 07.SRC.I1170.

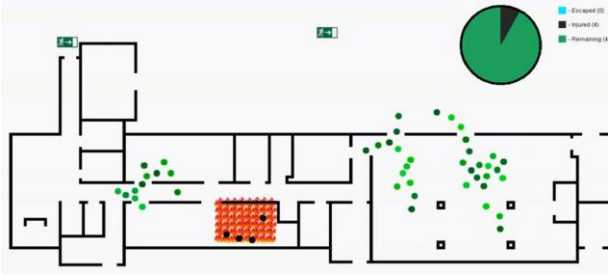


Figure 1- Screenshot of an EvacSim simulated evacuation

2. REQUIREMENTS

Simulation of evacuation requires an occupant model that accurately reflects the behaviour and movement of building occupants in order to properly reproduce key evacuation metrics such as congestion data and evacuation time. Furthermore, in application scenarios involving real-time response systems such as dynamic evacuation planning, there is a strong requirement for efficient, faster-than-realtime simulation to produce predictive results to assist decision making. This requirement necessitates a balance of realism and computational simplicity in order to provide useful simulation results under the time constraints. In multi-agent simulation, the overall complexity of the simulation is a product of the individual complexity of constituent agents. As such, for timely simulation we require individual agent computation to be kept low while still providing the degree of realism required to reproduce real pedestrian behaviour.

3. RELATED WORK

A number of model types for pedestrian egress exist, notably Cellular Automata (CA) models[3-6], Graph models[4,7] and multi-agent models[8-10]. While CA and Graph models have a low computational complexity, they are somewhat abstracted from the reality of the evacuation

process and building structure, and can underestimate congestion, the impact of merging flows of pedestrians, interaction of pedestrians and their use of building space. Multi-agent models have greater realism but this comes at the expense of computational complexity. While EvacSim makes use of a free-space multi-agent model, we designed our pedestrian movement model to simplify computation while maintaining realistic congestion and throughput characteristics.

In developing our pedestrian simulation model we take inspiration from the Social Forces model proposed by Helbing[11] and flocking techniques such as those described by Reynolds[12] and Olfati-Saber[13] to achieve a balance of realism with computational efficiency. Helbing[11] describes a detailed model for modeling pedestrians in terms of the forces acting on the pedestrians to preserve personal space, avoid collisions, and head towards exits. These forces are modeled as attractive and repulsive forces acting on the agent. The FDS-Evac simulation developed by Korhonen et al[9] is an implementation of the social forces model, integrated with the FDS Fire Dynamics Simulator. The EvacSim model accounts for similar pedestrian behaviour as the Helbing Social Forces model but reduces model complexity by having occupant agents compute simpler adjustments regularly which, over time, produce similar phenomena.

A variety of real-world pedestrian movement experiments exist in the literature. These experiments typically consist of tracking participant movement through bottlenecks of various widths, monitoring the rate of throughput and mapping this to the corridor width. In our work we performed experiments comparing EvacSim bottleneck throughput characteristics with real-world results from Seyfried[14], Kretz[15], Mueller[16] and Nagai[17]. We also compare with results from the t-junction experiment work of Zhang et al [18]. This approach to validation is similar to the approaches taken by Tavars[19], Heliovaaraa[20], Schadschneider[21] and Wagoum[8] in validation of their pedestrian simulation models.

In our previous work we investigated applications of faster than realtime simulation in near-future emergency outcome prediction[1] and decomposition of simulation space for highly scalable, multiple-future simulation[2]. As part of our work in [1] we investigated the simulation speed of EvacSim for a variety of population densities, demonstrating the capability of EvacSim for simulation of evacuation in much faster-than-realtime timeframes.

4. MODEL DETAILS

EvacSim models the world in 2-dimensional free space, representing distance in simulation "units". Time is represented as simulation "ticks"; in each tick of the simulation, each of the simulation elements (occupant agents, sensors etc) are updated according to their function or behaviour. Agents are represented in the world by 10-unit diameter discs and each agent possesses a motion vector combined with a 2-dimensional (x,y) coordinate pair which governs its moment-to-moment movement. The maximum amount of forward movement a given agent can make is taken from a standard distribution of maximum speeds, and the amount of actual forward movement performed tick-to-tick is governed by this value and the braking procedures of the movement model. Agent decision making processes modify this vector periodically, directing it towards a goal or adjusting it to avoid collisions depending on the agent high-level behaviour model and agent observations. Agents select visible goals to move towards as part of their decision-making behaviour by reasoning about a graph model of the building space; details of this higher-level decision-making behaviour are outside the scope of this paper.

Table 1 - EvacSim Variables

Variable Name	Description
S	Amount of space preserved around agent if possible (in simulation units)
ς (i)	Factor determining the rate of personal space adjustment agent i makes from tick-to-tick (in simulation units)
V(i)	The maximum distance agent i can traverse in a single tick (in simulation units)
V'(i)	The distance the agent i will traverse in the next tick, determined by Vi times η (i) (in simulation units)
φ(i)	Vector for agent i with a direction angle in degrees
pos(i)	2D floating point coordinate pair representing agent i's position in space
goal(i)	2D floating point coordinate pair representing the agent i's current goal
θgoal	Amount of adjustment permitted to SteeringVector angle each tick when pointing towards goals, in degrees
SD	Distance in front of the agent to check for obstacles, in simulation units
θevade	Amount of evasive action taken per-tick when avoiding obstacles, in degrees
η (i)	Floating point decimal between 0-1. Used with V(i) to determine V'(i) value
ω (i)	The radius of the agent i (in simulation units).
CD	Distance in front the agent to check for collision with other agents and obstacles (in simulation units)
θbrake	Angle cone in front of the agent in which upcoming collisions are considered (in degrees)

The agents attempt to preserve personal space around them, and avoid collisions with other agents by braking and steering to avoid them, similar to the social forces model

described by [Helbing,11]. Each agent possesses a number of variables (Table 1) governing the parameters of the movement behaviour, and a set of global constants dictate the parameters of general simulation, such as the amount of personal space agents will seek, or the distance in front of agents where collision avoidance is considered. In this validation, 20 simulation ticks corresponds to 1 second of simulated time, and 15 units corresponds to 1 metre of space. With this conversion rate, agent maximum speeds fall in the range 1.5-2.5m/s.

4.1. Agent Tick procedure

Each agent is called upon, once per simulation tick, to update its position based on personal space, collision avoidance and progress towards its goal. This procedure operates as follows:

Agent Tick procedure

1. Personal Space adjustment
2. Direct φ(i) towards goal(i)
3. Adjust φ(i) to steer around obstacles
4. Adjust pos(i) to sidestep obstacles
5. Apply braking factor to avoid collisions with other agents
6. Move forward V'(i) units, in φ(i) direction

4.2. Maintaining Personal Space

Accuracy of the agent model requires a mechanism for the agents to ensure that they are not overlapping each other. Typical behaviour for humans is to avoid close contact with other occupants if there is space to do so, maintaining a gap between other occupants, or *personal space*.

We achieve this by performing periodic checks to determine if there are any agents nearby that are within a threshold distance (**S**) and for the agent to adjust its position to move away from the closest infringing occupant (movement amount based on **ς (i)**). Through this mechanism of repeated observation and position adjustment, a compressed group of agents should spread out to cover an area. An agent is considered to be violating another's personal space if the distance between the centres of the two agents is less than the sum of the radii) plus **S**.

Having identified the closest other agent that is within the Personal Space threshold, the agent moves in the opposite direction by an amount based on the agent Space Adjustment value **ς (i)** and the distance between the two agents i, j according to the formula:

$$(\mathbf{1}-(\mathbf{1}-((\mathbf{S}-\mathbf{distance}(pos(i), pos(j))/\mathbf{S}))^2) * \boldsymbol{\varsigma}\,\mathbf{(i)})$$

This value indicates how far the agent pushes away from the nearest occupant in a single time unit, and scales the amount of movement based on inverse square of the distance. This personal space behaviour is implemented as part of the agents' "tick" method, called once per time unit. After all agents have moved to accommodate the Personal Space requirements, the simulation ticks forwards and the procedure repeats. In these scenarios, the agents will move apart again in the next few ticks if possible but for very dense populations of agents there may be no stable positioning without personal space violations occurring.

4.3. Steering towards Goals

A crucial capability agents need is the ability to move from their current position towards a goal. This fundamental capability can be combined with path planning to produce a variety of microscopic movement behaviour. The basic steering mechanism is based on the agent pointing themselves towards their current goal, and adjusting their position from time unit to time unit, to move themselves closer to the goal. This steering of an agent i is performed using a **Steering Vector** φ(i) and the movement amount is governed by V'(i). The rate of steering adjustment is given by θgoal**,** giving the number of degrees of adjustment per tick until the agent is satisfied that it points towards the goal.

4.4. Avoiding Obstacles

With the Steering Vector, the agent can direct itself towards a goal and move towards it. Without obstacle avoidance, the agent would bump into obstacles or other agents close to its chosen path. By adjusting the Steering Vector to steer around obstacles in its path, the agent can safely travel to its goal. To adjust the Steering Vector, the agent first needs to look ahead to determine if it is approaching an obstacle. To do this, the agent creates 2 "Eye" vectors. These vectors are projected out directly from the sides of the agent by an amount equal to the Obstacle Lookahead distance, **SD,** and aim in the same direction as the Steering Vector φ(i) .

The lines described by these Eye Vectors are checked against the building geometry to determine if there is a collision coming up (Figure 2). If there is a Collision on the "Left Eye" but not on the "Right Eye", then the upcoming obstacle can be avoided by steering more to the right; similarly if there is a collision on the Right Eye and not the Left the agent steers left to avoid it. The agent then also adjusts its current position by "sidestepping". This movement causes the agent to step 1 unit to the left or right (i.e. perpendicular to the steering vector) if there is a collision on the Right or Left Eye. If both Eyes are in collisions, then the agent slows down to avoid impacting the obstacle while the steering vector is adjusted to steer away over the next few ticks. The amount of collision avoidance steering adjustment is governed by the parameter **θevade**.

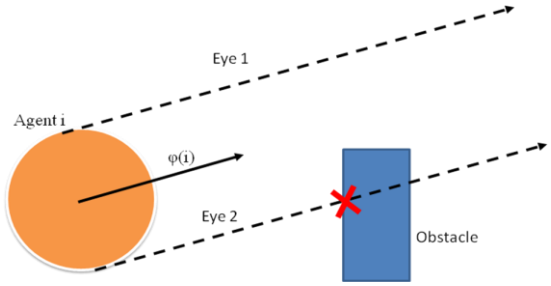
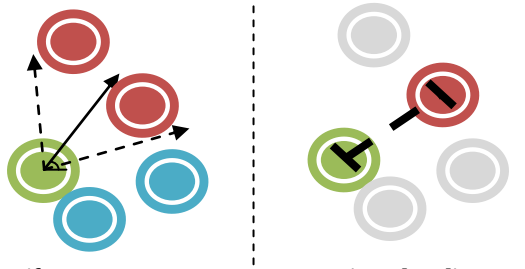


Figure 2 - Agent Lookahead for Avoiding Obstacles

4.5. Braking and Avoiding collisions between Agents

If an agent is approaching another agent, it needs to slow down to avoid a collision. To produce behaviour such as queuing at bottlenecks, the agents need to be able to slow themselves down to avoid collisions and to stay at this low speed while waiting for the area ahead to clear up. This braking behaviour is achieved by periodically scanning the area ahead of the agent for other agents in line of sight within a vision cone delineated by **θbrake** (Figure 3). The agent then adjusts the impact the Speed value **V(i)** has on movement by multiplying it by a Brake Factor value **η(i),** between 0 and 1.0. η(i) is based on the proximity of the nearest other agent j and the angle difference between φ(i) and φ(j). This is implemented by iterating through all other agents in the simulation, and placing any agents within a set number of degrees (Brake Angle parameter **θbrake**) to the left or right of the Steering Vector in a "Brake Candidate" set. This set is iterated through to determine the distance to the closest agent in the set. This distance determines the amount of braking to be applied to **V(i)** to produce the final movement amount, **V'(i)**.



Identify occupants ahead within the frontal cone (Red)

Determine the distance to the nearest of those occupants.

Figure 3: Agent Braking to avoid collision

4.6. Resolving Overlap ("Enforce core")

In some cases, agents may be in a collision with each other (i.e. their radii overlap). This can occur when very dense populations are placed in confined areas where no amount of adjustment can produce a configuration without collisions, or when fast moving agents don't slow down quickly enough and collide with slow moving agents. In such a situation, the agents should prioritize the resolution of the overlap over their goal-directed movement. To achieve this, we implement an "Enforce Core" behaviour invoked during collisions which disables goal-oriented movement until the overlap is repaired (via Personal Space adjustments described in Section 3.2.

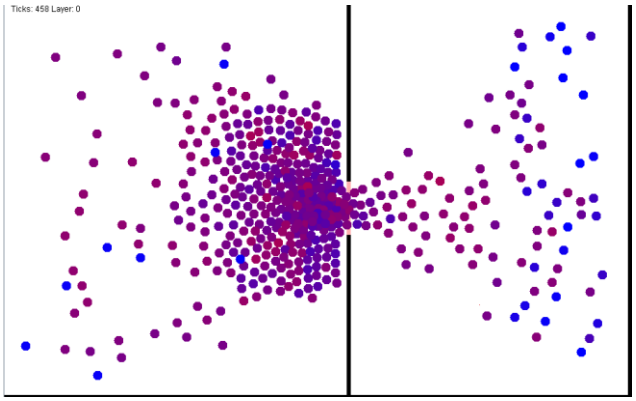


Figure 4 - Agent Overlapping without core eforcement

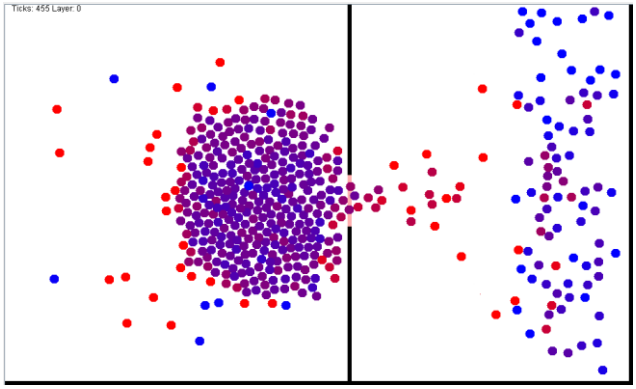


Figure 5 - Enforcing Agent core

Implementation of the Enforce Core behaviour was tested by generating large numbers of agents in a room to the left of the simulation world and sending them through a narrow doorway to the right. The over-supply causes high congestion and without the Enforce Core mechanism we find a high degree of agent overlapping as the pressure of agents trying to get to the goal exceeds the force of the agents attempting to maintain personal space (Figure 4). With Enforce Core enabled (Figure 5) we find that the goal-movement pressure is reduced as agents stop to correct for personal space whenever overlaps occur, and there is a much more stable configuration near the doorway and an eightfold reduction in agent overlapping.

5. VALIDATION EXPERIMENTS

The EvacSim pedestrian model was tested in a series of experiments to compare with real-world pedestrian movement experiments. As a key factor for building evacuation planning is the number of occupants that can be

safely moved through a given area of the building, bottleneck throughput is a significant metric for evaluating egress simulation models. For adequate decision support, evacuation simulation needs to produce throughput metrics in line with real-world pedestrian behaviour.

The first set of experiments was performed to determine the flow rate of pedestrians through a bottleneck. This experiment involved generating agents and instructing them to walk through a narrow corridor and out the other side, where they are removed from the simulation. The experiment is repeated with a variety of bottleneck widths and the maximum flow throughput is recorded in each case.

First we set the experiment up with the widest corridor width to be tested (2m). Agents are introduced in the left-side room and instructed to move to the right. The first introduction rate tested is initially set low (1.25/second), with the procedure repeated for progressively higher introduction rates (up to 6.66.../second); and the actual throughput achieved is noted for each inflow rate attempted. The purpose of this procedure is to discover an optimal throughput value for the corridor. At low supply rates the throughput achieved increases linearly but at higher supply, we observe *lower* throughput (Figure 6) caused by crowding at the entrance and inter-agent jostling, corresponding with the traffic models of Nagel, K; Scheckenberg [21]

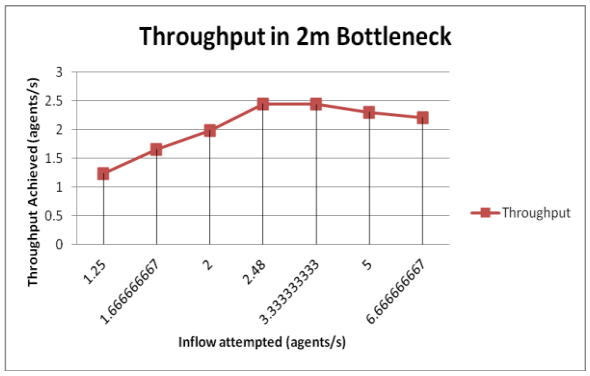


Figure 6 - Throughput Achieved for 2m corridor

The experiment is then repeated varying the corridor width, and gradually increasing the agent supply until the throughput peaks, giving the optimal throughput for width of corridor. These results are compared against the real-world data described in Section 3 in the graph of Bottleneck Throughput (Figure 7). In this comparison we observe that EvacSim throughput results fall well within the range of these real-world results. These results shows that EvacSim's pedestrian model accurately reproduces bottleneck congestion and throughput found in real pedestrian behaviour. We observe an underestimation of throughput for very narrow bottlenecks (70cm and below); however bottlenecks of this size are not likely to be a feature of building evacuation (as they would be more narrow than the minimum door width typically stipulated by building regulations[22,23]).

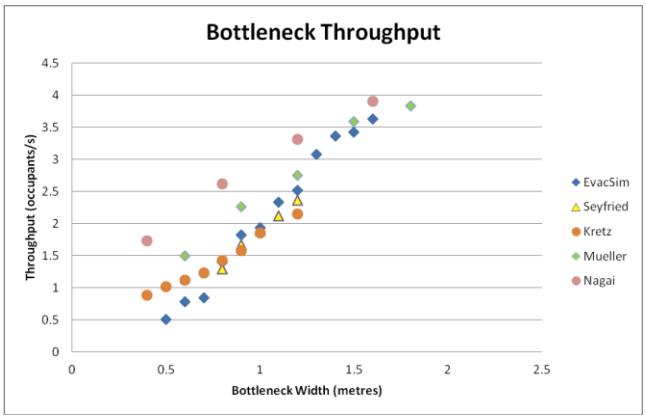


Figure 7 - Bottleneck throughput comparison of EvacSim with real-world experiment results

6. T-JUNCTION EXPERIMENTS

Crowd merging is a key feature of building evacuation, as small groups in the building head towards exits and come together into larger groups. Particularly significant for evacuation planning are the movement characteristics of occupant flows merging together. Zhang et al [18] performed real-world experiments on crowd merging in t-junction spaces and identified a phenomenon wherein the average velocity for a given density of occupants is lower at the point of merging than it is in the space after the merge occurs. They suggest that the cause of this is likely to be slowdown from negotiation of the merging of two streams and tentativeness.

6.1. T-junction experiments with default agent

Experiments were performed in a scenario based on the t-junction pedestrian movement experiment performed by Zhang et. al[18] (Figure 8). In these experiments, pedestrian agents are introduced at two sides of a t-junction, and instructed to move to the point of junction and then continue down the central route. All corridors have the same width, and a grid of simulated square "sensors" is used to periodically sample the velocity and density values in different areas of the t-junction space.

We label appropriate sensors as "Before", "Centre" and "After"; Before and After correspond to the "in front merging, left" and "behind merging" areas of [18]. "Centre" is the merging area of the t-junction, an area unlabelled in [18].

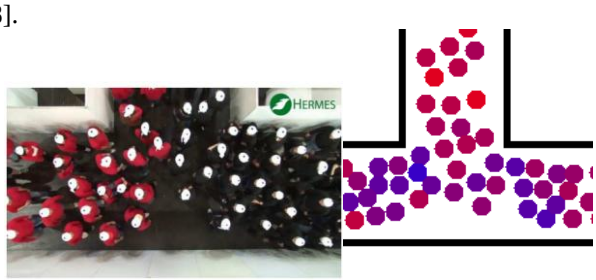
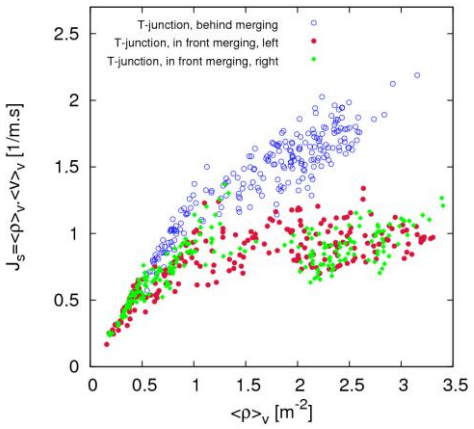


Figure 8 - Zhang[18] and EvacSim t-junction experiments

As with IV-b, we start with a low inflow of agents and gradually increase this inflow over time. The results of each sensor sampling are given as data points in the Specific Flow graph in Figure 10 as Js (average velocity for a given density of occupants) versus ρ/ m^{-2} (occupants per m^{-2}).

With the standard agent model, we find that the Before, Centre and After have little to differentiate each other, in contrast with the data from [18], in which "behind merging" shows a clear seperation from the areas before the merge (Figure 9). The standard agent model doesn't differentiate between agents coming towards one another (i.e. two streams merging) and agents heading in the same direction in a single stream; agents will apply braking even if the agents in front of them are also heading in the same direction.



.

Figure 9 - Flow vs Density in real-world T-Junction experiments (Zhang[18])

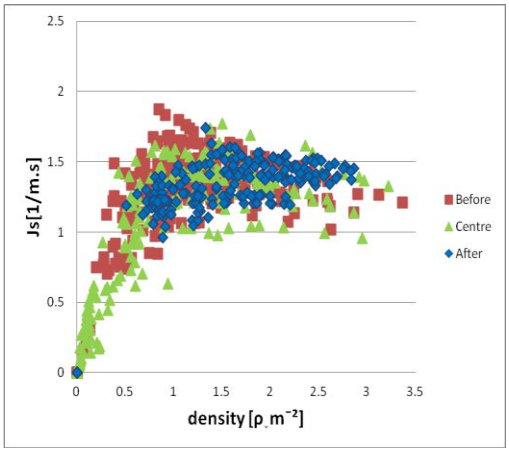


Figure 10 - EvacSim T-Junction Flow vs Density (default agent model)

6.2. T-junction experiments with stream formation behaviour

To reproduce the phenomenon identified by [18], the agents need to distinguish between two different cases when another agent enters the braking area delineated by θbrake and SD (Section 4.5). To achieve this, we extend the braking behaviour to consider the average vector of recent movement for agents in the braking area. If a brake check would normally occur, but the agent in front is heading in the same direction as the braking agent, then the braking agent can maintain its current speed, which we call "stream formation". Agents in a crowd moving in a common direction should travel at a faster average speed due to the lower degree of braking occuring.

Each agent possesses a record of recent coordinate positions which is used to produce an average past motion vector. Agents observe one another over time to produce past-motion vectors for other agents and use these to determine the difference in motion angle between one another, the "motion difference angle". We modify the braking procedure described in Section 4.5 to ignore braking for agents greater than 25% of the Obstacle Lookahead value **SD** away with a motion difference angle of less than the parameter "MotionDifferenceAngle tolerance", or "MDA".

The experiment described in section 5.1 was repeated using two MDA values: 20 degrees and 60 degrees, with the relationship between specific flow and density shown in Figure 11.

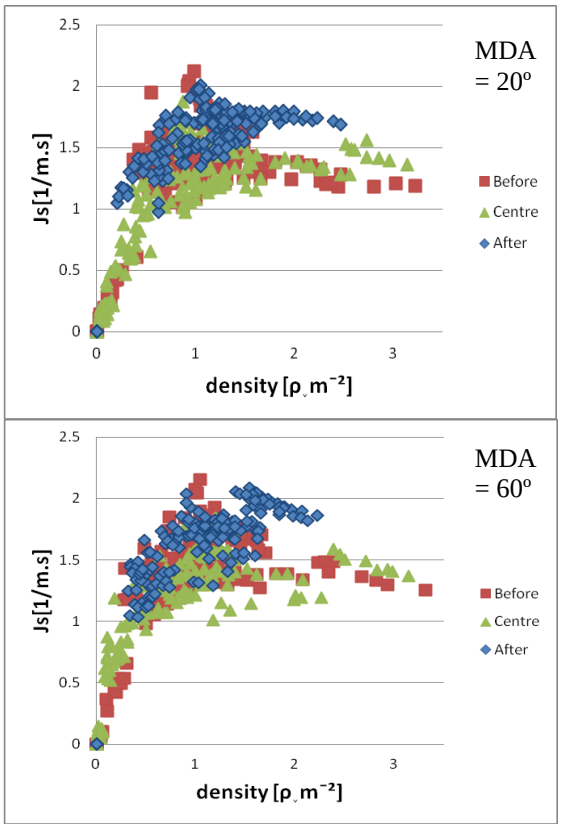


Figure 11 - T-Junction Flow vs Density (Stream-forming agent model)

We find a clear separation between the area after merging and the areas before the merge when using the stream formation behaviour. We also observe greater average specific flow due to less braking occurring in the whole system generally.

6.3. T-junction Heatmaps

Data from the square grid "sensors" is averaged out over the duration of the experiment and used to produce heatmaps for comparison with heatmaps from [18] (Figure 12) illustrating the average densities (Figure 13) and velocities for agents in the t-junction experiment space (Figure 14). In these heatmaps we can observe the impact of a wider tolerance in Motion Difference Angle. With MDA of 20 degrees we observe the velocity of agents in the "centre" area to be lower than the mid-corridor flow after merging, as in [18]. With MDA of 60 degrees, the velocities through the merging centre area remain high as the agents are more likely to ignore the braking procedure despite negotiating a merging.

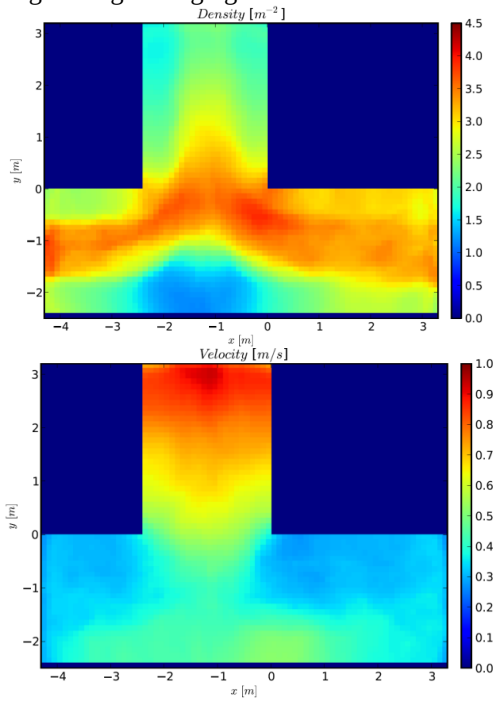


Figure 12 - Density and Velocity heatmaps for Zhang[18] t-junction experiments

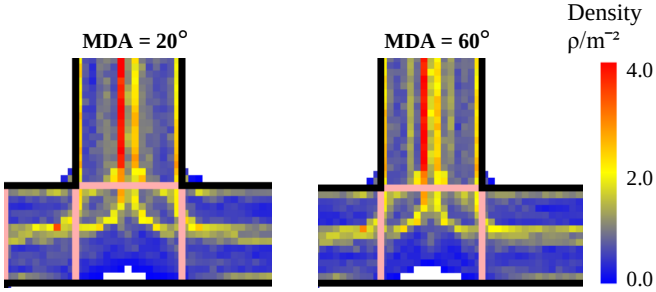


Figure 13 - Average Density heatmaps for EvacSim Stream-forming agents, MDA=20°, 60°

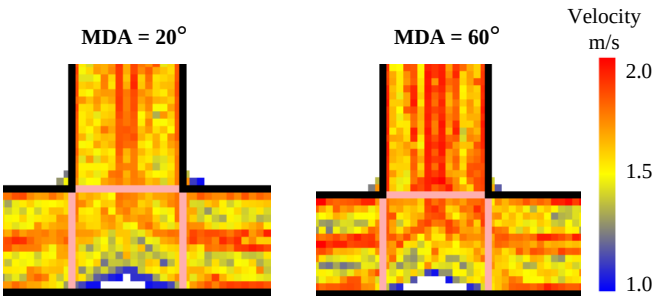


Figure 14 - Average Velocity Heatmaps for EvacSim Stream-forming agents, MDA=20°, 60°

7. CONCLUSIONS AND FUTURE WORK

In this work we described the requirements and details of the EvacSim pedestrian evacuation model. This model achieves complex occupant behaviour through periodic, low-complexity computation to produce realistic pedestrian movement. The accuracy and realism of this model is of key importance in evaluating emergency evacuation planners and providing useful predictive information for safety personnel. To show the model realism, it was validated against real-world pedestrian movement experimental data in bottleneck throughput scenarios, demonstrating that this evacuation simulation model produces throughput metrics that match closely with real-world experiments investigating pedestrian movement.

We identified the special case of crowd merging in building spaces as being of particular importance in evacuation. We made adjustments to the agent model to reproduce real-world experimental results for crowd merging in t-junction spaces. These adjustments reproduces the phenomenon of reduced flow in crowd merging areas relative to unidirectional corridor flow.

In future work, we will continue development and evaluation of the EvacSim simulation model. As part of on-going development of the agent model, we will evaluate the impact of higher-level agent behaviour such as multi-goal path planning and agent group formation on the throughput characteristics of the agent model.

References

[1] Murphy, Seán Óg, Kenneth N. Brown, and Cormac Sreenan. "Problem Decomposition For Evacuation Simulation Using Network Flow." In *Proceedings of the 2012 IEEE/ACM 16th International Symposium on Distributed Simulation and Real Time Applications*, pp. 101-108. IEEE Computer Society, 2012.

[2] Murphy, Seán Óg., Kenneth N. Brown, and Cormac Sreenan. "Predictive Simulation & Evacuation Monitoring in Wireless Sensor Networks". In *Proceedings of Proceedings of*

Artificial Intelligence and Cognitive Systems (AICS) 2011 (pp. 36-45)

[3] Kneidl, Angelika, Markus Thiemann, André Borrmann, Stefan Ruzika, Horst W. Hamacher, Gerta Köster, and Ernst Rank. "Bidirectional Coupling of Macroscopic and Microscopic Approaches for Pedestrian Behavior Prediction."*Peacock et. al.[73]* (2010): 459-470.

[4] Chooramun, Nitish, Peter J. Lawrence, and Edwin R. Galea. "An Agent Based Evacuation Model Utilising Hybrid Space Discretisation." *Safety Science* (2012).

[5] Yang, L., K. Zhu, and S. Liu. "Cellular Automata Evacuation Model Considering Information Transfer in Building with Obstacles." *Pedestrian and Evacuation Dynamics* (2011): 317.

[6] Schadschneider, A., and A. Seyfried. "Validation of CA models of pedestrian dynamics with fundamental diagrams." *Cybernetics and Systems: An International Journal* 40, no. 5 (2009): 367-389.

[7] Dimakis, Nikolaos, Avgoustinos Filippoupolitis, and Erol Gelenbe. "Distributed Building Evacuation Simulator For Smart Emergency Management." *The Computer Journal* 53, no. 9 (2010): 1384-1400.

[8] Wagoum, Armel Ulrich Kemloh, Mohcine Chraibi, Jonas Mehlich, Armin Seyfried, and Andreas Schadschneider. "Efficient and Validated Simulation Of Crowds For An Evacuation Assistant." *Computer Animation and Virtual Worlds*(2012).

[9] Korhonen, Timo, Simo Hostikka, Simo Heliövaara, Harri Ehtamo, and Katri Matikainen. "FDS+Evac: Evacuation Module For Fire Dynamics Simulator." In *Proceedings of the Interflam2007: 11th International Conference on Fire Science and Engineering*, pp. 1443-1448. 2007.

[10] Kobayashi, Hiroyuki, Yutaka Ishimoto, Masaki Fujioka, and Kenichi Ishibashi. "A Multi-agent Evacuation Simulator To Design Safe Cities For High Quality Of Life With Computer Clustering." In *SICE, 2007 Annual Conference*, pp. 3043-3046. IEEE, 2007.

[11] Helbing, Dirk, and Peter Molnar. "Social Force Model For Pedestrian Dynamics."*Physical review E* 51, no. 5 (1995): 4282.

[12] Reynolds, Craig W. "Flocks, Herds And Schools: A Distributed Behavioral Model." In *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4, pp. 25-34. ACM, 1987.

[13] Olfati-Saber, Reza. "Flocking For Multi-agent Dynamic Systems: Algorithms And Theory." *Automatic Control, IEEE Transactions on* 51, no. 3 (2006): 401-420.

[14] Seyfried, Armin, Oliver Passon, Bernhard Steffen, Maik Boltes, Tobias Rupprecht, and Wolfram Klingsch. "New Insights Into Pedestrian Flow Through Bottlenecks." *Transportation Science* 43, no. 3 (2009): 395-406.

[15] Kretz, Tobias, Anna Grünebohm, and Michael Schreckenberg. "Experimental Study Of Pedestrian Flow Through A Bottleneck." *Journal of Statistical Mechanics: Theory and Experiment* 2006, no. 10 (2006): P10014.

[16] K. Müller. *Zur Gestaltung und Bemessung von Fluchtwegen für die Evakuierung von Personen aus Bauwerken auf der Grundlage von Modellversuchen*. Dissertation, Technische Hochschule Magdeburg, 1981

[17] Nagai, Ryoichi, Masahiro Fukamachi, and Takashi Nagatani. "Evacuation Of Crawlers And Walkers From Corridor Through An Exit." *Physica A: Statistical Mechanics and its Applications* 367 (2006): 449-460.

[18] Zhang, Jun, Wolfram Klingsch, Tobias Rupprecht, Andreas Schadschneider, and Armin Seyfried. "Empirical Study Of Turning And Merging Of Pedestrians Streams In T-junction." *arXiv preprint arXiv:1112.5299* (2011).

[19] Tavares, R. Machado, and Edwin R. Galea. "Evacuation Modelling Analysis Within The Operational Research Context: A Combined Approach For Improving Enclosure Designs." *Building and Environment* 44, no. 5 (2009): 1005-1016.

[20] Heliövaara, Simo, Timo Korhonen, Simo Hostikka, and Harri Ehtamo. "Counterflow Model For Agent-based Simulation Of Crowd Dynamics." *Building and Environment* 48 (2012): 89-100.

[21] Nagel, Kai, and Michael Schreckenberg. "A Cellular Automaton Model For Freeway Traffic." *Journal de Physique I* 2, no. 12 (1992): 2221-2229.

[22] Technical Guidance Document L. PartB (Fire Safety) (2006). Department of Environment, Community and Local Government, Ireland

[23] Architectural, U. S., and Transportation Barriers. "Americans with Disabilities Act (ADA) Accessibility Guidelines for Buildings and Facilities." *Federal Register* 56 (1991): 173.