Instituto Politécnico
de Castelo Branco
Escola Superior
de Tecnologia

ECSMoS

ECS based pedestrian mobility simulatior

Rafael Souza Cotrim

Orientadores

Alexandre José Pereira Duro da Fonte

João Manuel Leitão Pires Caldeira

Dissertação/Trabalho de Projeto / Relatório de Estágio (Deixar apenas a designação aplicável e sem sublinhado e eliminar esta nota) apresentado à Escola Superior de acrescentar nome da unidade orgânica do Instituto Politécnico de Castelo Branco se aplicável acrescentar nome da instituição associada (caso contrário, eliminar esta nota) para cumprimento dos requisitos necessários à obtenção do grau de Mestre em designação do mestrado, realizada sob a orientação científica do categoria profissional do orientador Doutor nome do orientador, do Instituto Politécnico de Castelo Branco.

Data

1. Introduction

Pedestrian Dynamics is an area of study focused on understanding the movement of pedestrians, which often happens as crowds. Such studies can take the form of an analyzing of data collected from studying the movements of people in the real would [SRC], proposition and evaluation of methods for modeling crowd behavior [SRC], predicting movement patterns in certain spaces [SRC], among others [SRC?].

As such, pedestrian dynamics are of practical use when creating spaces meant to be utilized by people. A better understanding of how people move though a structure may aid during the design phase of a building, allowing for better planing of fire escape routes [SRC]. Similarly, not taking into account how people will behave may turn concerts or other large events into deadly crowd crushes or increase the number of trampling incidents [SRC]. Finally, even when there is little risk of loss of life, they may still be useful for incising the throughput of infrastructure such as trains stations, airports and others [SRC].

One of the best tools from this area of study comes in the form of simulation models. Computer simulations allow us to check how pedestrians will behave in a certain environment without having to spend time and resources on the construction and evaluations of scale models on the real world. This reduces costs and promotes fast iterative designs that may better align with the requirements of spaces.

There are many models for simulating the flow of pedestrians in an environment, however, designing and implementing models capable of reproducing phenomena seen on the real world is a complicated task. Pedestrian Dynamics is an inherently interdisciplinary science [SRC] due to its object of study. It relies on concepts from areas as diverse as physics, engineering, psychology, computer science and sociology. Simpler models may take into account only the physical part of crowd movement and ignore all else, while others deal with the effects of having people with different ages [SRC], disabilities [SRC] or even states of mind such as calm and in panic [SRC].

When these models are implemented, it is often done on top of an exiting simulator or framework. These allow model authors to focus on the most relevant parts of their research while other tasks are handled by code already written and validated by others. Nevertheless, building a model on top of these simulators also comes with certain disadvantages. Their architecture imposes restrictions on how the model can operate, meaning that certain simulators may not be compatible with a model because it breaks one or more of the assumptions made when the simulator was being designed. [NOTE: Example? Here or later?]

Similarly, the architecture and technologies used by the simulator have a performance impact on the simulation. While less impactful than full incompatibility between model and simulator, low performance can significantly slow model development and usage. Larger or more complicated sceneries may take much longer than desired to be evaluated or required additional hardware.

While there are many ways of mitigating such problems, using architectures found in other areas such as Entity Componet Systems (ECS) offer some benefits. ECS is a software architecture commonly used in game development due to their much stricter performance requirement than standard simulators. Users and developers expect such frameworks to be able to handle thousands of entities updating multiple times a second, allowing them to deliver frame rates of upwards of 60 frames per second. Not only that, but, as this work will show, this paradigm is very flexible, allowing it to be used into many scenarios, including the study of crowds. Despite this, ECS is still very uncommon in the scientific world.

With this in mind, I propose a new simulation framework for pedestrian dynamics: Entity Componet Systems Mobility Simulator (ECSMoS). This simulator is based on the ECS architecture and has the objective of being as flexible as possible for model authors and implementers while maintaining a high performance.

[Section description]

[TLDR of results]

2. Overview of the locomotion models, frameworks and architectures

2.1 Pedestrian modeling

Modeling pedestrians is a complicated affair. Humans have both conscious, subconscious and even physical aspects which affect how one moves. Because of this, there are many competing models in use, each one with its own benefits and limitations. There are a many ways of categorizing them generally. One of the most common is based on the scale of the entities involved [3]. Microscopic models focus on individual pedestrians and their movement, while macroscopic models avoid dealing with interactions between discreet entities, preferring to model the behaviors of whole crowds directly.

Another way of categorizing models is proposed by [4], which divides models into the following groups:

[These could (should?) be expanded]

- Mechanics-based models: Inspired by continuum mechanics or force models.
- Cellular automata models: These interpret the world as discretized units on a grid. Agents operate under defined rules that determine how they move on the grid.
- Stochastic models: Use random or probabilistic models to determine behavior.
- Agency models: Pedestrians are treated as agents that are able to sense and reason about the world. They make choices about where to move depending on their perceptions of the outside world.
- Data-driven models: Based on data collected from experiments and other sources. Uses that for building and calibrating a model.

However, both kinds of divisions are descriptive, not prescriptive. Models routinely fit into multiple or in between. Where they fall is mostly determined by the requirements and constraint of the model authors. Exotic requirements may lead to models being completely out of these constraints.

This thesis will focus mostly on the Social Forces Model (SFM), a microscopic mechanics-based model originally proposed in 1995 [2]. That is because many implementations of it have been proposed and added in various simulators over the years. Providing an even ground for comparisons between simulators. Its behavior is known and well understood. In particular, the version described in [1] will be used, as it is more similar to the implementations present in today's simulators.

The SFM describes interactions between pedestrians and the environment as forces. Pedestrians are modeled as circles and, at each point in time, certain forces applied upon them and are used to compute its acceleration, velocity and ultimately position. Though careful calibration, these forces can reasonably model the behaviors of people moving. The original model defined three main kinds of forces. However, some implementations may add more force, taking into consideration other aspects.

The first kind of force to take into account is the driving or motivation force, computed by equation (1). This is a force that attempts to move the pedestrian in the directions of its ultimate destination. If a lone agent was placed in an empty plane and given a target destination, the motivation force would be a constant vector pointing to its destination. If there are obstacles on the way, the direction of the force might change to steer the pedestrian into the shortest available path. Which is the shortest path needs to be computed by some other means, as the model itself does not specify how.

$$F_i^{\vec{drv}} = \frac{v_i^0 e_i^{\vec{0}} - \vec{v_i}}{\tau} \tag{1}$$

where:

$F_i^{\vec{drv}}$ is the driving force on pedestrian i

v_i^0 is the desired speed: the speed at which the pedestrian i would prefer to walk

e_i^0 is a unit vector pointing to the desired direction

$\vec{v_i}$ is the current speed of pedestrian i

τ is the reaction time: the time it takes pedestrians to notice changes in theirs surroundings

The environment around an agent may also impose repulsive forces upon them. Pedestrians naturally attempt to keep some distance between themselves and others both for comfort and to guaranty the ability to continue moving forward. This is modeled as a repulsive force between them. The force between pedestrians i and j is symmetrical between then, and the final force (2) on i is the sum of all repulsive forces applied by other pedestrians (3).

$$F_i^{\vec{rep}} = \sum_i \vec{f_{ij}} \tag{2}$$

$$\vec{f_{ij}} = [A_i e^{\frac{r_{ij} - d_{ij}}{B_i}} + kg(r_{ij} - d_{ij})]\vec{n_{ij}} + \kappa g(r_{ij} - d_{ij})\Delta v_{ji}^t \vec{t_{ij}} \tag{3}$$

where:

$F_i^{\vec{rep}}$ is the total repulsive force from other agents on i

$\vec{f_{ij}}$ is repulsive force from j on i

$r_{ij} = r_i + r_j$ is the sum of the radius of i and j

d_{ij} is the distance between the center of i and the center of j

g is the contact distance between the pedestrians. If they are not touching, it is 0, otherwise it is the distance between the center of i and the center of j

$\vec{n_{ij}}$ is a unit vector pointing from j to i

$\vec{t_{ij}}$ is a unit vector perpendicular to $\vec{n_{ij}}$, rotated counterclockwise

$\Delta v_{ji}^t = (\vec{v_j} - \vec{v_i}) \cdot t_{ij}$ is the tangential velocity difference of i and j

A, B, k, κ are calibration constants

In practical terms, the repulsive force has two main components: a normal and a perpendicular vector. The normal forces two pedestrians away from each other, increasing depending on how close they are. If the pedestrians are close enough to touch, that force additionally increases by a secondary factor that is dependent on how much they intersect. In this way, the normal component captures both the psychological desire for space and the physical constraints preventing pedestrians from packing too tightly. The perpendicular vector represents sliding forces. When two pedestrians touch, there is some friction between them that acts to slow them down. Friction acts perpendicularly to the normal and against the direction of movement.

[Put example images]

The final class of force to consider is the obstacle force. Much like the previous repulsive forces, agents also attempt to keep a certain distance from them to walls and other obstacles in their surroundings. If

they touch those objects, there is also an additional factor that increases the normal repulsion and a factor describing friction. However, most obstacles are not perfect circles, so the function uses the closes point of the obstacle for distance measurements.

$$\vec{F_i^{obst}} = \sum_o \vec{f_{io}} \tag{4}$$

$$\vec{f_{io}} = [A_i e^{\frac{r_i - d_{io}}{B_i}} + kg(r_i - d_{io})]\vec{n_{io}} + \kappa g(r_i - d_{io})(\vec{v_i} \cdot \vec{t_{io}})\vec{t_{io}} \tag{5}$$

where:

$\vec{F_i^{obst}}$ is the total force from obstacles on agent i

$\vec{f_{io}}$ is the force from obstacle o on i

r_i is the radius of i

d_{io} is minimum distance from the center of i to any point of o

g is the contact distance between the pedestrian and the obstacle. If they are not touching, it is 0, otherwise it is equal to d_{io}

$\vec{n_{io}}$ is a unit vector pointing from the center of i to the closes point of o

$\vec{t_{io}}$ is a unit vector perpendicular to $\vec{n_{io}}$, rotated counterclockwise

A, B, k, κ are calibration constants

[Put example images]

In each simulation step, all of these forces are computed for each agent, then they are used to compute its new speed and position. Using basic Newtonian physics, equations (6) and (7) can be derived. Once the agents are moved to their new locations, the simulation forces are calculated once again. This process repeats until the desired results are achieved.

$$\vec{v_i^{new}} = \vec{v_i} + (F_i^{drv} + \frac{\vec{F_i^{rep}} + \vec{F_i^{obst}}}{m_i}) \cdot \Delta T \tag{6}$$

$$\vec{pos_i^{new}} = \vec{pos}_i + \vec{v_i^{new}} \cdot \Delta T \tag{7}$$

where:

$\vec{v_i^{new}}$ is the new speed of the agent i

$\vec{v_i}$ is the previous speed of i

$\vec{F_i^{drv}}$ is the driving force on pedestrian i

$\vec{F_i^{rep}}$ is the total repulsive force from other agents on i

$\vec{F_i^{obst}}$ is the total force from obstacles on agent i

m_i is the mass of i

ΔT is the length of the simulation step

$\vec{pos_i^{new}}$ is the new position of i

$\vec{pos}_i$ is the previous position of i

Despite its relative simplicity, this model is quite robust and is capable of replicating certain phenomena that are seen in imperial studies. For example, it replicates the "Faster-Is-Slower Effect", which happens when the flow rate of pedestrians though an obstruction paradoxically decreases once a certain speed is surpassed [1]. The SFM reproduces lane formation, which is when pedestrians form lanes of people going in the same direction when two streams of people converge in a limited space [SRC].

[Put example images]

2.2 Existing simulation frameworks

[This needs a better "warm-up" before just going into the text itself]

While there are many commercial products for simulating the behavior of pedestrians, they are all for the most part closed source and proprietary, meaning that it is difficult to scrutinize their results or implement new models. Solutions like this include [List some commercial solutions]. Due to their limitations, scientific research tends to focus on more open simulators.

To determine which simulators were most used and relevant to this work, a survey was performed. Various scientific article databases were searched with terms relevant to this area of study and the simulations frameworks used were noted. A substantial portion of the works used custom-made software for the specific use case the researchers had in mind. Of the general simulation frameworks found, only open source ones with at least one update since 2015 were considered. This last requirement is to guarantee that the evaluated simulators are either in active development or occasionally receive additional updates.

- Vadere
- Menge
- JuPedSim
- SUMO (Only needs to be mentioned, is not robust enough for this use case to be used in the comparison)
- MomenTUMv2 (needs to mentioned at least, possibly be part of the comparison)
- FDS+Evac (Only needs to be mentioned, is not used for general simulations, mostly fire escape)
- Cromosim?
- jCrowdSimulator?
- Mesa (only needs to be mentioned, it is more of an agent-based simulation library and does not include any specific to pedestrian simulations)
- Agents.jl (only needs to be mentioned, it is more of an agent-based simulation library and does not include any specific to pedestrian simulations)

[At the end of this section, I need to give "a list o problems" that these simulators have so that the architecture section can have some context. At the end of that section, I can raise some points on how this architecture could solve them]

3. ECS architecture

Despite its general use in game development, the ECS architecture has a loose definition and is sometimes confused with other similar software architectures that are also common in the area. Additionally,

the various implementations can be substantially different due to their underlying technologies and objectives. Because of to this, some may not have most of the benefits associated with it, effectively making them simply code organizational strategies. This section will discuss the more common aspects.

Generally speaking, in a ECS implementation, most things is divided into three main kinds of concepts: entities, components and systems. Entities represent general objects in the world. For example, in a crowd simulation, each pedestrian would be an entity, but so would obstacles and other things that might affect them. Usually, they consist of a simple identifier which can be used to group other kinds of data.

Components can be considered the attributes of an entity. Each component stores the data relevant for a certain aspect of an entity's behaviors. Following the previous example, each pedestrian would have a position component and a speed component, each of which contains all the data necessary to characterize the entity on the relevant aspect. Entities may have as many components as necessary to describe their behavior. However, components do not contain logic. Their job is to store values.

Systems are processes that can read and modify components. They are the main place for logic in this architecture and generally speaking can access all data as though it was global. A simple system might move entities to their next position according to their speed component. More complicated one can simulate physics, UI, user interactions and much more. Systems operate in a defined sequence and usually don't have internal state, however this varies across implementations. Still, most of the data will be stored in the components.

[Example image]

Another way of understanding is to compare these concepts to a traditional relational database. In this paradigm, there is a table that contains all entities and one additional table for each component type. A system then would perform queries for the data in database. The entity table can be used for joining data from different component tables and the presence or absence of each component can be used to filter out certain entities. In this way, systems can perform operations only on entities which are relevant to the systems function. A system may also write back to the database and following systems can access the newly written data.

[Example image]

[Benefits]

[Drawbacks]

4. ECSMoS implementation

4.1 Chosen Technologies

- Rust
- Bevy ECS

4.2 ECSMoS Architecture

4.3 Features?

5. Evaluation

6. Conclusion

References

[1] Dirk Helbing, Illés Farkas, and Tamás Vicsek. “Simulating Dynamic Features of Escape Panic”. In: *Nature* 407 (Sept. 28, 2000), pp. 487–490. DOI: 10.1038/35035023.

[2] Dirk Helbing and Peter Molnar. “Social Force Model for Pedestrian Dynamics”. In: *Physical Review E* 51.5 (May 1, 1995), pp. 4282–4286. ISSN: 1063-651X, 1095-3787. DOI: 10.1103/PhysRevE.51.4282. arXiv: cond-mat/9805244. URL: http://arxiv.org/abs/cond-mat/9805244 (visited on 05/17/2025).

[3] Benedikt Kleinmeier et al. “Vadere: An Open-Source Simulation Framework to Promote Interdisciplinary Understanding”. In: *Collective Dynamics* 4 (Sept. 3, 2019), pp. 1–34. ISSN: 2366-8539. DOI: 10.17815/CD.2019.21. URL: https://collective-dynamics.eu/index.php/cod/article/view/A21 (visited on 11/08/2024).

[4] Francisco Martinez-Gil et al. “Modeling, Evaluation, and Scale on Artificial Pedestrians: A Literature Review”. In: *ACM Comput. Surv.* 50.5 (Sept. 2017), 72:1–72:35. ISSN: 0360-0300. DOI: 10.1145/3117808. (Visited on 05/17/2025).