ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Burning Man:
Simulating the behaviour of people in the event of fire

Project Report

Nico Hauser, Andri Horat, Elias Schmid, Jonas Spieler

Zurich
December 2019

Agreement for free-download

We hereby agree to make our source code for this project freely available for download from the web pages of the SOMS chair. Furthermore, we assure that all source code is written by ourselves and is not violating any copyright restrictions.

Nico Hauser

Andri Horat

Elias Schmid

Jonas Spieler

Contents

1 Abstract

This paper describes a model for the behaviour of people in the event of an emergency situation and it's simulation. The model tries to incorporate the general repulsion of people standing too close, the formation of groups in panic situations, the placement of exit signs and many other factors such as different speeds. In order to make observations on the defined model, it is simulated using a physics engine written in javascript that makes it accessible from any place at any time and allows for reproducible and verifiable results. The paper defines a general parameterized model of the environment and the agents which, with the right choice of simulation parameters, may even allow for a safety evaluation of specific settings. Simulations using this model were able to verify results of other papers but also put them into perspective.

2 Introduction and Motivations

This paper was created during the course *Agent-Based Modeling and Social System Simulation* held by Dr. Nino Antulov-Fantulin and Thomas Asikis at ETH Zurich. The goal was to create a project matching the title of the course meaning the modelling of a complex system where humans are the agents. A complex system is in contrast to complicated system not per se hard to implement but rather consists of many simple small parts that on themselves act based on simple rules. The fact that there are a lot of these so-called agents gives rise to behaviours of the whole system that are not always easy to predict or even understand even if the simple rules of a single agent are well-known. As computers keep getting faster and new technology is developed every day, simulations are a good way of understanding these complex systems by playing with the available parameters while looking for emerging patterns and then interpreting the implications for the real world.

3 Description of the Model

For readability this chapter is split as the definition of the environment is to some extent independent of the agent's model.

3.1 Environment model

Before agents can react to a fire an environment has to be created which in our case is a building. The environment model consists of walls, obstacles, doors and signs all of which are objects the agent interacts with. Walls are the most important part

as they give the environment a shape and restricts the agent's movement. The same goes for obstacles, the reason why walls and obstacles are separated in our model is that the agent use a raytracing algorithm for finding a path and obstacles such as tables may prevent an agent from moving but do not interfere with their vision. Next are the doors and signs which also have similar roles. As it is the case in real life, safety signs guide an agent to the closest exit where agents are able to escape. Doors have the same effect on agents but represent something different as they're not a physical object but rather represent the agent's memory of where they entered a room. Both signs and doors are directional in our model as both, safety signs and doors, should only be used in one direction when escaping a building.

3.2   Agent model

For the agent model a so-called social force model is used. In short agents behave like particles in newtonian physics. Every agent has a mass, position and velocity and and the intentions and interactions of the agents are represented by forces. The agents are approximated as a circle in the physics engine where each agent has different intrinsic properties that are randomly generated using a normal distribution using a mean and a standard deviation. On one hand the circle radius of the agents is varied representing the different shoulder widths as it was done in [1]. In contrast to this paper, the desired velocity is chosen from a normal distribution as well as this seems more reasonable than all agents having exactly the same desired velocity. In addition we also randomly choose a reaction time that determines how fast an agent does get moving into a given direction.

After the generation of these random values, the agents are a sole product of their environment and their behaviour is based on a set of simple rules.

- Wall repulsion

  To model human behaviour, agents should in general prefer to move away from walls as most humans prefer to have some space around them.

- Agent repulsion

  For the same reason people do not like to stand close to each other, especially not in emergency situations. The closer the distance is, the greater is the applied force on every pair of agents to move away from each other.

- Target attraction

  Last but definitely not least the agents need a task. In the best case, they should try to reach one of the defined escape zones in order to safe themselves. In real life, safety signs guide people to these safe zones so the agents are instructed

5

to always be on the lookout for visible signs guiding them to safety, so we do the same in our model and guide agents to the closest visible safety sign or door. In addition agents will remember what signs and doors they have already passed and won't go into the direction of the same sign or door twice. Should be signs and doors be placed in a strange order, this behaviour ensures that agents won't go into the wrong direction twice. An exception is made to this rule if an agent doesn't have a target anymore. Then they "forget" everything they visited before and start from scratch. This allows for some more complex behaviours such as an agent being pushed back into the room he just was in because of other agents rushing the hallway. If the memory wasn't cleared, the agent now wouldn't leave the room a second time which of course would be a real bad behaviour.

4 Implementation

The implementation of this model is done in javascript using a game library called phaser that already includes a physics engine and has means of drawing the result onto a `HTML` canvas meaning the whole simulation can easily be rendered on a webpage and doesn't require any setup.

4.1 Environment construction

As people are at least to some extent the product of their environment, it is very important that the simulations does not have a fixed environment in which all simulations take place. In fact it is one of the most important parameters when looking at the results, i.e. how many people survived a fire emergency. People in a totally enclosed room have no chance of escaping whereas agents sitting next to an escape zone will have no issues. To make the change of this parameter as accessible as possible, the implementation uses an open source format called `Tiled's map format` (TMX).
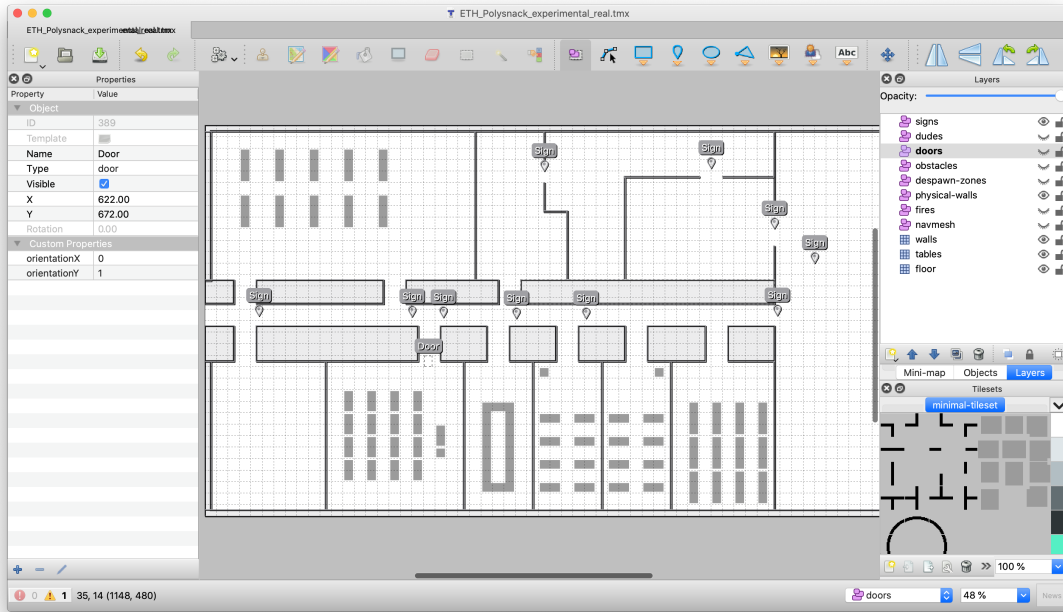
The editing screen of Tiled

Using this editor anyone with basic computer skills is able to adjust the given examples or of course define their own setting. This makes the model much more accessible and allows anyone to make their own tests using this model.

4.2   Agent behaviour

As phaser includes a physics engine we can make use of this by simulating the behaviour of the agents based on forces acting on them where the final movement is the vector addition of all these forces. Phaser has an `update` function that is called every time a new frame is calculated. which is perfectly suited for such a task. The different behavioural rules are implemented as follows

- Wall repulsion

    -

- Agent repulsion

    Agent repulsion can be implemented rather easily: for every pair of agents a force

$$f_r = c_1 \cdot e^{-\frac{d}{c_2}}$$

where c_1 and c_2 are adjustable parameters but constant for the two agents and d is the distance between the two agents. As the force is proportional to e^{-d}, the force is exponentially decreasing when linearly increasing the distance, i.e. exponentially increasing when linearly decreasing it. The initial values for c_1 and c_2 were empirically determined and are dependant on phaser's internal units.

- Target attraction

  For checking what signs are visible, we iterate over all of them and use a technique called raytracing. This means we draw a line from the agent's position to the sign's position and check whether this line intersects any wall defined in the environment.

| An agent's vision blocked by a wall | An agent's sight of a sign visualized by a red "ray" |
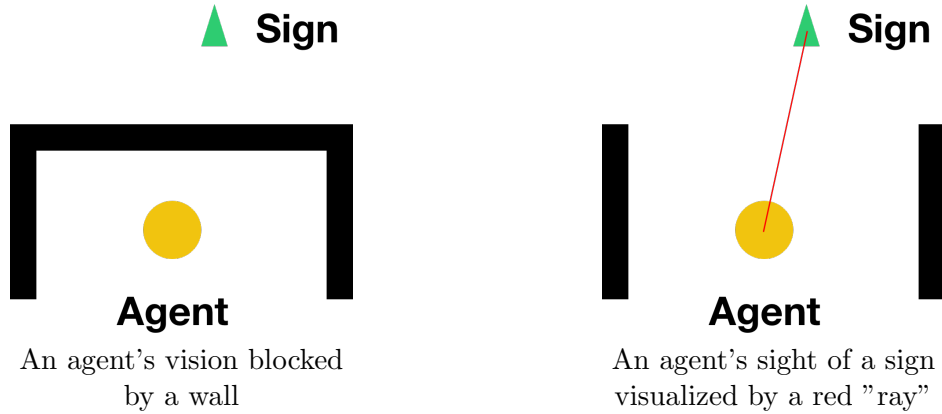
  In the first figure above the agent is surrounded by walls meaning a line between the agent and the sign would intersect the front wall and thus in this case the agent wouldn't move towards the sign.

  If that's not the case, the sign is indeed visible and we can apply a force into the direction of the sign. This works fine until you introduce obstacles drawn in gray instead of black which stands for a wall. Obstacles collide with the agent as de walls, meaning the agent won't "climb" over it, the difference is that the agent can see "through" or over an obstacle. A good example for obstacles are tables.

Sign

Agent

Agent stuck within obstacles

Sign

Agent boxed in obstacles using pathfinding
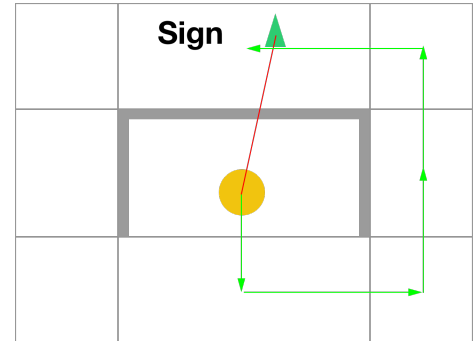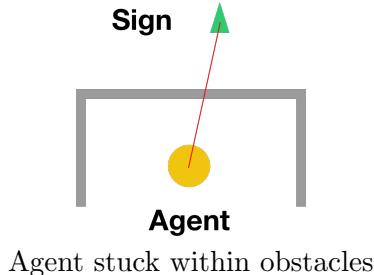
Now the issue with directly applying a force to the agent into the direction of the ray becomes obvious, the agent will get stuck as it cannot move through the wall. To circumvent this issue, a so called pathfinding algorithm is used, once the agent sees a sign. Now the agent doesn't just move into the targets direction but rather finds a way to the target and this way is able to walk around obstacles. The pathfinding algorithm works by generating an ordered list of points that the agent can follow to get to the target position.

As soon as we know the target of an agent, we can compute the velocity-correcting force, which tries to accelerate the current physical velocity v_0 of the agent to the desired velocity v. We compute the force by

$$f_t = m \cdot \frac{v_0 - v}{\tau}$$

where m and τ are constants, that can be interpreted as the mass and the reaction time of an agent. In our simulation the m is chooses based on the randomly chosen radius and τ is defined per agent and can be used directly.

5   Simulation Results and Discussion

To test our tool againt existing papers, we tried to replicate experiments of an existing article [1]. Our experiment 1 was described in [1], whereas our experiment 2 was shown in the video that belongs to the article [1].
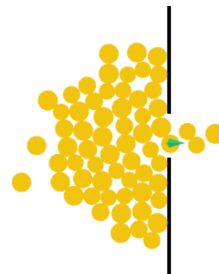
5.1   Experiment 1

For the first experiment we replicated the experiments from the paper *Simulating dynamical features of escape panic* by Dirk Helbing, Ills Farkas and Tams Vicsek in

9

which the agents had to pass a bottleneck in an escape situation. The conclusion was that for a small bottleneck there was no improvement in efficency of escaping when an agents desired velocity was increased. Thus the only parameter that determines the rate of escape is the size of the bottleneck.
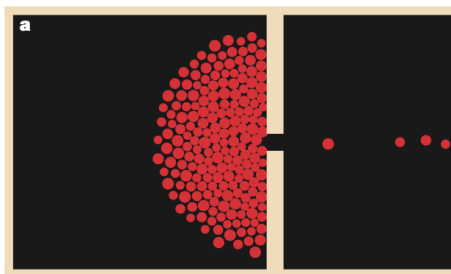
Setup from [1]   This paper's setup
We created a setup similar to that in [1] which is the right image above.

5.2   Experiment 2

5.3   Experiment 3

5.4   Interpretation of the simulation result

When running the simulation several times with a range of different parameters, we can observe some similar behaviour as it is already described in other articles.

5.4.1   Clogging

When running the simulation, one might notice that the agents competing for a door are closer together. This is due to the fact, that all agents are pushing towards the door, and hence pushing them together. It worsens the situation, as the friction between the agents increases, which slows down the escape.

5.4.2   Keeping the distance

Interestingly, we can observe how agents are pushed back into rooms. Sometimes one could even witness how some agents use another path to avoid the crowd. This is actually what we would expect because of the mutual repulsion of the agents.

5.4.3   'Faster-is-Slower effect'

There are already many published papers regarding the social-force-model. Of which many describe the 'Faster-is-Slower effect' [1, 2]. The 'Faster-is-Slower effect' de-

scribes the phenomenon where the escape through an exit door is slower if the agents are pushing harder. When running our simulation on our default map, we did not observe this effect. The chances are, that this is due to the long corridor, where the agents make up time with a higher speed.

5.5   Adapting the parameters for better results

5.6   Finding the model's limits

5.6.1   Evaluation of the building model

5.6.2   Evaluation of the agent model

5.7   Evaluation of the simulation

6   Summary and Outlook

- agents should only interact via forces if they can see each other (improve using raytracing) - use empirical values for the parameters and constants - use a more stable physics engine - implement a random fluctuating force to prevent deadlocks

7   Bibliography

[1] Dirk Helbing, Ills Farkas, and Tams Vicsek. Simulating dynamic features of escape panic. *Nature*, 407:487–490, 09 2000.

[2] Peng Wang. *Understanding Social-Force Model in Psychological Principles of Collective Behavior*. PhD thesis, 05 2016.