Emotion contagion model for dynamical crowd path planning with panic

Andrej Sušnik, Timotej Zgonik, and Ema Leila Grošelj

Collective behaviour course research seminar report

February 28, 2025

Iztok Lebar Bajec | izredni professor | mentor

In this article, we implement An Emotion Contagion Model (ECM) for Dynamical Crowd Path Planning from [1]. This model aims to simulate crowd behavior during fire evacuations, considering individuals' personalities based on the OCEAN personality model. The simulation captures a diverse crowd that reacts to the environment in different ways, with agent characteristics evolving based on surrounding individuals and fire sources.
A key focus of the model is the dilemma individuals face when choosing between the nearest exit or opting for a farther but less congested route. Surrounding individuals are estimated by employing clustering techniques. The clustering algorithm by Wu et al. [2024] is compared against other clustering methods. Additionally, we introduce the concept of panic — agents experiencing higher panic levels exhibit more erratic and unpredictable movement patterns. The simulation also incorporates obstacles, such as an office environment, to test agent behavior under constrained conditions.

Dynamic crowd path planning | Crowd behavior | Simulation | Emotion contagion

1. Introduction

Efficient crowd path planning is crucial in various real-world applications, from urban transportation management to emergency evacuations. Wu et al. [2024] present an innovative approach by integrating emotional dynamics into crowd behavior modeling, offering a look into how individual emotions influence collective movement patterns. Their model is called an Emotion Contagion Model (ECM).

This project aims to reproduce the results of the original study to validate its findings and explore potential improvements and other adjustments.

2. Methods

Related work. The key components of the emotion contagion model are shown in 1, and are further explained below.

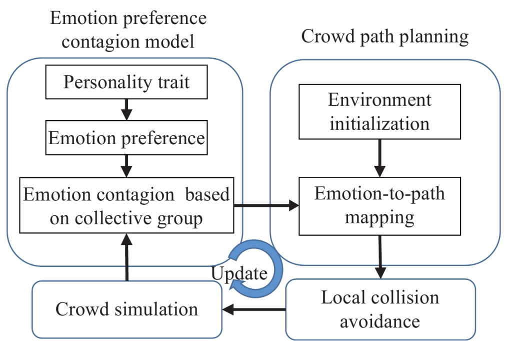
Emotion preference contagion model
Personality trait
Emotion preference
Emotion contagion based on collective group
Crowd path planning
Environment initialization
Emotion-to-path mapping
Update
Crowd simulation
Local collision avoidance

Figure 1. The emotion contagion model as proposed by Wu et al. [2024].

Trait model and mapping to the least time consuming exit. The source article proposes generating agents with distinct values for the five OCEAN traits: openness to experience, conscientiousness, extroversion, agreeableness, and neuroticism. Based on those factors, agents are then assigned an initial distance preference P_d and an initial velocity preference P_v. Agents have larger distance preferences if their personality factors are simple, aggressive, and fragile (corresponding to OCEAN factors: O−, E−, A+), and agents have larger velocity preferences if their personality factors are variable, energetic, and functional (corresponding to OCEAN factors: C−, E+, N+). If an agent has a large distance preference, its velocity preference is usually smaller, and vice versa. An agent with a larger velocity preference will seek to find less crowded paths even at the cost of the distance being longer, while those with a larger distance preference will stick to a path even as it becomes crowded. The model incorporates a strategy where a least expected-time objective function is used to dynamically select paths, factoring in both environmental variables and agents' emotional states and preferences [1].

Change of traits by emotion contagion and other sources. An emotion contagion mechanism further enhances the model by allowing emotional states to spread among neighboring agents, mimicking the way emotions such as stress or calm influence a group. A dampening factor regulates this contagion to ensure an agent's inherent personality traits are at least partially preserved. For contagion of emotions, the simulation requires a method to define collective clusters among the agents to determine which of them will be mutually affected. Agents are defined as being collective neighbors if they share similar motion or have a mutual goal. Information is contagious between neighboring agents in the same group and is affected by the distance between agents and the difference between emotion preferences. Wu et al. [2024] propose an algorithm that begins by identifying the agent with the highest density as the centre of the first cluster. It then evaluates each remaining agent in order of increasing density, checking whether it belongs to an existing cluster based on its distance to the nearest high-density agent. If no cluster matches, a new one is created. Additionally, agents may be affected by environmental contagious sources, such as a fire disaster [1].

Panic contagion and influence. Wang et al. [2022] describe how panic could influence the path planning of an agent. The authors propose a model where panic is contagious and negatively impacts the ability of an agent to find an exit.

Proposed additions.

Panic. To introduce the concept of *panic* to the simulation, each agent is assigned a *panic parameter* and a *panic susceptibility parameter*, where panic inhibits the ability of the agent to move to their goal. As the panic parameter increases, the agent's movement becomes increasingly erratic and random.

The panic susceptibility parameter - how susceptible is someone to surrounding panic, in our approach, is calculated from the OCEAN traits of the agent, as shown in Equation 1. The subscript $_0$ next to a given trait denotes the value being shifted by 0.5 from the original expected value, so that the new expected value is equal to 0. Openness, conscientiousness and agreeableness have negative correlation with panic susceptibility, while neuroticism has positive correlation. The distribution of panic susceptibility remains equal to the distribution of the traits: $N(0.5, 0.1)$. We decided that all traits are equally important: $w_O = w_C = w_A = w_N = \sqrt{1/4}$.

$$panic\_susceptibility = -w_O * O_0 - w_C * C_0 - w_A * A_0 + w_N * N_0 + 0.5, \quad \text{where } w_O^2 + w_C^2 + w_A^2 + w_N^2 = 1 \qquad [1]$$

Each agent is also assigned a current *panic parameter*. This parameter represents the probability that the agent will not behave according to their personality and will either freeze or move randomly. This parameter can be modified in three ways. Every agent seeks to reduce their panic to a baseline level, the return to the baseline level being dependent on their panic susceptibility parameter. The relationship is described in Equation 2 for every Δt.

$$current\_panic -= (0.5 + (0.01 - 0.5) \cdot panic\_susceptibility) * (current\_panic) \qquad [2]$$

There are two means by which the panic parameter of an agent may increase. Agents are susceptible to the panic of other agents in their cluster and their panic parameter value will accordingly change to match the cluster's average, with the speed of the change governed by their panic susceptibility, as described by Equation 3.

$$\begin{gathered} panic\_diff = average\_cluster\_panic - current\_panic \\ current\_panic += panic\_susceptibility * panic\_diff * 0.05 \end{gathered} \qquad [3]$$

An agent's current panic may also increase if they are near a panic-inducing source, which is described by Equation 4.

$$\begin{gathered} dist = \frac{1}{|position - panic\_source\_position|} \\ current\_panic+ = dist * panic\_susceptibility * 0.001 \end{gathered} \qquad [4]$$

Clustering algorithm evaluation. To implement the emotion contagion, we need to identify the community clusters. The clustering as proposed by Wu et al. [2024] operates as follows: two agents are neighbors if they have the same exit as a goal or if they at least have a similar position and direction. Moving from the agents with most neighbors to the ones without them, we merge clusters so that an agent is in the same cluster as its closest neighbor with a higher or equal degree. Wu et al. [2024] do not provide any reasoning as to why a specialised clustering algorithm was necessary, so as part of this project, we chose to evaluate the performance of their clustering algorithm by comparing it to hierarchical clustering and Fast Label Propagation [2].

Enhanced navigation graph. When an agent chooses an exit, we need to also choose a path that leads from an agent's current position to their chosen exit. This is efficiently done with pre-computing a navigation graph that stores all the shortest paths to a given exit. Additional movement options have been added to the navigation graph, as described in Section 3.

Corrected error. We believe to have found **an error in the source article** with its equation:

$$d_{ori}(i,j) = |arccos(vel_i) - arccos(vel_j)|$$

which does not appear to be a operation that can be performed on a vector, so we instead propose:

$$d_{ori}(i,j) = |atan2(vel_{i_y}, vel_{i_x}) - atan2(vel_{j_y}, vel_{j_x})|$$

where we obtain the correct angle θ from the x-axis [3] for each of a pair of agents and then subtract the angles to obtain the difference between their orientations. The article does not acknowledge why this was used instead of a dot product, and in good faith, we assumed there must have been a reason for that decision.

3. Results

Navigation graph. For each exit, we constructed a navigation graph of shortest paths. Every such graph is a tree or a forest with roots among the nodes belonging to an exit, representing the shortest paths from all possible positions on the grid to one of the exits. It was constructed using Dijkstra's algorithm, allowing movements to all 8 neighboring fields as well as to fields located two steps in one direction and one step in a perpendicular direction. An example graph is shown in Figure 2.

This navigation graph would, in theory, allow for modeling the movement of agents in a more natural way than the basic left-right-up-down version of it.

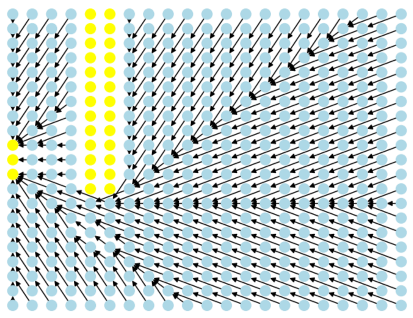

Figure 2. Example navigation graph for a grid of dimensions 10 times 10. Yellow nodes represent the exit of width 3 and the obstacle. Every blue node has its shortest path towards the exit nodes show.

Performance evaluation. To test the limits of our implementation, we conducted a performance evaluation. We used a 256 x 256 grid as an environment and ran the simulation on a machine with an Intel i5-1260P CPU and 16 GB of RAM. Table 1 shows the results, showing that when the number of agents is smaller or equal to 200, the performance could be considered real-time (above 30 frames per second). When we increased the number of agents, the frame rate dropped significantly, reaching only a single frame per second with 1000 agents. Upon profiling the code, we determined that the major bottlenecks were cluster formation and the contagion of emotional preferences step.

Num. agents	Average frames per second
10	290
50	263
100	172
200	70
500	10
1000	1.5

Table 1. Results of performance evaluation

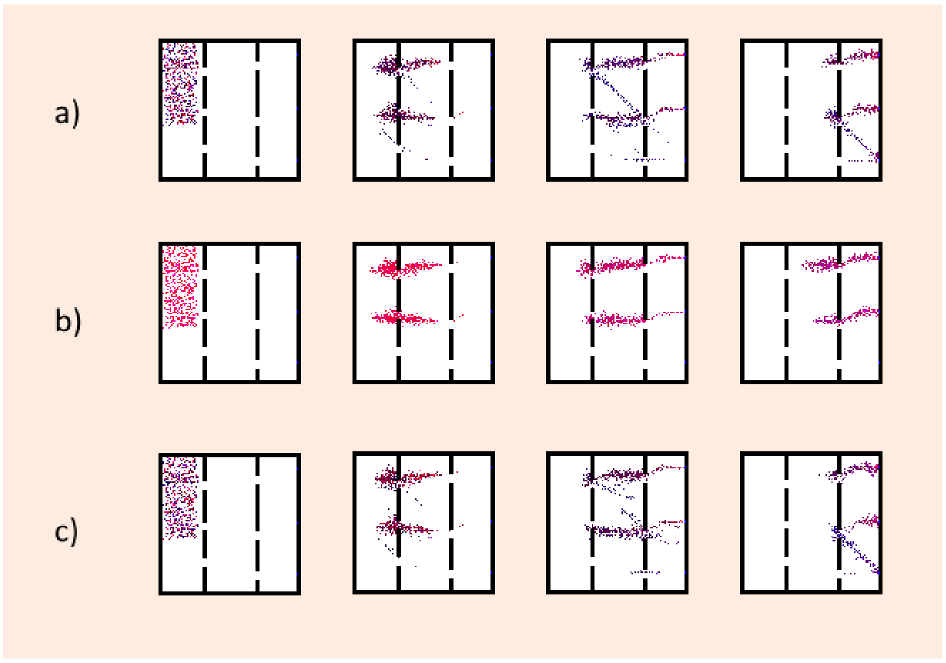

Figure 3. The demonstration of crowd path planning with different emotion preferences and different clustering algorithms. In all cases, the screenshots are taken at frames 1, 25, 55 and 100 of the simulation. Blue dots indicate agents with a high velocity preference, while red dots indicate those with a high distance preference.

Crowd path planning under different emotion preferences. Figure 3 shows the results of the simulation under different initial parameters. In a), around 50% of agents have a high distance preference and the other 50% have a high velocity preference. Agents with a high velocity preference avoid congestion, opting for exits farther away. When agents are near an exit, emotion contagion takes effect, reducing the aggressiveness of all agents, including those with a preference for velocity. In b), all agents have a high distance preference. They are all willing to wait in lines and almost no agent chooses an exit that is farther away. In c), around 50% of agents have a high distance preference and the other 50% have a high velocity preference. We use hierarchical clustering instead of clustering algorithm proposed by Wu et al. [2024]. Hierarchical clustering creates smaller clusters, reducing the time it takes the agents to reach the destination.

Panic. We discovered that the panic parameter has great influence on the paths, as shown in Figure 4. Without accounting for crowd panic, all the paths are very efficient, which is not necessarily the most accurate reflection of a real-world evacuation. When exposed to a contagious source such as fire, agents behave erratically the closer they are to the panic source.

4. Discussion

We have implemented a simulation workflow where we can define the environment with a bitmap image and then run the simulations. We compared the hierarchical clustering algorithm with the algorithm described in the source article and found that hierarchical clustering usually provides better results. The introduction of a panic parameter induced panic-like behavior into the paths of agents, bringing the simulation closer to real-life scenarios.

Many aspects of the implementation had to be reinvented, as Wu et al. [2024] at some points did not elaborate on their solutions. One such instance was with the implementation of the navigation graph generation, as for small environment sizes in the range of 100 × 100 tiles were reasonably fast, but larger environments were not feasible as the time to construct the graph was excessive. To allow for greater precision of

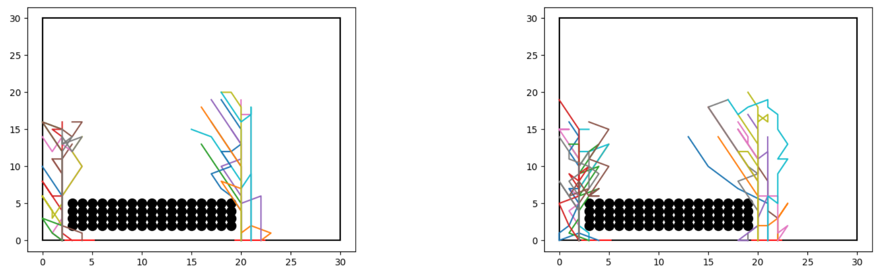

Figure 4. Example runs without a source of panic (left) and with a source of panic (right) included. The more erratic movement as a result of the panic is clearly seen in the right part of the figure.

agent movement, it would be preferable to optimize the navigation graph generation or at least cache it for repeated use.

Future work could explore alternatives for the navigation graph that would allow for the agent positions to be points instead of real numbers. Additionally, the agent paths could be curves parametrized by continuous time parameters, which would allow for continuous time instead of discrete time steps. Another thing that could be explored in further work is the optimization of clustering algorithm and contagion of emotional preferences, so we could test larger number of agents.

CONTRIBUTIONS. **AS** worked on the code (environment representation, agent movement, density calculation, code optimization) and on the report, **TZ** worked on the report and its editing as well as proofreading, and oversaw project management, and **ELG** worked on the code (contagion, navigation graph, clustering, visualization) and on the report.

Bibliography

1. Wu Y, Huang X, Tian Z, Yan X, Yu H (2024) Emotion contagion model for dynamical crowd path planning. *International Journal of Network Dynamics and Intelligence* 3(3):100014.
2. Traag VA, Šubelj L (2023) Large network community detection by fast label propagation. *Scientific Reports* 13.
3. Organick E (1966) *A FORTRAN IV Primer.* (Addison-Wesley).