XCrowd: A Realistic Crowd Simulation Tool for Efficient Movement Management

Jan Appel, Andreas Weiler

Institute of Computer Science, Zurich University of Applied Sciences (ZHAW), Winterthur, Switzerland

Abstract

This demo paper introduces XCrowd, a solution aiming to bridge the gap between theoretical crowd movement optimisation and real-world operations. By simulating crowd movements on graph networks, XCrowd provides an efficient tool to evaluate crowd behaviour under specific conditions and, for example, validate the efficiency of a certain environment layout. We discuss the methodology, implementation, and potential implications for crowd movement management, as well as outline future directions, including scalability, integration with scheduling models, and addressing real-world dynamic factors.

Keywords

Real-Time Crowd Simulation, Graph Networks, Crowd Movement Optimization

1. Introduction

Events like Music Festivals, Sport Events, or Political Events are usually hosting more than thousands of visitors. Some events have become key destinations for leisure and entertainment, underscoring the importance of efficient visitor scheduling. Visitor Attraction Management is a extensive research area. Leask *et al.* [1] present a survey about different aspects like Visitor motivations, preferences, expectations, and segmentation. However, existing theoretical visitor models, may fall short in accounting for dynamic crowd movements and real-world disturbances within defined environments and events.

Bridging this gap, XCrowd seeks to provide a solution, employing crowd simulation on graph networks to provide a realistic and adaptable, yet computationally feasible, approach to crowd movement management. We introduce XCrowd, detailing its methodology, implementation, and potential implications for crowd movement management. By offering a tool to evaluate crowd behavior under specific conditions, XCrowd aims to validate the efficiency of environment layouts and scheduling models, as well as the accuracy of theoretical scenarios, bringing a different perspective to the intersection of theoretical crowd movement optimization and real-world operations.

Nowadays, almost any person is trackable by GPS data within a smartphone, wearable, or any other device. Therefore, we are able to know the coordinates of any person in a defined environment. In this presented approach, we currently simulating the GPS signals of persons within a pre-defined environment. However, this could be easily exchanged by the real-time GPS data of mobile devices by single persons or groups of persons.

2. Approach

XCrowd is designed as a client-side web application, offering real-time simulation capabilities with the aid of WebGL for efficient rendering and cross-platform support. The core of the simulation relies on a linked graph structure which effectively models the layout and pathways of pre-defined environments. As early performance tests confirmed, a modern personal computer is capable of simulating several thousands of visitors in real-time. To ensure efficient path finding, XCrowd employs a modified version of Dijkstra's algorithm [2] that takes into account the current capacity of edges, allowing visitors to navigate within the environment and avoid congestion in a manner reflective of real-world crowd dynamics. The application utilizes a real environment map as its foundational base, providing a realistic reference for the graph network. Furthermore, XCrowd envisions future integration with real-world environment semantics, enhancing the accuracy and applicability of the simulation to diverse event scenarios.

2.1. Implementation

The core of the simulation is based on a graph representation of the environment, realised as a linked graph of `MapNodes`, which represent geographical locations in the environment, such as restaurants, shops, crossroads, or simply points along a pathway, and `MapEdges`, which represent the connections between these locations, i.e., the pathways. Each `MapNode` knows its properties such as its coordinates or the type of location it represents, as well as which `MapEdges` are connected to it. Each `MapEdge` knows its properties such as its length, the `MapNodes` it connects, or its capacity.

Visitors of the events are represented as `Persons`. When they are created at a `MapEntrance`, which is a special kind of `MapNode` that allows `Persons` to enter and exit the environment, they are assigned a list of stops they wanted to visit. Currently, these are random locations within the environment and serve as a placeholder for visitor schedules. Upon initialization, the `Person`'s route to the first stop is computed using a modified version of Dijkstra's algorithm.

Each simulation step, each `Person` moves along its current `MapEdge`. Once it reaches a `MapNode`, three possible things may have occurred: Firstly, the `Person` may have reached its destination, i.e., the `Person`'s last stop. Usually, this is a `MapEntrance`. If so, the `Person` is removed. Secondly, the `Person` may have reached one of its stops. If so, the stop is removed from the `Person`'s list and a new route to the next stop is computed.
Thirdly, the `Person` may have simply reached a `MapNode` within its current route, in which case the `Person` checks whether the next `MapEdge`'s current capacity is greater than zero. If so, the `Person` enters the `MapEdge`. Otherwise, a new route to the next stop is computed. This way, the

Published in the Proceedings of the Workshops of the EDBT/ICDT 2024 Joint Conference (March 25-28, 2024), Paestum, Italy
appeljan@students.zhaw.ch (J. Appel); andreas.weiler@zhaw.ch (A. Weiler)
0000-0002-8385-2537 (A. Weiler)

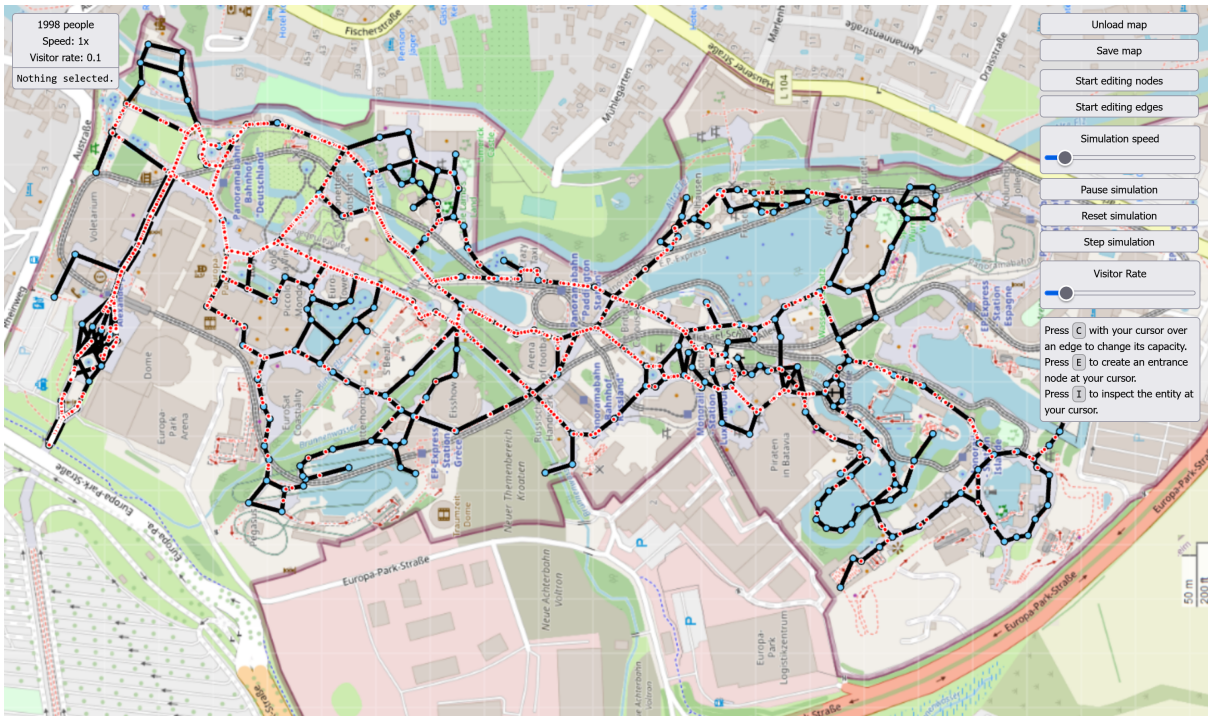

Figure 1: Overview of the XCrowd application interface. *Blue dots represent nodes, green dots represent entrance nodes, black lines of different thickness represent edges of different capacity, and red dots represent people. The park layout used for reference is from Europa-Park in Rust, Germany and was rendered from OpenStreetMap data [3].*

`Person` will try to avoid congested pathways and might find a way around the congestion much like a real-world visitor would.

The route computing algorithm is based on Dijkstra's algorithm. In addition to the `MapEdge`'s length, however, its currently remaining capacity is also a factor:

With

$$\mathit{capacityPunishment}(p, c, m) = 1 + \frac{a}{(\frac{\max(0, c-p)}{m})^b + 1},$$

p is the capacity needed by the `Person` (usually 1, but this allows for groups of people), c is the current capacity of the `MapEdge`, and m is the maximum capacity of the `MapEdge` used for normalization. a and b are arbitrary constants describing the maximum punishment and sensitivity, respectively. We chose $a = 100$ and $b = 100$. The cost of a `MapEdge` e is computed as

$$\mathit{cost}(e) = \mathit{distance}(e) \cdot \mathit{capacityPunishment}(e).$$

Since in some circumstances of high congestion, the `Persons` may not be able to find a route to their next stop and end up calling the route computing algorithm over and over, we introduced a cool down of 20 simulation steps for each `Person`. This way, the `Person` will simply wait for up to 20 simulation steps before trying to find a new route, which reduces the computational load dramatically.

Although under specific circumstances, a situation may occur where the `Persons` on a `MapEdge` are unevenly distributed, this poses no problem in the overall simulation as this effect is only noticeable on graphs with very few or very long `MapEdges` and vanishes as the graph becomes more detailed.

3. Results

The current status of the project showcases XCrowd as a functional web-based application, offering users real-time simulation and graph editing capabilities. A publicly accessible instance is available at https://xcrowd.cloudlab.zhaw.ch/, providing a platform for users to interact with the simulation and assess the project.

XCrowd serves as a supportive application for evaluating and enhancing environment efficiency by identifying potential bottlenecks and improving visitor flow within the simulated environment. While the current version represents a functional foundation, the project continues to evolve with future enhancements and features planned to further refine its capabilities and broaden its applicability to diverse environmental setups or other scenarios as can be seen in the Figures 1 and 2.

Figure 2 presents a case studie for a pre-defined music festival environment. The famous heavy metal festival Wacken increased its visitor amount from 800 in 1990 to 100.000 in 2022 and 61.000 in 2023. Therefore, the management of the crowd movement becomes more and more crucial. By using the XCrowd application we are able to simulate the movement of the pre-defined amount of visitors between the different spots, stages, and other points of interest. By defining the weight of the single edges between the spots, we can also simulate how many persons would be able to use this path. Hereby, it would be possible to evaluate emergency exits and the paths to them.

Figure 2: Another use case of the XCrowd application interface. *This map represents the festival environment of the Wacken Festival 2023. We can simulate the movement of the visitors within the environment.*

4. Vision

To accommodate larger environments, there is potential to shift towards server-side simulation or even make use of distributed computing. This would enable the simulation of more detailed scenarios as well as larger crowds and more complex environment layouts, ensuring scalability to meet the demands of diverse environments.

We are also plan to evaluate further work with an extended version of XCrowd, which is specialized for theme parks, to identify features such as daily schedule planners for theme park visitors. By combining XCrowd with these models, the application could offer insights not only into crowd dynamics but also into the overall visitor experience, including the effectiveness of their schedules or the park's resource allocation.

Enhancing the realism of the simulation involves incorporating unexpected events such as changes in ride availability, or variations in visitor behavior. Factors such as weather conditions, the need for food or restroom breaks, and potential emergencies could be integrated to create a more realistic and adaptable simulation, providing a broader spectrum of scenarios for evaluation.

Future iterations of XCrowd should also focus on improving the user interface and experience. This includes working on the simulation customization options, the map editing tools, and the inspector capabilities, making the application more intuitive and user-friendly for event managers and planners.

To enhance compatibility and streamline the workflow, XCrowd could support common standards and allow direct import for data such as the events's expected visitor numbers, opening hours, or geographical layout. This would facilitate the integration of real-world event semantics and enable users to work with a wider range of data sources seamlessly.

XCrowd offers a work-in-progress and a vision for the future of crowd movement management evaluation by combining theoretical models with more realistic crowd simulations. The application provides valuable insights into visitor behaviors and event dynamics, paving the way for more informed decision-making and enhanced visitor experiences. By integrating recorded or live GPS data, which of course requires the confirmation of the visitor, of the visitors the application gets even more powerful. It would be possible to simulate past crowd movements and detect special cases or abnormal behavior. Furthermore it would be possible to reconstruct single paths from monitored visitors or groups.

References

[1] A. Leask, Visitor attraction management: A critical review of research 2009–2014, Tourism Management 57 (2016) 334–361. URL: https://www.sciencedirect.com/science/article/pii/S0261517716301029. doi:https://doi.org/10.1016/j.tourman.2016.06.015.

[2] F. Schulz, D. Wagner, K. Weihe, Dijkstra's algorithm online: an empirical case study from public railroad transport, ACM J. Exp. Algorithmics 5 (2001) 12–es. URL: https://doi.org/10.1145/351827.384254. doi:10.1145/351827.384254.

[3] O. contributors, Planet dump retrieved from https://planet.osm.org, https://www.openstreetmap.org, 2017.