University of Sheffield

Sketching for Real-time Control of Crowd Simulations

Luis Rene Montana Gonzalez

A report submitted in partial fulfilment
of the requirements for the degree of
Doctor of Philosophy

The University of Sheffield
Faculty of Engineering
Department of Computer Science

May 13, 2021

Declaration

I, the author, confirm that the Thesis is my own work. I am aware of the University's Guidance on the Use of Unfair Means (www.sheffield.ac.uk/ssid/unfair-means). This work has not been previously been presented for an award at this, or any other, university.

Name: Luis Rene Montana Gonzalez
Date: 30th October 2020

Acknowledgements

I would like to express my gratitude to all the people who have made this thesis possible. I would like to thank my supervisors Dr.Steve Maddock and Dr. Paul Richmond for all the guidance and support throughout my PhD. A special thank you to my family and friends for being a constant source of support throughout all these years. Finally, I would like to thank CONACYT for sponsoring this research.

Abstract

Controlling the behaviour of a crowd simulation typically involves tuning of a system's parameters through trial and error, a time-consuming process relying on knowledge of a potentially complex parameter set. Numerous graphical control approaches have been proposed to allow the user to interact with a simulation intuitively. This research investigates the use of a real-time sketch-based approach for crowd simulation control. This is done by modifying the environment of the simulation. Users can create entrances/exits, barriers and flow lines in real-time on top of an environment. This process requires a data structure to represent the environment and navigate the crowd through it. Two alternatives are presented: grid and navigation mesh. A detailed comparison shows that the navigation mesh is a more scalable approach since it uses less memory, has a similar pathfinding time, and is a better structure to represent the environment than the grid.

The thesis also presents extensions to the sketch-based approach in the form of novel control tools, including storyboards to define the journey of the crowd, a timeline interface to simulate events through the day, and a sketch-based group storyboard to link behaviours and paths to be followed by a group. These tools are used to create two complex scenarios to exemplify possible applications of the sketch-based approach. The work on timelines also raises a new problem for an approach that dynamically modifies an environment in real-time which is 'when does the crowd know about the change?' Some initial solutions to how this should be handled are presented.

The sketch-based system is evaluated by comparing it to a validated commercial system called MassMotion. The comparison takes into account the plausibility of the simulation and usability of the user interface. A user study is carried out to evaluate the graphical user interface of both systems. Formal evaluation methods are used to make the comparison: the benchmark suite 'steersuite', an adapted version of the Keystroke-Level Model (KLM) and the System Usability Scale (SUS). The results show that the sketch-based approach is faster and easier to use than MassMotion, but with fewer control options. An implementation of the sketching interface in a Virtual Reality environment is also considered. However, when compared to the desktop interface using a proposed adaptation to KLM for VR, the results show that sketching in a VR environment is slower and less accurate than the desktop version.

Publications

- Gonzalez, L. R. M. and Maddock, S. Sketching for Real-time Control of Crowd Simulations. In Computer Graphics and Visual Computing (CGVC). The Eurographics Association, 2017
- Gonzalez, L. R. M. and Maddock, S. A sketch-based interface for real-time control of crowd simulations that use navigation meshes. In Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 1: GRAPP, pages 4152. INSTICC, SciTePress, 2019
- Gonzalez, L. R. M. and Maddock, S, A Sketch-based Interface for Real-time Control of Crowd Simulations that Use Navigation Meshes, Journal of Virtual Reality and Broadcasting JVRB (accepted)

I also contributed to the following publications, although these are not part of the thesis contributions:

- Charlton, J., Gonzalez, L. R. M., Maddock, S., and Richmond, P. Fast Simulation of Crowd Collision Avoidance. In Gavrilova, M., Chang, J., Thalmann, N. M., Hitzer, E.,and Ishikawa, H., editors, Advances in Computer Graphics, Lecture Notes in Computer Science, pages 266-277. Springer International Publishing, 2019
- Charlton, J., Gonzalez, L. R. M., Maddock, S., and Richmond, P. Simulating crowds and autonomous vehicles. In Transactions on Computational Science XXXVII, pages 129-143. Springer, 2020

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Pedestrian simulations have numerous applications, including films, video games, training, urban planning, and emergency evacuation simulations. With increases in computing power, bigger crowd sizes and more complex pedestrian behaviour are possible. However, such simulations include many parameters to control aspects such as individual and collective pedestrian behaviour and constraints in the environment, such as barriers to control pedestrian flow. To tune all these parameters, complex interfaces are used which require technical knowledge to understand.

An alternative approach is to use graphical interaction to facilitate crowd control and to enable non-expert users to modify pedestrian behaviour. For instance, a user could select a group of pedestrians using a mouse or sketch a curve in the environment to indicate the direction of movement for pedestrians. Crowd movement can then be controlled directly by interacting with the pedestrians or indirectly by modifying the environment rather than by altering lots of parameters. Examples of non-expert users include police planning events involving large crowds; and building designers defining the best layout of a venue to improve the crowd flow.

Sketching to control crowd simulations has not been extensively explored in the research literature. Previous work includes sketching the spawning location of pedestrians and sketching a path to be followed by them [129], sketching flow lines to guide the movement of the crowd [73, 132], sketching behaviour areas in the environment [66], and using sketching to create and move group formations [46, 54, 4].

However, previous work has not used sketching to modify the environment. Jin et al. [73], Patil et al. [132] updated the underlying vector field to guide the crowd, but it is not possible to change the environment. Hughes et al. [66] used sketching to create the navigation mesh representing the environment, but it cannot be updated to add more elements after its creation. Group formation and movement work does not use a data structure since the environment is not represented. This limits the crowd to follow the sketched path without interacting with the environment. With regards supporting data structures, Patil *et al*'s work [132] relies on a grid to guide the crowd, whereas Hughes *et al*'s work [66] depends on a navigation mesh. There is no research comparing these two data structures. Also, sketching has not been extended to create more complex features. The idea of creating a sketch-based storyboard to define the journey

of the crowd was mentioned as future work in [66], however, no further work was found. This thesis will address each of these issues.

1.2 Aims

The research aims to investigate the use of sketching to graphically control crowd simulations. This method involves directly drawing in the environment to alter pedestrian behaviour and to modify the environment itself. It is essential to distinguish between computer-based sketching, where users can freely draw any shape or curve, and rubber-banding, where users stretch lines between points fixed by mouse clicks. This research focuses on sketching.

A data structure is required to represent the environment and support the crowd simulation. The sketch-based approach needs to update this data structure to modify the simulation environment. The two most commonly-used data structures are considered in this thesis: grid and navigation mesh. These structures are compared to find the best approach to support the simulation and the sketching approach.

Current work on sketch-based crowd control can be extended to simulate more aspects of a simulation. An example of these aspects may be the itinerary followed by the pedestrians and changes in the environment over time. Chapter 2 discusses in detail the limitations of the graphical crowd control approaches. This thesis proposes several ideas to overcome these limitations. The ideas include sketch areas and behaviours to give the user more control; create a sketch-based storyboard to define the journey followed by the pedestrians; select groups by sketching a shape around them; and add simulation time to transition between sketch constraints based on time. These proposed novel ideas and current work extensions are fully described in Chapter 3 and 4.

Previous sketch-based crowd control interfaces have been compared to conventional interfaces [129, 4]. However, the evaluation was not made following formal system evaluation approaches. The sketch-based proposed in this research is compared against the validated commercial system called MassMotion. The comparison is made in terms of plausibility of the simulation and usability of the user interface. Formal human-computer interaction (HCI) evaluation techniques are used to assess these aspects.

The thesis addresses the following research questions:

1. Is sketching an intuitive control approach? Can sketching be used to reduce the time for tuning a crowd simulation? What are the benefits of a dynamic update process rather than an approach that involves stopping the simulation, altering the parameters and starting the simulation again?

2. What components of a simulation control system need to be altered to better support a sketch-based approach? For example, which underlying scene representation best supports a sketch-based approach?

3. Is sketching better suited to creation rather than real-time control? For example, how does dynamically changing a system affect the behaviour of agents, and when do they become aware of any changes to their environment?

4. Is it feasible to implement sketching in a VR environment? How does it compare to the desktop interface?

1.3 Contributions

The novel contributions of the research are as follows:

- A navigation approach to support sketching for controlling crowd simulations. Two underlying data structure are explored: grid and navigation mesh (Chapter 3). This work answers the second research question and produced two publications [43] and [44].
- Storyboarding, events and group control (Chapter 4). These features offer more control flexibility by mixing sketching, clicking and more complex interfaces. This work answers the second question and was published as part of work in [44].
- Dynamic environment knowledge (Chapter 4). A problem that arises from dynamically changing the environment is identified, and some initial solutions are proposed. This work answers the third research question and produced a paper that has been accepted for the journal JVRB (Journal of Virtual Reality and Broadcasting)[1].
- Comparison of the sketching approach with a commercial system (Chapter 5). This work evaluates sketching control by comparing it with the validated crowd simulation software MassMotion through a user study. Traditional human-computer interaction (HCI) techniques, such as the Keystroke Level Model (KLM) [20] and System Usability Scale (SUS) [18] are used to analyse the results of the study. This work answers the first research question.
- VR implementation (Chapter 6). The sketching approach is implemented in VR to explore its advantages and disadvantages compared to the desktop interface. An adapted KLM version is proposed to evaluate the VR interface. This work answers research question four.

1.4 Thesis Structure

Chapter 2: Literature Review

This chapter provides an overview of related work on crowd simulations, agent-based modelling and data structures used to represent virtual environments. Graphical crowd simulation control approaches are discussed and classified depending on how users interact with the simulation. Moreover, commercial systems are analysed emphasising the complex interfaces needed to set up the simulation. Lastly, multiple methods to evaluate the realism of simulations and the usability of graphical user interfaces are covered.

[1] Gonzalez, L. R. M. and Maddock, S, A Sketch-based Interface for Real-time Control of Crowd Simulations that Use Navigation Meshes, Journal of Virtual Reality and Broadcasting JVRB (accepted)

Chapter 3: Sketch-based control and navigation methods

This chapter describes the sketching work implementation, the overview of the system and the use of a grid and navmesh as navigation methods. A comparison of these two approaches, considering environment representation, memory usage and path-finding time, is provided at the end of the chapter.

Chapter 4: Extending the sketch-based approach

This chapter explores additional and more complicated control options that not only involve sketching. Storyboards define the crowd of the journey by clicking the areas of the environment to be visited. A timeline interface allows the creation of events during a day, such as creating barriers and closing an exit. The group control feature directly modifies the agent parameters instead of the environment. Users select a group of agents and create a sketch-based storyboard by sketching and linking paths and behaviours. A couple of complex scenarios exemplify possible applications of the sketching approach. Additionally, a new problem arising from the dynamic interaction with simulation is identified.

Chapter 5: Comparing the sketching approach with MassMotion

This chapter evaluates the sketching approach by comparing it with the validated commercial system MassMotion. The comparison is made in terms of plausibility of the simulation and the usability of the user interface. Multiple tools such as 'steersuite' [164], KLM [20] and SUS [18] are employed to carry out the evaluation.

Chapter 6: Virtual Reality

This chapter explains the VR interface and the special guidelines followed to implement it. The advantages and disadvantages of sketching in a VR environment are discussed, and both interfaces are compared using an adapted KLM version and with a simple sketching test. Last, other input devices are briefly explored to assess their use for sketching, control the camera and interact with the graphical user interface.

Chapter 7: Conclusions

This chapter presents the main conclusions of the research and identifies possible areas of future work.

Chapter 2

Literature review

This chapter reviews the related work on crowd simulations and graphical control approaches. Section 2.1 covers the existing agent-based modelling techniques and levels of control to modify the agent behaviour. Additionally, the section studies the methods used to represent virtual environments, focusing on navigation meshes. This section sets the context of the main components of crowd simulations, agent model and environment, which can be modified to control the behaviour of the crowd. Understanding how these components work is crucial to create an intuitive graphical control approach which is the main focus of this work. Section 2.2 analyses commercial systems and their limitations regarding user interface and simulation interaction. Section 2.3 discusses graphical crowd simulation control solutions and classifies them according to the way users interact with the simulation. This categorisation provides the context to define where the approach proposed in this research fits. Finally, Section 2.4 presents methods to evaluate the realism of simulation models and the effectiveness and usability of graphical user interfaces. Some of these approaches are used in Chapter 5 to evaluate the agent model and the sketching interface proposed in this thesis.

2.1 Crowd simulation

A crowd simulation could be defined as the computational model of pedestrian motion and dynamics within a virtual environment [87]. This crowd movement and pedestrian behaviour can be very diverse and depends on several factors. However, some patterns can be identified, such as people avoiding each other and individuals walking towards a destination following the optimal path. Crowd simulations try to model these characteristics realistically.

Several techniques have been used to create such simulations. Martinez-Gil et al. [112] categorised them based on the pedestrian dynamics modelling into five groups: Mechanics-based, Cellular automata, Stochastic, Agency and Data-driven. The mechanics category uses mathematical equations and physics to describe the movement of the pedestrians; social forces and continuum mechanics are included in this group. Cellular automata models discretise the environment to define the crowd motion with finite automata. In the stochastic model, the calculation of crowd dynamics involves arbitrary decisions and stochastic processes. Agent-based models represent pedestrians with agents capable of making autonomous decisions. The

data-driven category uses real crowd data to model the behaviour of pedestrians.

This categorisation by Martinez-Gil et al. [112] does not consider hybrid models - a typical example is an agent-based model using social forces to calculate agent movement.

Yang et al [190] provides a comprehensive review of existing crowd modelling techniques classified into macroscopic, microscopic and mesoscopic crowd models. The macroscopic approach focuses on global behaviours of the crowd as a whole; pedestrians cannot be identified, and the movement is described by flows and densities [154]. Microscopic models control the motion and interaction of pedestrians at an individual level. Mesoscopic models are simulations taking into account interaction between human and computer, pedestrian dynamics and social factors.

In contrast, Pelechano et al. [136] and Duives et al. [32] classified pedestrians models into either macroscopic and microscopic. Hughes [64, 65] proposed an example of a macroscopic model, where the flow of pedestrians is modelled with a continuum theory and the crowd is defined by two attributes: density and velocity. individual pedestrian behaviour is not controlled in this method. Treuille et al. [172] extended this continuum theory by presenting a model to integrate local avoidance and global navigation. However, this approach is not suitable for complex environments in which congestion may occur, causing undesired behaviour. Barnett et al. [12] suggested a solution for this issue by getting the capacity of each path and adjusting the pedestrian trajectories to alternative routes. Similarly, Jiang et al. [72] handled complex environments creating a hierarchical structure formed by blocks to represent different areas of them. Each block is discretised into a grid and linked to determine the crowd movement. Lastly, Narain et al. [121] developed a system to handle the behaviour of highly populated crowds, proposing a novel inter-agent avoidance algorithm that does not depend on the number of agents. A disadvantage of macroscopic models is that they cannot simulate local interactions.

This thesis focuses on a microscopic approach. The next section overviews the literature on agent-based modelling, which is a common microscopic approach.

2.1.1 Agent-based modelling

Agent-based modelling is the most used approach to simulate virtual crowds. Each agent calculates its movement based on a set of rules, behaviours or forces. This individual behaviour is computationally expensive since the resulting motion of every agent has to be calculated. Consequently, a trade-off between the complexity of the agent model and crowd size exists [9]. Agent-based modelling offers the advantage of emergent global behaviours such as lane formation and queuing.

Reynolds' pioneering work [144] simulated a flock of birds by controlling the agents with a set of rules. Three rules determine the movement of each agent or boid: Collision avoidance, which prevents flock members from colliding with each other; Velocity matching, where every agent tries to align with the speed and orientation of neighbours; Flock centring, where each boid moves towards the centre of the group. Additional rules or steering behaviours, such as seek, flee and pursuit, were defined in subsequent work by Reynolds [145, 146]. Rodrigues et al. [150] used a space colonisation algorithm to generate steering behaviours such as pursue, escape, surround and group formation. This method leaves markers in walkable areas and adds

paths connecting them. A tree path is formed from the agent position to its destination. The individual agent movement produces complex global behaviours that are complicated to predict and control. Anderson et al. [5] controlled the flock of Reynold's work by adding spatial and shape constraints. Similarly, Ho et al. [61] created a flexible flocking model that adjusts the shape of the formation depending on the environment and the path followed.

Helbing proposed a different approach to model agent dynamics using 'social forces' [56, 57, 58] to determine the movement of the agents. Three main effects result from the interaction of pedestrians with the environment and with each other: goal, collision avoidance and attraction. This approach was extended and applied to simulate emergencies [59, 16]. Lakoba et al. [97] improved Helbing's work avoiding agent overlapping and obtaining better results after modifying the parameters that caused counter-intuitive results in small crowd simulations.

Sociological aspects have been considered when modelling pedestrian behaviour [120, 130, 194]. Musse and Thalmann [120] divided agents into groups with a set of attributes such as interests, emotional status and relation with other groups. These parameters, combined with a set of rules, determine if an agent stays in its group or joins a new one. Pan et al. [130] simulated human and social behaviours during evacuations. The agent model is divided into three categories: locomotion, steering and social. Locomotion behaviours are simple actions such as walk, run, stop and turn. Steering behaviours – seek, follow, collision avoidance – are used to navigate throughout the environment in conjunction with locomotion. Social responses, competitive, queuing and herding behaviour, emerge from the interactions of groups. Yu and Terzopoulos [194] presented an agent model to create advanced behavioural animations with social interactions between pedestrians. A decision-making network combines probability, decision and graph theories to select the action to the corresponding situation. The behaviours implemented by the system include emergency response, acquaintance, partnering and collision avoidance.

Similarly, several agent-based models have taken into account psychological factors. Ulicny and Thalmann [174, 175] created a model useful for VR training in emergencies. Agents select their behaviour based on a set of internal psychological and physiological factors such as fear, mobility and injuries. Pelechano et al. [135] presented a solution to simulate agents in high-density crowds based on situations and psychological and physiological traits. These attributes include impatience, panic and energy level. Rao et al. [143] suggested an approach to control the local movement of agents in a real-time environment. The local motion model includes psychological traits such as panic, impatience, memory and orientation. Agents are driven by forces: repulsive to avoid static and dynamic obstacles (including agents); and steering to guide pedestrians to the destination.

Li et al. [108] introduced a model in which agents decide their personality, needs, and interests. These psychological factors are defined using Maslow's Hierarchy of Needs and the OCEAN model [184]. Durupinar et al. [33] extended the system presented in [135] by integrating the OCEAN personality model. Each personality trait is mapped to behaviour parameters such as leadership, communication, panic and personal space, among others.

Pelechano et al. [134] incorporated roles within pedestrian groups and communication between agents. Additionally, multiple parameters such as arousal, fatigue, and thirst are

considered to calculate the behaviour of a pedestrian. This work is not the only model dividing the pedestrians into groups. Musse et al. [118] and Musse and Thalmann [117] suggested a hierarchical structure consisting of a crowd, groups and agents. The behaviour of the agents is controlled at group level in three possible ways: externally guided, scripted and autonomous by following rules. Lu et al. [110] followed the same structure as [118] but used a finite state machine and a decision tree to decide the actions of the groups. A distinct group model was introduced by Li et al. [107]. Each crowd consists of a leader and followers with flocking behaviour. Qiu and Hu [142] created an agent group model taking into account intra-group structure and inter-group relationships to model a crowd simulation.

Particular types of agents have been suggested. Yeh et al. [191] introduced the concept of 'composite agents'. This model consists of a regular agent and proxy agents to influence other agents. This interaction enables the modelling of complex behaviours emerging from people's response to psychological and social factors - for example, aggression, social priority, authority, protection and guidance. Schuerman et al. [156] created a model called 'Situation agents'. These agents possess the ability to influence the movement and behaviour of nearby pedestrians by modifying their parameters. These agents are useful in deadlocks and group formations. Stvel et al. [166] presented 'Torso crowds', agents represented by a capsule to emulate the human body shape (from the top). These pedestrians rotate their body while moving through crowds.

A different approach to model the behaviour of the agents is to use data from real-world crowds. Lerner et al. [105] manually extracted agent trajectories from a video of real pedestrians. A database is created with the trajectories. Each agent is assigned a trajectory querying the database trying to find a similar example based on the surroundings. Lee et al. [102] created an agent model based on two-dimensional trajectories tracked from aerial view video of a human crowd. The model determines the motion of the agents taking into account the environment and nearby pedestrians. Kim et al. [95] presented an algorithm to model crowd simulations combining real-world data and existing agent models. This method extracts pedestrian dynamics from real pedestrian videos. These dynamics consist of the start position, preferred velocity and the collision avoidance rule. Then agent models such as social forces can be combined with the obtained dynamics to create the final agent trajectories.

A recent approach for agent navigation is the use of reinforcement learning with deep neural networks. Reinforcement learning uses an agent, its environment and their interaction to find the most desired agent state. It can be used to find agent trajectories from their current position to the goal. Lee et al. [100] proposed a deep reinforcement learning method for agent navigation. The approach uses reinforcement learning to solve a decision making problem. This is defined as a Markov decision process formed by: States, represented by the agent location, goal and visual sensory inputs; Actions or decisions; Probabilities of moving between states given an Action; and Rewards, the desirability of the states. This approach does not need a set of predefined rules or any parameter tuning. This system presents two main limitations. First, agents cannot differentiate between other agents or obstacles. Second, the resulting path followed by the crowd is not always the shortest.

Several systems where the environment is divided into layers or includes information to facilitate pedestrian navigation have been developed. This information includes agent position;

objects and behaviours associated with a certain area [35]; communication between agents [14]; or information for agent decision-making [109]. Banerjee et al. [9] divided the surface into three layers: Occupancy to avoid collisions between agents; Obstacles for environment collision avoidance; and Path-plan for global navigation. Similarly, Tecchia et al. [171] used layers for inter-agent and environment collision, but two additional layers were implemented to produce complex behaviours. A further division was suggested by Goldenstein et al. [42]. Three layers determined the pedestrian movement. First, the local layer is used for reactions such as obstacle avoidance. Second, the environment keeps track of the agents surroundings and collision detection. Last, the global layer guides pedestrians towards their goal. Lu et al. [110] followed a similar approach using layers for inter-agent collision, environment obstacle avoidance, and global navigation. This work supports dynamic obstacles separating static and dynamic layers.

Shao and Terzopoulos [160] proposed an environment model with several maps: topological, represented by a navigation graph; two perception maps for static and dynamic objects; and two path maps for path planning. Lastly, Jiang et al. [71] created a hierarchical structure to represent multilayer virtual worlds. The hierarchy has three levels. The geometric level represents the 3D model of the environment. The semantic level divides the model into regions; describes their connectivity; and stores the height coordinates of the world. The application level uses the semantic information to generate maps storing the position of objects and agents and the computed path for each agent.

Agent-based simulations present the problem of homogeneity. Agents behave in the same way, and creating a different look for each pedestrian is a challenging task. To alleviate this problem, Rudomin et al. [151] suggested an approach to render crowds with varying shapes, clothes and behaviours. The characters are generated using body part templates such as legs, head and torso. The combination of these parts results in heterogeneous crowds. Gu and Deng [45] addressed the problem of behaviour homogeneity by assigning different motion styles to the agents. The motion of several behaviours was captured and stored in a database. These motions include walking, running and waiting. The database is queried at runtime to obtain the corresponding animation.

Two levels of control can be observed in pedestrian simulations. Local motion defines the short-range movement of the agents, considering their immediate surroundings. This level relates to the previously mentioned work using rules, behaviours and other factors. Global navigation guides the agents through environments where pedestrians could get stuck in local minima when using local rules.

2.1.2 Local motion

Obstacle avoidance and agent interaction are typical behaviours resulting from this level of control. Agent models with sensors or perception modules to detect the environment have been proposed. Shao and Terzopoulos [159] presented a model including aspects such as perception, cognition, behaviour, and motion. The perception module is capable of sensing ground height and static and dynamic objects. This information is used to analyse the situation, make a decision and behave accordingly. Sakuma et al. [153] modelled agent behaviours in a crowd based on the positional relations among nearby pedestrians. Each agent has a personal space

defined by two radii, one for a critical area and the other for a warning area. The avoidance response depends on which zone is invaded by other agents. Agents are equipped with a visual sensor to detect surrounding agents. Pan et al. [130] introduced a framework to simulate human behaviour in emergency evacuations. Agents are equipped with sensors using ray tracing to detect the type and distance of nearby objects. This sensed data, a set of rules and psychological factors produce the agent actions.

Kapadia et al. [84] suggested the use of scalar and vector fields to sense the environment within a radius with agents as the origin. Three phases determine the motion of pedestrians. The perception phase detects objects in the world and predicts the position of nearby agents from their speed and direction. The collected data is fed to the affordance phase to assign a value to all the possible actions that can be performed. The last stage uses these values to select the optimal action based on agent goals. Ondřej et al. [127] used a vision-based model to avoid collision between pedestrians. Agents perceived the environment through an OpenGL camera positioned at eye level. This camera is used to detect moving objects and evaluate their velocity based on size change. Lastly, the time to collision is calculated to adjust the position of the agent accordingly.

Agent collision avoidance is a crucial feature of local motion and, therefore, has been widely explored. Pettr et al. [141], Olivier et al. [125] conducted an experiment to observe how humans interact to avoid collisions. The interaction is divided into three phases: observation, reaction and regulation. As a result of the study, a metric named *minimum predicted distance* determines the need to adapt trajectories to prevent a collision.

Fiorini and Shiller [36] proposed a seminal approach named *Velocity Obstacle (VO)*. In this model, a cone of velocities that cause a collision is generated for each agent. The apex of the cone is the position of the agent bounded by two tangents to the obstacle radius. All the velocities outside the VO are guaranteed to not collide with the obstacle. However, this method may produce an oscillatory movement when agents continuously change between the original and a collision-free velocity. Van den Berg et al. [178] addressed this issue with a new model called *Reciprocal Velocity Obstacle (RVO)*. This approach assumes that other agents are going to react similarly to avoid the collision. In contrast to VO, the new velocity is the average of the current velocity and one outside the cone. Collision-free trajectories are not guaranteed for multi-agent simulations.

Van den Berg et al. [179] extended their previous work [178] by limiting the set of velocities obtained by the RVO solution. *Optimal Reciprocal Collision Avoidance (ORCA)* uses linear programming to select the closest possible velocities to the desired agent velocity. ORCA ensures collision-free movement of independent agents. Charlton et al. [23] implemented a GPU-optimised version of ORCA. The optimisation resides in balancing the GPU workload among the threads by subdividing the calculations. The simulation size in this model is limited by the large data requirements per agent to store the ORCA half-planes used to create permissible velocities. Figure 2.1 shows an 8-way crossing scenario with 10,000 agents using this approach. Charlton et al. [24] extended their previous work to incorporate autonomous vehicles to the simulation. Figure 2.2 shows a section of an ORCA simulation with 2,500 pedestrians and a few autonomous vehicles.

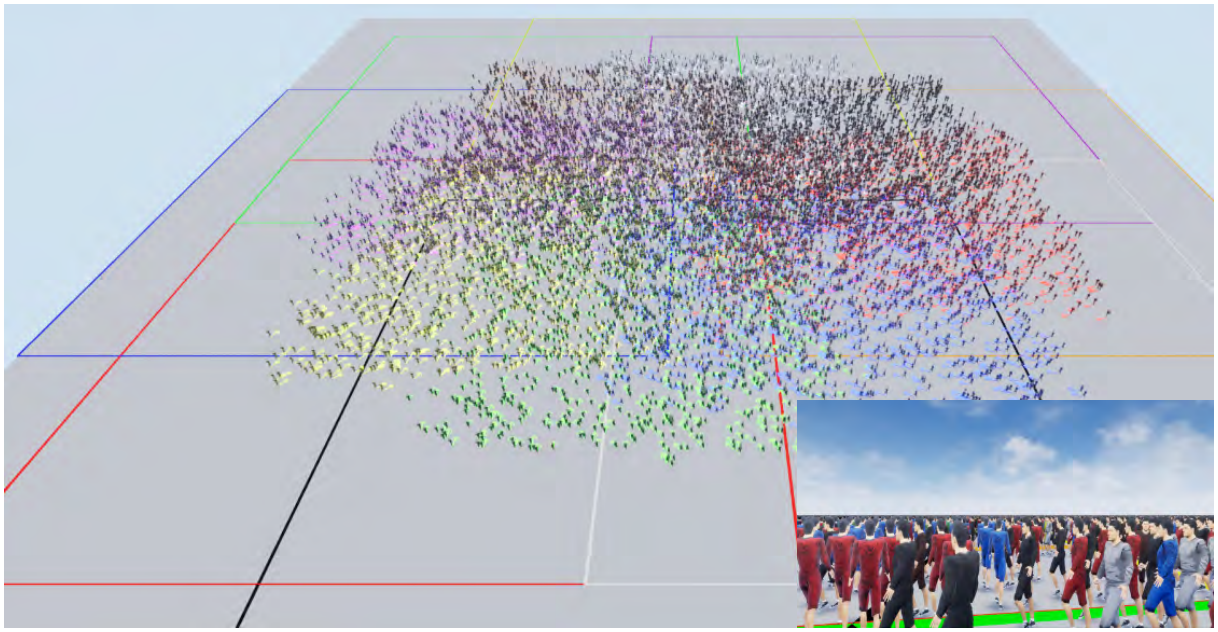


Figure 2.1: Crowd of 10,000 agents visualised in Unreal. Eight groups moving to the opposite environment end. Different colours are used for to identify each group. Top: scene view from above; An agent is highlighted with a green circle; Inset: view from the perspective of the selected agent

Figure 2.2: Zoomed-in section of 2,500 agents crowd visualised in Unreal. Four groups navigate in opposite directions, two moving from left to right and two heading between top and bottom. Top: scene view from above showing agents and autonomous vehicles; Inset: shows an eye-level view of the simulation

2.1.3 Global navigation

Global navigation is used to navigate agents through complex environments, where they could get stuck in local minima when using local rules. Different techniques have been proposed for global path planning. These methods include Roadmaps, Navigation graphs, Flow fields and Navigation meshes.

Roadmaps

Roadmaps represent the walkable areas by randomly sampling the environment. The sampled points are tested for collision and discarded if their location is an obstacle. The remaining points are connected, using straight lines, to their nearest neighbours. Links with detected collisions are removed, and the rest of the links form the roadmap [136]. Bayazit et al. [13, 14] suggested a simulation system with a roadmap capable of storing behaviour rules. The rules of each node determine the movement of the agents traversing them. Sud et al. [167] presented an algorithm for adaptive roadmaps taking into account dynamic objects. Links of the map are removed as obstacles interpose between two nodes. Deleted links are added again when the path between the nodes becomes collision-free. Guy et al. [49] created an algorithm to guide pedestrians following the path with least effort. A roadmap with weights assigned to each edge represents the environment. These weights represent the required effort to move through the corresponding link. The edges are dynamically updated as the velocity of the agents change. The algorithm calculates the path from the current agent position to its goal with the least effort.

Geraerts and Overmars [38] introduced a method to generate roadmaps that ensures finding a path between two points. Additionally, it creates shorter paths with no detours by adding nodes and edges. This approach also provides alternative routes for pedestrians. Paths with clearance are obtained by retracting them to the medial axis. Geraerts and Overmars [40] created 'Corridor maps', a novel approach to represent the environment and path planning. This method consists of a road map. Nodes correspond to a point in the environment, and edges represent a path connecting the nodes. The corridor map is a graph with clearance information stored in the paths. Agents move inside the corridor using potential fields by following attraction points which are located in the calculated path. Geraerts and Overmars [39] improved the corridor map approach by using graphics hardware to obtain a generalised Voronoi diagram to create the graph. Additionally, a search structure was defined to accelerate the path-finding process. Geraerts [37] developed a new structure to represent an environment and path planning derived from previous work on corridor maps Geraerts and Overmars [40]. The corridor size is explicitly defined and used to obtain paths with varying clearance. The inside of the corridor is triangulated to find the shortest path from the start to the destination.

Lamarche [98] suggested 'Topoplan', a method to create a topological and roadmap from a 3D environment based on a prismatic spatial subdivision. The constructed map takes into account parameters such as maximum slope, minimum height, and maximum step. The first step is to project the boundaries of each 3D polygon of the mesh on the xy plane. This projection creates a set of 2D polygons in the plane. Then, constrained Delaunay triangulation [27] is applied to the polygons. Each 3D environment polygon projects on multiple 2D triangles of the plane.

Every 2D triangle is associated with the polygons projecting on it. Then, a 3D triangle (cell) is created for every 2D triangle projecting them to their associated 3D polygons. Adjacent cells are connected to form zones. With zones defined, a new Delaunay triangulation is performed to get the final topological map. The roadmap is constructed by sampling the borders and edges of the topological map. The sampling is done trying to maximise the distance to the obstacles.

Jorgensen and Lamarche [77] proposed an algorithm to identify and classify regions of the environment. This method applies the same spatial subdivision as Topoplan [98]. The algorithm uses the topological plan to: identify outdoor and indoor areas; extract floors and stairs; and divide the floors into rooms and doorsteps. A hierarchical structure is defined by dividing the environment into four levels according to the identified zones. This hierarchy is used to improve path-finding quality.

Kallmann [81] developed an automatic approach to generate roadmaps. A Voronoi diagram is used as a reference to get the nodes of the map. Then, random points are selected and retracted to the Voronoi diagram. This process ensures maximum clearance for the nodes. The edges formed when the nodes are connected may be close to obstacles. This issue is solved by dividing the edge down the middle and retracting the new node to the Voronoi diagram. The resulting graph has straight-line edges causing unnatural agent paths. Circular blends are added to bypass connecting nodes and obtain better-looking trajectories.

Roadmaps present some disadvantages. First, since they are based on sampling, they may require many points to find paths in complex environments. Second, the paths obtained depend on the sampled points, which could result in unrealistic trajectories. Last, all the walkable areas of the environment are not represented by the roadmap.

Navigation graphs

A navigation graph consists of dividing the walkable areas of the environment into cells. After the decomposition, an adjacency graph is calculated to obtain adjacent cells which are linked with edges to create the graph [99]. Pettre et al. proposed a method to represent multi-layered and rugged environments with a navigation graph [138, 139, 140]. The search algorithm computes multiple solution paths to produce heterogeneous pedestrian trajectories.

Yersin et al. [192] presented work where information could be added to a navigation graph. The user can add a description or label graph nodes and assign them as goals to the crowd. Sud et al. [168] introduced a global navigation structure for path planning in crowd simulations. The structure called 'Multi-agent navigation graph' is computed using second-order Voronoi diagrams and can be used in dynamic environments.

Navigation graphs have similar disadvantages to roadmaps. The environment coverage depends on cell shape and size. Furthermore, many cells are required to represent complex virtual worlds, thus impacting the path-finding time. Lastly, the paths found may require smoothing to produce more realistic crowd movement.

Flow fields

Reynolds [145] was the first to propose the idea of flow fields. In this approach, the environment

is discretised into a two-dimensional grid with cells containing a vector force. Agents align their movement to the vector present in the cell corresponding to their position. This method presents the disadvantage that an accurate environment representation and fluid movements depend on the grid resolution causing poor scalability. Figure 2.3 shows an example of a flow field.

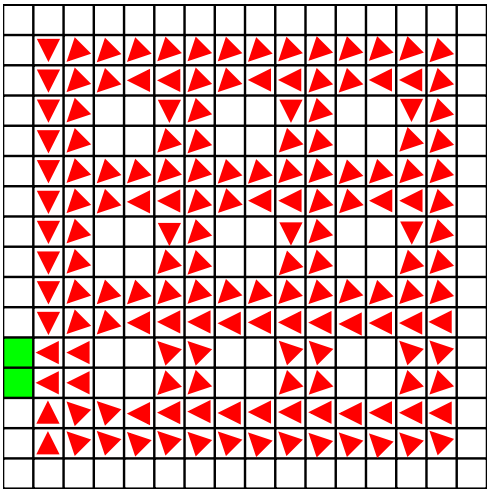


Figure 2.3: Flow field guiding pedestrians towards the bottom left exit

Several simulation systems have used this global navigation technique. Chenney [26] proposed a tool to create flow fields by connecting small areas of forces called 'Flow tiles'. Jin et al. [73] developed a system to change the underlying flow field by drawing arrows on the virtual world. Similar work was carried out by Patil et al. [132]. Users sketch lines on top of the environment to create 'guidance fields'. The system computes a final flow field based on the user inputs. Karmakharm et al. [90] used multiple layers of flow fields to guide pedestrians to different destinations. The force vectors and pedestrians are designed as agents to allow parallel computation using GPU hardware.

Navigation mesh

Snook [165] introduced the term navigation mesh in his work, creating a mesh of triangles to represent the walkable areas of a 3D environment. A navigation mesh is a decomposition of the environment into a mesh of convex polygons. Convexity ensures the free movement of the agents inside the polygons [29, 30]. According to Kallmann and Kapadia [82], a navigation mesh should have the following properties:

- Linear number of cells. A small number of nodes in the navigation mesh is required for fast path-planning algorithms.
- Quality paths. The paths followed by the agents should be realistic.
- Clearance. The generated paths should have enough separation from the obstacles.
- Robustness. The navigation mesh should be robust to describe the geometry of the environment.

- Dynamic updates. The mesh should dynamically adjust to changes in the environment.

Several techniques to generate navigation meshes from the geometry of the environment have been proposed. Van Toll et al. [180] used medial axis to create a navigation mesh for multilayer worlds. Connection zones link these layers. The medial axis is computed for each layer without taking into account the connections. Then, the links are opened to update nearby points and connect the layers to get the final medial axis. The points of the medial axis are connected to the nearest obstacle by a line to form the navigation mesh. A limitation of this method is that the layers must be planar to compute the medial axis. van Toll et al. [181] created a navmesh using Voronoi diagrams to support dynamic updates in real-time. The insertion and removal of points, lines and polygons are possible. The navigation mesh is updated locally in the affected areas. A disadvantage of this approach is that all obstacles (original environment and real-time insertions) must be partitioned into convex polygons. Some environments may need additional work to be suitable for this navmesh technique.

Delaunay triangulation is an alternative technique explored to construct navigation meshes. Kallmann et al. [83] represented a 2D environment with a mesh of polygons using constrained Delaunay triangulation. The system allows the insertion of points and edges as constraints to represent obstacles. Kallmann [78] extended his previous work [83] to add path planning. In this new method, obstacles can be added and removed during runtime. The A* algorithm is used to find the path between triangles. Kallmann [79] created a new structure for path planning based on the constrained Delaunay triangulation work [83]. Clearance information was added to the triangle edges to determine if an agent can traverse the triangle. The resulting paths have arbitrary clearance from the obstacles. This work was extended to add dynamic updates [80]. The navmesh created by these approaches is formed by triangles producing more nodes that polygon-based navmeshes. Even though this accelerates the navmesh generation process, it has an impact on path-finding calculations.

Hale et al. [53] suggested an approach to represent the walkable areas of an environment with a 2D navigation mesh. The environment is discretised into a grid; cells containing obstacles are discarded. Square regions are seeded in the grid to start forming the polygons. The square grows in every direction following the contours of the cell marked as obstacles. The convexity of the polygons is checked every iteration of the algorithm. The seeds grow until the walkable space is covered. This work was extended to allow the 3D decomposition of environments [52]. Neither approach supports real-time environment updates.

Oliva and Pelechano [123] presented an algorithm to create navigation meshes for 2D environments. The environment is represented as a single polygon that may have holes in it. This algorithm does not support dynamic updates and only works for 2D. The main idea is to check the convexity of every vertex in the polygon. If the angle is concave, a new edge is added to create two new convex polygons. Redundant edges are deleted to reduce the number of nodes. Oliva and Pelechano [124] introduced a system to automatically generate a navmesh from a 3D environment taking advantage of the GPU. The user defines four parameters: maximum step, maximum slope, character height and walkable seed. The first step of the process is a GPU voxelisation to obtain the walkable areas, which are classified into layers. Zones not accessible to the agents are discarded. A 2D high-resolution floor plan is obtained after refining the walkable

layers. The last step is to generate a navmesh with the floor plans and apply convexity relaxation to reduce the number of nodes.

Akaydın and Güdükbay [1] created a navigation mesh based on images. A top view image is used to obtain a grid describing the topology of the environment. Cells represent walkable areas or obstacles. This grid is divided into rectangular regions called clusters which are created by selecting a set of initial seeds. These seeds are expanded until all neighbours are occupied or non-walkable cells. The final clusters represent nodes of the graph. The cost of traversing an edge is the distance between neighbour nodes. The shortest path between every node is calculated. A vector field is constructed from these paths. Berseth et al. [15] generated a navmesh based on the curvature of the original mesh. The acceleration is calculated in each vertex from this curvature. Traversable areas are determined by a maximum acceleration threshold. The edges of vertices exceeding the threshold are marked and split. The modified mesh is re-triangulated to obtain the final mesh.

Recast[1] is an open-source software system used in video games to generate navmeshes for a given 3D environment automatically. Chapter 3 will give more details about the use of *Recast* in this research. Figure 2.4 shows a navmesh generated with *Recast*.

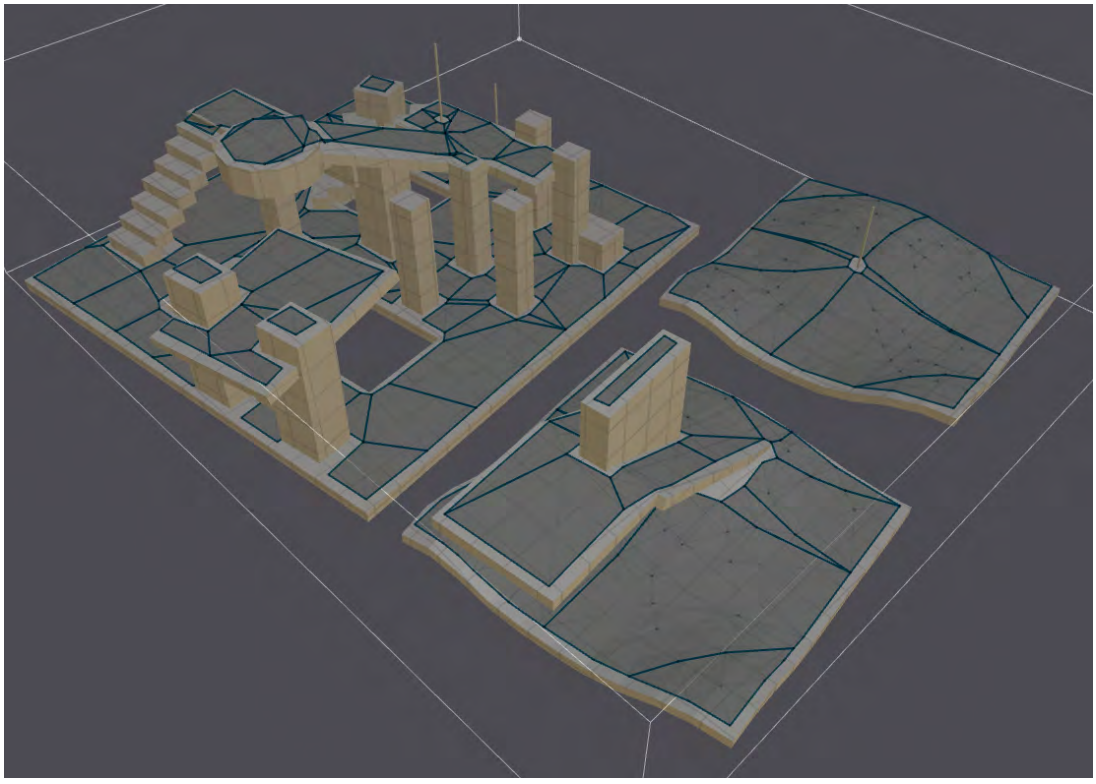


Figure 2.4: Environment represented by a navmesh generated with Recast. The navmesh is represented by the grey polygons formed by the blue lines.

This research implements and compares flow field (grid) and navmesh to find the most suitable and scalable approach to represent the environment. Section 3.4 provides a detailed

[1] http://masagroup.github.io/recastdetour/

comparison considering environment representation, memory usage and path-finding time.

2.2 Commercial Systems

This section briefly describes some of the existing pedestrian simulation software systems, mentioning the pedestrian navigation method used and some available user-configurable parameters to control the simulation. Challenger et al. [22] claim that available simulation tools can be used in different areas, such as security, transport and sports, but they do not show the reality and do not give definite answers. The purpose of the section is to highlight some of the issues that these systems present, specifically, complex user interfaces that are not intuitive for inexperienced users due to the large number of parameters to configure. An additional limitation is the inability to interact with the simulation in real-time.

Exodus [177] is a software tool used to simulate emergency evacuation situations and pedestrian dynamics. Three types of interactions are considered: pedestrian-pedestrian, pedestrian-structure, and pedestrian-environment. A rule-based approach determines the movement of the agents. Users can set multiple agent parameters including gender, age, height, weight, patience, mobility, agility and speed. The environment can be created with the tools provided by the interface or imported from third-party sources. It is divided into a 2D grid with nodes connected by links. Users can set different types of terrains and states by modifying the environment nodes. Figure 2.5 shows the graphical user interface of Exodus. The main window shows the layout of the environment; the dots represent nodes whose parameters can be configured as shown in the right side windows. This parameter tuning could be potentially laborious, and the final behaviour might be difficult to predict.

Simulex [162] simulates pedestrian motion and evacuation of a user-defined environment. Emergent behaviour, such as queuing and overtaking, is observed during the simulation. Multistorey buildings are permitted. The user specifies the position of the exits. Distance maps are created to calculate the distance from each location to the exit. These maps are used to guide pedestrians to the nearest exit. Agents have some configurable parameters including position, body size, speed and reaction time to alarms. Simwalk [163] models pedestrian flows and emergency evacuation behaviour. Fire and explosions can be simulated. Simwalk presents a graphical interface where users can create the environment and define objects such as ramps, escalators, stairways, service areas, destinations, attraction and danger areas. Multiple agent parameters can be tuned: speed, age, gender, height, priority and restriction of mobility. The agents are guided to their destination using potential fields.

MassMotion [122] is a crowd simulation and analysis tool. The user can create the environment using the graphical tools provided or by importing models from other software such as AutoCAD. An obstacle and an approach map represent the 3D world. These maps define the obstacle positions and the distance from every point to the exits. Pedestrians are modelled as autonomous agents driven by a social forces model. Figure 2.6 shows MassMotion's interface. A menu on the top has a few tabs with multiple objects to create in each one. Moreover, created objects are displayed on the left panel. Each object has a configuration window like the one shown in the middle of the figure. Such a complex interface provides many options to create a

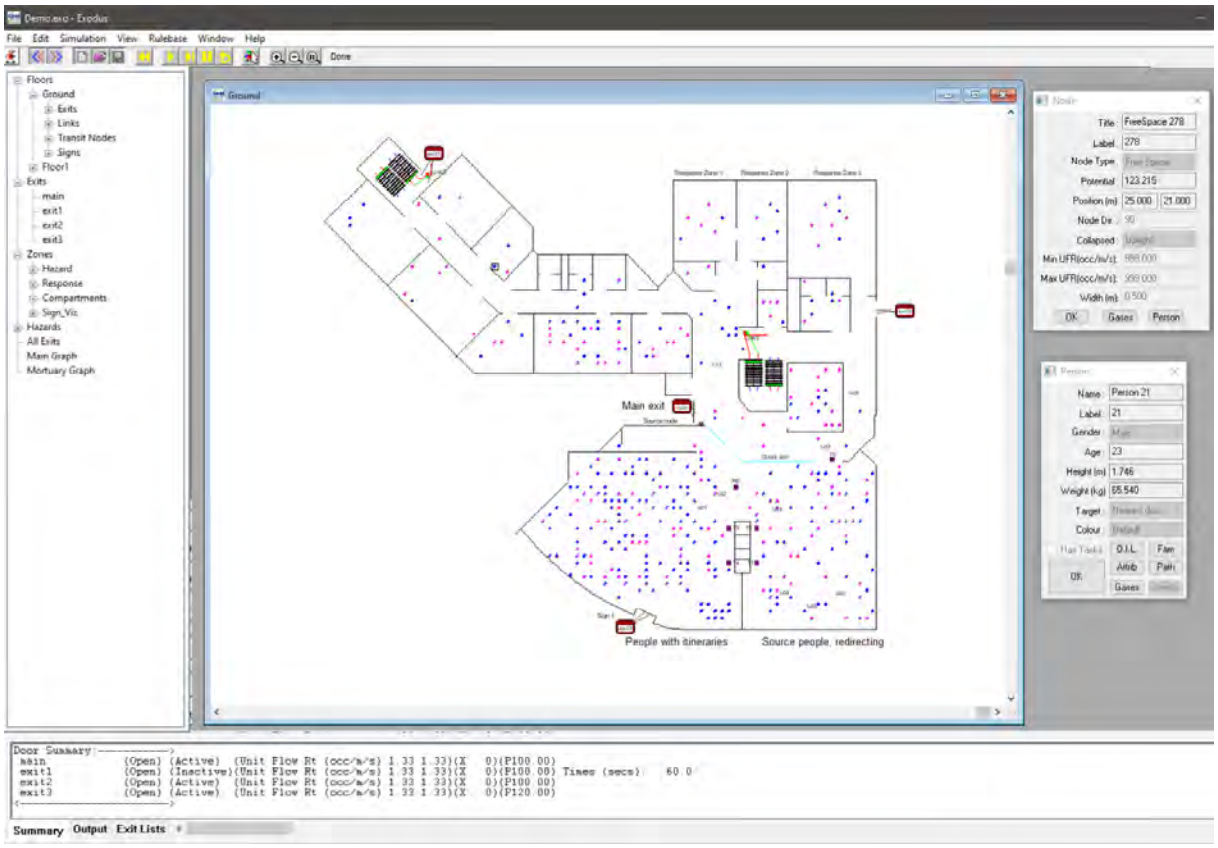


Figure 2.5: Exodus' user interface

realistic simulation but perhaps at the expense of ease of use.

OpenSteer [128], an open-source library written in C++, creates steering behaviours for agents in simulations. These agents can represent pedestrians and vehicles. This software is capable of modelling several behaviours, including wander, seek, flee, path following, obstacle avoidance, separation, alignment, cohesion, pursuit, and evasion.

Legion [104] offers several tools for pedestrian simulation, emergency evacuation analysis, and 3D visualisation. The simulation considers pedestrian interaction with each other and with the obstacles in the environment. Pedestrians change their behaviour reacting to environmental factors such as visibility, temperature, and toxicity. Agents decide their movement following the least-effort principle. This decision takes into account several factors: goal, speed, comfort, agent's preferences, environment and other agents. The walkable space is considered as a continuum rather than a discrete 2D grid.

Massive [113] is used to create large crowds with thousands of agents. Every agent interacts with the environment and other agents using 'fuzzy' logic. This logic returns real values between 0 and 1 rather than the usual two values of binary logic. Additionally, Massive offers Artificial Intelligence tools to create custom agent behaviours. Pedestrians possess multiple modules to perceive their surroundings: vision, hearing, and touch. The user can control the agent parameters such as position and behaviour. Massive can be installed as part of

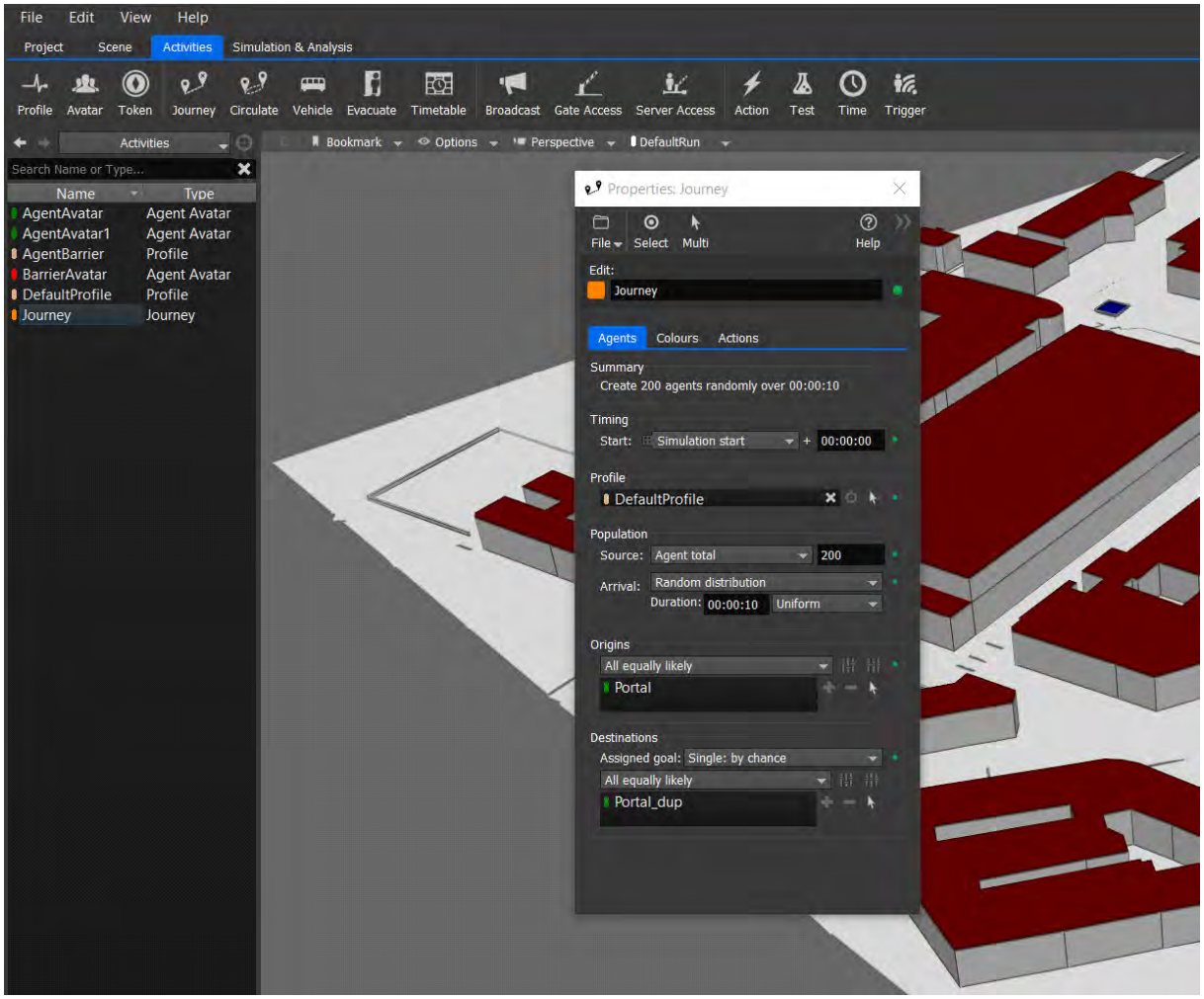


Figure 2.6: MassMotion's user interface

3D modelling software such as Maya and 3ds Max. This feature allows the creation of more complex environments.

The main issues of commercial systems are the complexity of their interfaces and the lack of real-time interaction with the simulation. Users might find it difficult to use these systems without guidance. The next section discusses graphical control solutions that have been proposed to facilitate the setup process and control of a simulation.

2.3 Graphical control

Virtual crowd simulations usually involve many parameters to configure to control the behaviour of the pedestrians. The process of tuning these parameters can be a trial-and-error task since it is difficult to predict the animation output with complete certainty [76, 92, 96, 129, 170]. Additionally, prior knowledge is required to identify which parameters need to be adjusted to produce the desired behaviour. Therefore non-expert users are not able to create or modify a

pedestrian simulation. Graphical tools for the control of crowd simulations make it possible for a user to interact with a simulation intuitively, eliminating the time-consuming task of parameter tuning.

This research identifies five categories to describe the different graphical control approaches: Navigation Graph, Map, Patch, Direct Interaction and Sketching, with a visual illustration of the different approaches given in Figure 2.7:

- Navigation Graph. An interface enables the creation or modification of navigation graphs to control the movement of the agents.
- Map. This method consists in drawing maps or layers on top of the environment to add information and indirectly modify the pedestrians' behaviour.
- Patch. Users can create large environments by combining multiple small predefined patches or blocks.
- Direct interaction with agents. A user controls the behaviour of the agents by editing parameters or specifying movement trajectories.
- Sketching. The user can sketch paths or motion trajectories directly on the environment, which virtual agents will then follow during the simulation.

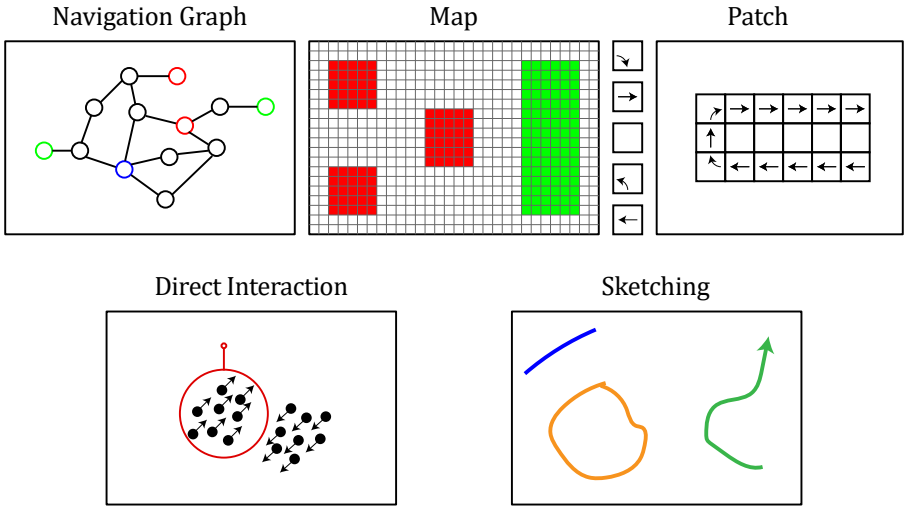


Figure 2.7: Methods to control a pedestrian simulation using a graphical interface

2.3.1 Navigation Graph

Yersin et al. [192] suggested a method, based on a navigation graph, in which users can steer the crowd towards different locations by using high-level instructions. A predefined environment is divided into navigable regions where agents can walk. These areas represent nodes of the graph. Zones outside the nodes are not accessible for the pedestrians. A graphical interface

is presented to the user showing all the nodes of the graph. Users can select the nodes and add a description or label to them. Marked nodes can be assigned to agents to change their destination. A limitation of this approach is that it is not possible to modify the environment or the graph. Furthermore, assigning new goals to agents in a large crowd or environment could be a complicated and time-consuming task.

2.3.2 Map

A map refers to a drawing or layer on top of the environment used to add information or behaviours to specific zones of the environment. A two-level crowd control system was developed by Sung et al. [169]. Situations are used to define the behaviour of the crowd at the high level. Two types of situations can be defined: spatial and non-spatial. Spatial situations are defined by drawing regions in the environment using a painting interface. Multiple layers can be overlapped to create complex situations. At the low level, an algorithm selects the action of the agents from a set of choices.

Millan and Rudomin [116] proposed a system called *Agent Paint*, where image maps are used to define attributes of the environment. These maps include: height maps to indicate the height of the terrain; texture maps to modify the scene's appearance; labelled maps to mark specific areas; and collision maps used to avoid collisions between agents. A 2D interface is provided for the creation of these maps.

Jordao et al. [76] developed a crowd simulation system where the user can determine the direction and density of the crowd. These parameters are specified by painting grayscale layers on top of the environment model. The graphical user interface was implemented as a plugin of an open-source image processing software; therefore, tools such as brushes, strokes, and gradients are available to the user. The type of layers that can be created are: density, depending on the brightness of the pixels; direction, defined by sketching a gradient; and obstacles, areas of the environment with zero density.

Pedestrian Environment Designer (PED) [115] is a tool that uses layers on top of the environment to define the behaviour of the crowd. An interface is provided to the user to add, delete, rename and hide layers. Figure 2.8 shows an overview of this system. The left image shows the graphical interface with the defined layers, and the final simulation is running on the right side. The user can define entrances, exits, collisions, areas of attraction, areas of avoidance and areas of interest. The layers are used to produce navigation force vector fields used to guide the movement of the pedestrians as shown in Figure 2.9.

In general, this graphical control method might not be intuitive since the maps are painted in a separate piece of software [116, 115] or are not created in real-time [76]. Defining a large environment by painting multiple maps could be a laborious task.

2.3.3 Patch

Patches are small blocks that can be connected to create large environments. Chenney [26] suggested an approach using small areas of force fields called *Flow Tiles*. These tiles can be connected to form larger fields to guide pedestrians through an environment. Connections

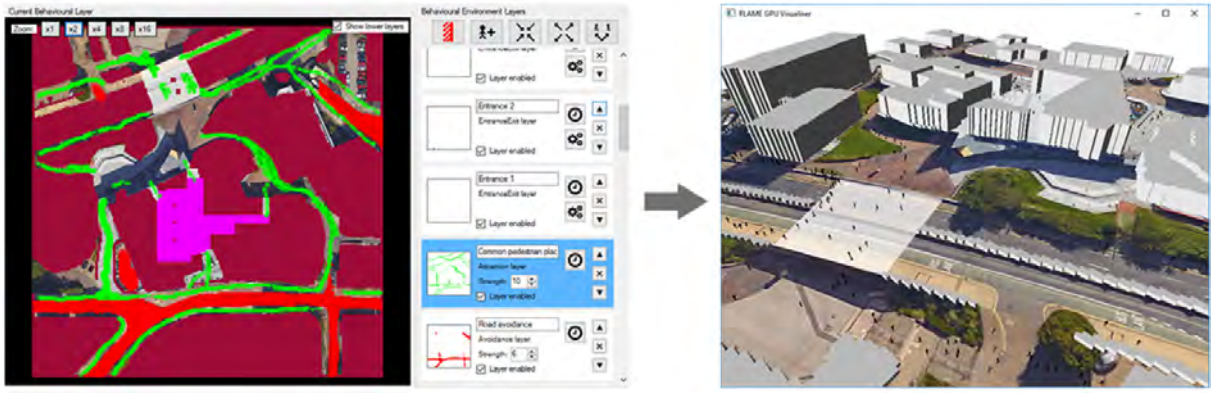


Figure 2.8: PED's interface with the created layers on the left and the simulation running on the right. Used with permission [115]

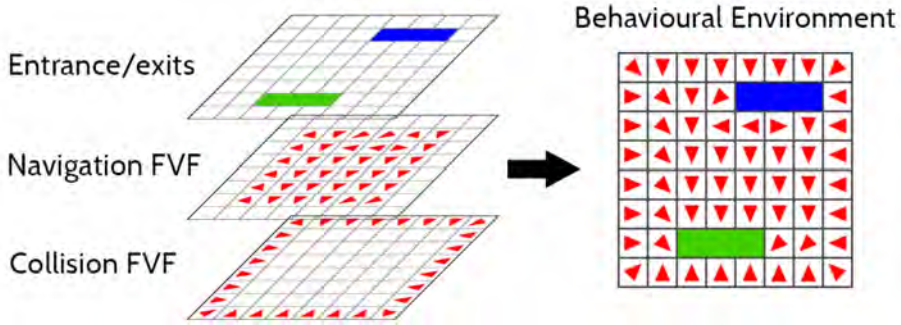


Figure 2.9: Layered force vector fields determining the agent movement. Used with permission [115]

between tiles have to meet specific requirements. For this reason, an editing tool assists the user during the tiling process.

Crowd Patches is an approach proposed by Yersin et al. [193]. Patches are blocks with pre-computed animations and may contain static and animated objects. These animations are repeated cyclically over time. Users can define the environment in two ways: progressively connecting patches in an empty scene; or dividing a predefined environment into polygonal shapes. Jordao et al. [75] extended Yersin's work [193] to create a new method called *Crowd Sculpting*. In this approach, *crowd patches* can be stretched, shrunk, bent, combined or deleted using a user interface. A virtual environment can be created from a single patch by deforming it until the desired configuration is achieved.

Lee et al. [101] presented an approach to generate an environment where animated characters can interact and navigate. This environment is built with a set of predefined blocks called motion patches. Each block has embedded motion capture data to animate the characters. Examples of the animations are: walk, climb, slide, idle, chat and sit down. The patches are constructed from the geometric shape of the environment. An interface is provided to the user to create and modify in real-time an environment with multiple objects.

Kim et al. [94] developed a simulation with interacting characters. This work uses the concept

of motion 'patches' [101]. The movement of two actors was captured while performing actions such as dancing, handshaking, and carrying objects. The patches are constructed by identifying the interaction between agents from the raw data. Two blocks can be stitched if the last and first frame have the same pose. Users can define features in the environment to indirectly control the characters. Additionally, the user can directly determine the action to be performed, the location and the time.

This control approach suffers from two main drawbacks. First, patches are used to specify the environment but not to modify it in real-time [26, 193]. Second, patches are predefined; this reduces the freedom or level of customisation when creating the environment.

2.3.4 Direct interaction with agents

In this graphical control approach, users directly modifies the behaviour of the agents by defining their trajectories or updating their parameters. The environment is not modified. The previously mentioned work by Sung et al. [169] also falls into this group. Users can create non-spatial situations such as friendship. These situations are directly assigned to the agents to modify their behaviour. Ulicny et al. [176] created a system where the user can directly modify the behaviour of the virtual characters by using artistic tools, such as brushes, in a 2D canvas that represents the 3D scene. These tools are controlled with a mouse and a keyboard. The designer is allowed to create and delete pedestrians, start animations, create paths and modify the appearance of the agents. This process consists of three steps. First, an area of the 2D world is selected by the user, and the system chooses 3D objects. Second, the designer changes the execution mode of the brush, defining how the crowd will be affected. For instance, a random operator on the creation brush will spawn a varied set of pedestrians. Lastly, the selected agents will have their parameters affected by the brush.

Kwon et al. [96] devised a technique to edit an existing animation of a group of virtual characters. The motion clips show the trajectories of all the elements of the group. A graph is constructed from these clips, and the vertices are sampled from each path. Two types of edges are created: motion and formation. Users can merge or deform graphs by pinning and dragging their vertices to change the existing animation.

Similar work was done by Kim et al. [92]; the user can edit an existing animation by enclosing characters in a cage that supports space and time manipulation. The animation is a collection of clips or lines that represents the crowd trajectories. Additionally, it includes time tracks for each character. An interface assists users to construct a cage using a freehand selection tool and selecting a group of characters. After the cage is defined, users can drag and pin down boundary vertices or interior points to manipulate it. Merge, cut, append and crop are other options for changing the animation. The same operations can be applied to the time tracks enabling warping the time of existing clips. Kim et al. [93] also presented other work where users interactively modify the movement data of several characters. Users can manipulate the position, direction, and synchronisation with spatial and temporal constraints.

Takahashi et al. [170] presented an approach to control group formations while keeping the adjacency relationships between agents. The system requires a set of defined keyframe formations to interpolate through using spectral analysis. An interface is provided to allow

the user to define every keyframe formation by directly specifying the position of every agent. Ecormier-Nocca et al. [34] suggested an authoring tool to create a keyframe animation of animal herds. The tool interface uses herd photos to create the animation. Users place the photos on top of the environment to define the keyframes. The system generates a herd from each photo and calculates a global trajectory based on their centroids. Then, collision-free local trajectories are computed to produce the final animation.

Henry et al. [60] proposed a method to control pedestrian simulations using a multi-touch device. A deformable mesh represents the crowd; the user can modify it by selecting control points and dragging them to define the final mesh. With the deformed mesh completed, a two-dimensional grid is created on top of the environment to specify a flow field exerted by the obstacles. The vertices of the mesh are examined and displaced according to the forces in the corresponding cells. Shen et al. [161] also used a multi-touch device to control a crowd with user gestures. Multiple crowd motions are associated with user-performed gestures. This information is stored in a database and used at runtime to generate the motion.

In this direct interaction approach, users might need to modify the behaviour of agents individually or in small groups, which could be a repetitive task in large-scale simulations. Additionally, new agents are not affected by any previous modification since the environment is not altered.

2.3.5 Sketching

Traditional sketching refers to freehand drawing on paper. Sketching does not require special knowledge and could express ideas without precision. Computer-based sketching refers to drawing 2D freeform strokes on an interface with a mouse or any input device. These strokes need to be converted to polylines before using the information. This process is called sampling. Section 3.1.4 describes how sampling is done in this research. Sketching in computer graphics has many applications such as sketching 3D objects [68], modelling faces [48], flooding control [147] and controlling virtual crowds. This section focus on sketching as a graphical control approach for crowd simulations.

The final category includes work that interacts with the simulation by sketching. This interaction could be by modifying the environment or by controlling the path or actions of a group. Jin et al. [73] developed a system to control pedestrians at a global and local level by drawing arrows in the environment. Vector fields determine the global movement of the agents. The graphical interface allows the user to draw anchor points with direction to determine the path of the pedestrians. This system calculates a function that represents the final vector field using a radial basis function interpolation, the position, and direction of the points specified by the user. A disadvantage of this system is that to control multiple groups requires a vector field per group. Furthermore, the anchor points only guide the movement of the crowd; it is not possible to define the exact path of the pedestrians.

In Oshita and Ogiwara's work, [129], sketching is used to define the area where pedestrians will appear. Afterwards, the designer must determine the main moving trajectory of the entire crowd. Additional lines or paths might be drawn to set other parameters, such as the distance between virtual characters. From the user-specified routes, different parameters are extracted:

guiding path, speed, and distance between agents. These parameters produce three forces which dictate the final motion of the pedestrians. This control approach has some limitations. First, it does not support real-time control of the crowd. Second, it only focuses on groups and not on individual behaviours or interactions. Last, it does not modify the environment (i.e. barriers or waypoints), it only creates paths to be followed by the crowd.

Patil et al. [132] suggested a graphical user interface where brush tools are provided to the users. These tools enable the animator to draw arrows to guide pedestrians throughout the environment. These arrows contain controlling parameters specified by the user: width of the stroke and decay ratio. A navigation field is constructed using the drawn motion trajectories as a reference. The velocity of each virtual agent is obtained by mapping their positions into the vector field. The vector field used in this method is based on a grid which has a high memory cost and does not scale well for large environments.

Hughes et al. [66] presented a sketch-based approach to populate environments initially based on an image. These environments cannot be used in automatic navigation mesh generation tools. Thus, the user first defines the boundaries of the navigation mesh and the borders of the obstacles (e.g. buildings) in an offline process using sketching. The mesh is triangulated to obtain a navigation graph. Then, users can dynamically use sketching to add waypoints, select pedestrians, create a path, and define behaviour areas where agents perform a specific action. This work is the only approach that uses sketching on top of a navmesh. However, the navmesh is not updated in real-time based on user input.

The creation of crowd formations is a popular application of sketching interfaces. The goal of formation systems is to move agents from one position to a specific location. Therefore, these systems offer limited control to create complex scenarios such as emergency evacuations. Gu and Deng [47] carried out work where users can draw or sketch lines and curves, via an interface, to define the boundaries of the formation. More complex group formations can be created by including exclusive edges to form holes inside the group. After the shape has been defined, the system assigns agents special coordinates relative to the group centre.

Gu and Deng [46] extended their previous work [47] by adding new tools to facilitate the control of group formations. The user can input the formation in three ways. First, a brush painting tool is provided to generate simple formation shapes where only the width of the brush is specified. Second, texture maps could be used for more complex formations; the shape is obtained by mapping from texture to world coordinates. Last, 'boundary sketches' is a freehand tool to define the boundaries of the formation, similar to their previous work. Additionally, the user can sketch global trajectories or lines to guide the group from one location to a final formation.

Allen et al. [4] developed a similar system to control characters at individual or group level by creating formations specified by the user. An additional feature of this work is the possibility of defining subgroups and specifying their path. To accomplish this, users must sketch multiple lines using a graphical interface. First, a group is selected by drawing a shape around the characters. A subgroup could be defined by selecting units inside the first group. Second, users specify the moving trajectory. An additional line could be drawn if a different path is desired for the subgroup. Last, the final formation is sketched at the end of the trajectory. If users define

two trajectories, both groups arrive simultaneously to the final location by adjusting their speed. Figure 2.10 shows a group selected and a trajectory to be followed (left), the group moving along the sketched line (centre) and the group arriving at its destination (right).

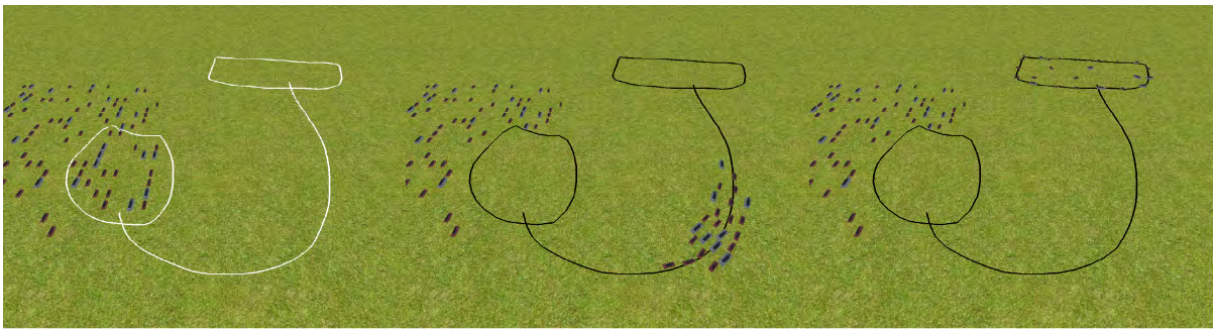


Figure 2.10: Group following a path to reach the destination. Used with permission [4]

Xu et al. [186] presented a flock simulation constrained by a user-defined shape. The flock followed the same three rules proposed by Reynolds [144]: separation, cohesion, and alignment. The user can define fixed positions over time and the path followed by the flock. Mathew et al. [114] suggested an inverse method to create crowd models. The system infers crowd motion models from trajectories. These trajectories can be from real or virtual data. The system interface allows the sketching of motion trajectories. The sketched data is used to obtain the origin and destination of the crowd, groups, paths and speeds. This information is used to create the final animation.

Xu et al. [187] developed a system to transform crowd formations. Users specify the source and target formations. The system matches the agent positions on both formations by calculating the squared Euclidean distance. Xu et al. [188] created a similar crowd formation transformation model but using different criteria to assign the final positions. Subgroups are formed to maintain the cohesion of the group. The movement of the agents is determined using the principle of least effort and an enhanced social forces model.

Zheng et al. [196] suggested a novel method to create formations with crowds. The system is based on geometry and does not require collision avoidance algorithms. The user defines the initial and final shapes of the group, and the path to follow. Source and target shapes are interpolated to obtain intermediate formations for a smooth transition. The agent distribution is calculated using centroidal Voronoi tesselation. Zhang et al. [195] carried out similar research. In this work, the user can specify the final formation by importing an image or sketching the shape. The system creates a 3D representation of the input and generates the final agent positions using a concentric circles model. Moreover, users can define the density of the formation. Hauri et al. [54] proposed a flocking algorithm to represent user-defined formations with a robot swarm. Users can create static shapes or animations using a drawing interface. The algorithm is based on Reynold's [144] work, which uses a set of rules to control the robots. A new rule is added to create the formation; robots outside the shape are moved towards the group.

Table 2.1 lists the previously mentioned works, indicates the category where they fall into and points out whether they control the agent and/or modify the environment. The majority

Category	Control	Discretisation	Research paper(s)
NavGraph	Env	Graph	[192]
Maps/Direct	A/Env	not stated	[169]
Maps	Env	Grid	[116, 115]
Maps	Env	Graph	[76]
Patches	Env	Grid	[26, 101]
Patches	Env	Graph	[193, 75]
Patches	Env	not stated	[94]
Direct	Agent	not stated	[176, 96, 92, 93, 170, 34]
Direct	Agent	Grid	[60]
Direct	A/Env	not stated	[161]
Sketching	Env	Grid	[132]
Sketching	Env	Navmesh	[66]
Sketching	Env	not stated	[73]
Sketching	Agent	not stated	[129, 47, 46, 4, 186, 114, 187, 188, 196, 195, 54]

Table 2.1: Summary of the graphical control approaches for crowd simulations. The Control column indicates whether the agent behaviour is controlled by changing agent (A) parameters and/or by modifying the environment (Env). The Discretisation column indicates how the environment is represented: Grid, Navmesh or Graph (where Graph includes techniques that use a graph structure based on circles or polygons).

of the work exploring sketching does not state what memory structure is used to represent the environment. The reason is that crowd formation has been the main focus, which only involves moving pedestrians from a starting position to a final destination. An underlying structure is not needed since agents can move freely (no elements are influencing their movement) to the specified position. There is a shortage of work exploring sketching to modify the simulation environment. This thesis explores sketching using the grid and navmesh navigation approaches. Current work on grids only includes the use of flow lines to guide the agent movement. The only navmesh work found uses sketching to define the navmesh and waypoints in an offline step. Table 2.1 highlights the lack of research done on this graphical control approach and the opportunity to explore more possibilities that sketching could offer, for instance, the creation of more control elements (barriers, areas, storyboards and real-time interaction.

2.4 Evaluation approaches

The graphical approaches described in the previous section not only require an intuitive user interface but also need to produce a plausible simulation. This section covers methods to evaluate the realism of crowd simulations, techniques that have been used to assess sketching interfaces in crowd simulations and some formal approaches to examine user interfaces in general.

2.4.1 Realism of the simulation

Three main techniques can be identified that have been used to evaluate how realistic a pedestrian simulation is. First, real-world data is used to obtain metrics, such as position, acceleration and distance, to compare them against the virtual crowd simulation. Second, the same idea of extracting metrics is used but to compare two simulations. Third, the realism is evaluated based on user perception producing more subjective results. This is the least reliable and objective technique since results depend on the perception of the users, which could be affected by many factors.

An objective approach to compare and evaluate pedestrian simulations uses real-world data. Multiple metrics have been proposed to compare virtual and real crowds. However, extracting these metrics from real crowds can be challenging [119]. Wolinski et al. [185] compared crowd simulations to real crowds by estimating the best set of parameters to match the real data. With values, the distance to the reference data can be obtained. Multiple metrics are computed: difference in agent positions, path length, inter-agent distance, progressive difference, vorticity and fundamental diagram. A different example of a metric is the fundamental diagram which compares the relationship between agent speed and density of the environment. This metric has been used to compare crowd from Germany and India in a corridor [25].

Karamouzas and Overmars [89] suggested two metrics to assess the behaviour of a group within a crowd. These measure the distortion and dispersion of the group. The metrics are used to validate the group simulation against real crowd data. Guy et al. [51] suggested an 'Entropy Metric' to evaluate crowd simulations against real crowds. This metric assesses the predictability of simulations based on their similarity to real-world data. First, a set of simulation states that better fit the real data is selected. Then, the simulation is used to predict succeeding steps. Finally, the prediction error is calculated by comparing both states. An experiment was carried out to compare this metric with perceptual evaluation. Participants were asked to rate the similarity of a set of video pairs (consisting of one real data video and one simulated video). The rates were compared to the metric obtained for each pair. The results showed a strong correlation between the Entropy metric and user perception.

Seer et al. [157] validated three social forces model using real-crowd movement data. The trajectories of the people are extracted and used to calculate parameters such as acceleration, position, and current and desired velocity. These values are used to evaluate the models based on non-linear regression parameter estimation. Wang et al. [182] proposed an approach to compare simulated and real crowd data by extracting path patterns of both data sets. These patterns involve global and local attributes of the crowd movement and provide quantitative attributes that are not obtainable from mere inspection. Furthermore, two similarity metrics are defined to compare individual and overall path pattern similarities. He et al. [55] proposed a method to decompose real-world crowd data with trajectories into a set of modes. These modes store multidimensional pattern information: space, time and speed. This data is used to propose two comparative metrics, average likelihood and distribution-pair distance. These are the same metrics suggested in [182], but now they consider three dimensions rather than just space.

Lerner et al. [106] used real crowd videos to create a database with a set of 'normal' behaviours and example state-action pairs. A state stores parameters such as speed, position

and nearby individuals at a specific point in time. An action could be a change of velocity. The simulated crowd is also analysed to obtain a set of query state-action pairs. A similarity function is used to determine the distance between the 'normal' and the simulated state-action pairs.

A further method to evaluate crowd simulations also extracts metrics from the crowd movement but compares two pedestrian simulations. Campanella et al. [19] proposed a validation score for pedestrian models. This metric is a combination of scores of quantitative and qualitative assessments. The quantitative assessments include average travel time, speed-density relation and bottleneck capacity. The quantitative score is based on the relative errors of the results. The qualitative assessments are graded by visual inspection. These focus on the avoidance behaviour in bidirectional flows, unidirectional flows and narrow corridors. Singh et al. [164] created a benchmark suite to evaluate steering behaviours. It includes test cases such as crossing, overtaking and bottlenecks, and offers metrics to compare algorithms. Metrics include the number of collisions, time and effort efficiency. Kapadia et al. [85] proposed a different set of metrics to compare steering behaviours in different scenarios: scenario completion, path length and total time. Each metric is defined as a ratio to its optimal value to allow comparison on an absolute scale. Musse et al. [119] proposed a method to compare the characteristics of two crowds (real or virtual). A 4D histogram is used to compare properties such as global flow, spatial occupancy, main orientations and speeds. This approach requires pedestrian trajectories to be extracted from the crowd.

A standard method to evaluate the realism of a simulation is perceptual experiments. The disadvantage of this approach is that it depends on the participants perception, potentially leading to non-consistent results. Paris et al. [131] used real data captured to calibrate and validate their pedestrian model. The qualitative evaluation, based on perception, focused on emergent behaviours such as lane formation. The authors concluded that similar phenomena emerged from both the model and the real data.

Peters and Ennis [137] conducted a perceptual study to assess the plausibility of the virtual crowd animation. Participants were shown a series of short animations, divided into three groups, to rate the realism of the simulation. The groups were: plausible, implausible and no groups. The results of the study confirmed that the plausible group was the most realistic for the participants. Guy et al. [50] carried out a study to evaluate the performance of a crowd modelled with personality traits. Multiple behaviours were modelled, and participants were asked to identify the video showing each behaviour from a set of videos. Participants were able to select the correct video for each behaviour with high accuracy.

Real world data is the most reliable source to validate virtual simulations. However, it might be challenging to capture situations that want to be simulated. For instance, obtaining data for emergency evacuations is risky since a dangerous scenario would need to be reproduced. This method is also not suitable for simulations that can be modified in real-time. In these simulations, users can modify the environment (add obstacles) and the virtual crowd immediately reacts to the user input. Real world data with the same changes to the environment might be difficult to find.

This research evaluates the simulation model using the second approach discussed, by

comparing the agent metrics extracted from the simulations. The comparison is made against the validated [149] commercial system MassMotion.

2.4.2 Graphical User Interface

There appears to be little work on evaluating sketching in crowd simulation. This section presents this work but also explores some formal approaches to evaluate user interfaces in general.

Sketch-based interfaces for controlling crowd simulations have been evaluated with user studies. Oshita and Ogiwara [129] evaluated the effectiveness of a user interface for controlling the path of a crowd with an experiment. Four people were given an example of animation and were asked to create it using the sketch-based system and a conventional interface. Completion time for each interface was taken and compared. The results showed that participants reproduced the animation more than ten times faster using the sketching approach. Similarly, Allen et al. [4] used participants to create scenarios, involving crowd formation and movement, in a classic control system and also using a sketch-based interface. The results indicated that the sketching method is more precise and easier to use, but takes more time to draw the shape of the formation.

Sketch-based interfaces for diverse applications have also been perceptually evaluated based on user experience. Xu et al. [189] assessed the usability of a user interface for conceptual/schematic design with symbols. The sketch-based interface was compared to a traditional button-based interface. Participants were asked to draw some sketches with both systems. Users found sketching more intuitive and faster compared to the button interface. Kara et al. [88] evaluated a sketch-based 3D modelling system conducting a study to find three aspects based on user perception: personal satisfaction, usefulness and ease of use. Participants were asked to complete a brief tutorial, design an object and complete a questionnaire. Users found the system intuitive and had a favourable opinion about the interface. However, some participants described the menus as cumbersome. This study made no comparison against traditional modelling systems.

Tsiros and Leplâtre [173] carried out a user study to evaluate the effectiveness and usability of a sketching interface to control a sound synthesiser. The study consisted of designing two soundscapes and answering a questionnaire. Overall, the participants were satisfied with the interface but also highlighted usability issues such as lack of options found in traditional image processing systems. Arora et al. [8] evaluated sketching in VR under different circumstances to analyse the factors influencing the ability to sketch strokes in mid-air. The user study was divided into two experiments. The first task compared traditional and VR sketching. Participants were asked to draw a predefined shape on a solid surface, mid-air in a VR environment and on a physical surface while using the VR headset. The difference between the target shape and the sampled sketch points was calculated. Traditional sketching showed the most accurate results. The second experiment evaluated the use of visual guidance to facilitate mid-air sketching in VR. The forms of guidance included a grid and a target stroke. Participants performed better when the grid and target stroke were used together.

A key aspect of determining the quality of user interfaces is usability. An ISO standard defines quality of use as: “the degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context

of use" [69]. Seffah et al. [158] summarises several factors that are considered when measuring the usability of a system: Efficiency, Effectiveness, Productivity, Satisfaction, Learnability, Safety, Trustfulness, Accessibility, Universality and Usefulness.

Wetzlinger et al. [183] used some of these metrics to compare the user experience when using laptops and tablets for frequently executed tasks. The tasks included writing an email, creating an appointment in a calendar app, browsing and filling forms. The user study measured Effectiveness as the task completion ratio based on the participant attempts. Efficiency was defined as the time taken to complete each task. Additionally, other perceptual metrics were considered, such as ease of use, usability and user experience. Usability was assessed by asking participants to complete the System Usability Scale (SUS) questionnaire designed by Brooke et al. [18]. The results showed that users struggled to complete some of the tasks on the tablet. All the tasks were accomplished faster using a laptop, and participants considered that the laptop was more comfortable to use. However, the questionnaire indicated higher perceived usability of the tablet. These findings suggest a discrepancy between measurable features and user perception.

The SUS questionnaire has ten questions that users answer using a 5-step Likert scale from "strongly disagree" to "strongly agree". SUS has been added to commercial evaluation toolkits and is referred to as an "industry standard" [17]. An advantage of this questionnaire is the fact that a single value representing the user perception is obtained [10]. This score ranges from 0 to 100, with lower scores indicating worse usability. However, it is still open to interpretation from which score the system is considered usable. Bangor et al. [11] conducted a study adding an eleventh question to get an overall usability user perspective of the system. The question has seven options from "Worst imaginable" to "Best imaginable". The purpose of the study is to provide an interpretation of the SUS score by matching it to user opinion. The study found that the adjective given by participants was strongly related to the SUS score. The results ranged from 12.5 (Worst imaginable) to 90.0 (Best imaginable).

A different approach to evaluate user interfaces is the Goals, Operators, Methods and Selection rules model (GOMS) proposed by Card et al. [21]. This user interface design model describes the knowledge required by the user to perform a task. Kieras [91] summarises in detail the components of the model. Goals are presented by an action-object pair to identify the tasks that users try to complete. Operators are actions to be performed by the user. Goals and operators are similar, but the difference is that the operator is executed and the goal is a task to be accomplished. Methods are a set of operators needed to complete a goal. Lastly, selection rules are used to choose a suitable method to achieve the goal. The GOMS model has generated a family of four modelling techniques based on it: Card Moran & Newell GOMS (CMN-GOMS), Keystroke-Level Model (KLM) [20], Natural GOMS Language (NGOMSL) and Cognitive-Perceptual-Motor GOMS (CPM-GOMS). John and Kieras [74] made a detailed comparison of the four techniques with an example task of editing a manuscript in a word-processor. Total time prediction showed similar values. NGOSML predicted more time since it includes more mental preparation operators. In contrast, CPM-GOMS was the fastest because it assumes extreme expertise and allows overlapping operators.

This thesis focuses on the KLM model since it is used as part of the evaluation of the system

in Chapter 5. Card et al. [20] proposed the *Keystroke-Level Model*, which is a quantitative analysis tool used to evaluate interactive systems. The goal of this model is to predict the time needed to complete a task by counting the number of low-level operations needed. This method is used in computer systems but has been adapted and extended to work with mobile phones [63] and in-vehicle systems [155, 103]. KLM assumes that expert users are performing the tasks being evaluated. Each low-level operation is represented by an operator which is given a time value to obtain the total amount of time required for the task.

The work done on assessing sketch-based interfaces mentioned in this section does not use a formal evaluation method such as KLM. The majority of the studies rely on participant perception and do not include a quantitative evaluation of the interface. This research proposes a modified KLM version to include a new sketching operator and compares the commercial system MassMotion and the sketch-based system. The KLM analysis is used to compare the number of actions used by each system, rather than predicting the task completion time.

2.5 Summary

This chapter has given an overview of virtual crowd simulation models focusing on agent-based modelling. Two levels of control were identified: local motion and global navigation. The local level refers to the short-range motion of the agents, taking into account their immediate surroundings. Global navigation is used to guide pedestrians through the environment to their destination. Four main approaches to represent a virtual world were analysed: roadmaps, navigation graphs, flow fields and navigation meshes.

Roadmaps present some disadvantages. First, they may require many points to find paths in complex environments. Second, the paths obtained could be unrealistic. Last, all the walkable areas of the environment are not represented. Navigation graphs have similar disadvantages to roadmaps. Additionally, environment coverage depends on cell shape and size, which may lead to a considerable number of cells for complex scenes. Grids scale poorly compared to navmeshes in terms of environment representation, memory usage and path-finding time.

Multiple commercial pedestrian simulation systems were analysed describing how the agents are controlled and mentioning the different options and variables that can be defined by the user.

A classification of graphical tools to control crowd simulations was proposed. This categorisation included: navigation graph, map, patch, direct interaction and sketching. The selected approach to explore in this research is sketching. Each of these graphical controlling techniques has some issues.

- The navigation graph method has the limitation of not supporting the modification of the environment or the graph.
- The maps approach might not be intuitive since the maps are painted in a separate piece of software [116, 115] or are not created in real-time [76].
- Patches suffer from two main drawbacks. First, patches are used to specify the environment

but not to modify in real-time [26, 193]. Second, patches have a limited level of customisation when defining the environment.

- In the direct interaction approach, users might need to modify the behaviour of agents individually or in small groups, which could be a repetitive task in large-scale simulations.
- The sketching method has only been used to force agents to follow a certain path [129, 132, 73] or to create formations, but not to modify the environment. The only work using sketching on top of a navmesh was proposed by Hughes et al. [66]. However, in that work, the navmesh is not updated in real-time based on user input.

Lastly, this chapter outlined the techniques used to evaluate the realism of crowd simulations and the effectiveness and usability of user interfaces. Metrics extracted from the crowd, real-world data and user perception are methods used to evaluate and compare virtual crowd models. User interfaces have been qualitatively assessed with user studies and questionnaires, such as the *System Usability Scale (SUS)*. Other methods have been suggested to carry out quantitative evaluations, for instance, the *Keystroke-Level Model (KLM)*. Sketch-based interface evaluations have not used a formal evaluation method and rely on participant perception.

Chapter 3

Sketch-based control and navigation methods

This chapter describes the sketching approach and the data structure to support it and represent the environment. Controlling a crowd simulation by sketching can be done in two ways (see Section 2.3.5): modifying the environment with the user strokes or directly changing the parameters of the agents (i.e. drawing a path). This chapter concentrates on the first approach. Users control the behaviour of the virtual crowd by sketching elements, such as barriers, flow lines and waypoints, on top of the environment. These sketches need to be mapped to a data structure representing the environment. Later chapters look into extending the sketching approach to create more advanced control features. These options include storyboards to specify the journey of the crowd and a timeline interface to simulate different times of the day.

This chapter uses two data structures, grid and navigation mesh, to develop a sketch-based control system to explore their advantages and disadvantages. Previous work [132] used a grid-based approach to sketch flow lines to direct the crowd. A contribution of this chapter is a grid-based simulation (extending the work in [132]) updated in real-time by sketching entrances, exits, obstacles and flow lines to modify the environment. Sketching with navigation meshes have also been used in crowd simulations. Hughes et al. [66] created the navmesh of the simulation environment by sketching the boundaries of the non-walkable areas. The novelty of the navmesh work in this chapter is the use of sketching to modify a navmesh in real-time by adding barriers, flow lines and areas. Another contribution is a detailed comparison between the grid and navmesh navigation approaches.

Section 3.1 gives an overview of the sketching system; some aspects vary depending on the underlying data structure representing the environment. These differences are described in later sections. Section 3.2 explains the implementation of the grid approach, the sketching process, and shows some simulation results. Section 3.3 covers the navmesh method and the modification of the tool *Recast* to support sketching. Section 3.4 compares both navigation approaches in terms of environment representation, memory usage and path calculation time. Finally, Section 3.5 summarises the sketching process and the work done on both data structures.

3.1 System Overview

The objective of the system is to create an intuitive and simple way for non-technical users to interact in real-time with crowd simulations. Figure 3.1 gives an overview of the system. In general, a domain specialist defines the agent model, their behaviour and the underlying data structure used to represent the environment (grid or navmesh). Users interact with the simulation by sketching on top of the environment to modify it in real-time. The shortest path from the agent position to the goal is computed from the new scene. Agents follow this global path and use local forces (based on the model and behaviour specification) to calculate their movement avoiding inter-agent collisions.

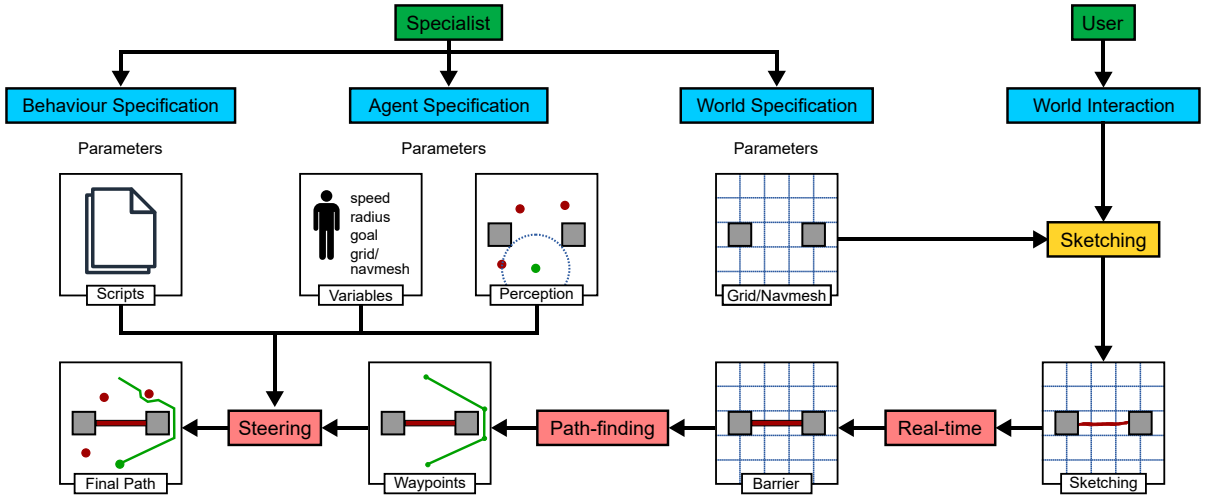


Figure 3.1: Overview of the crowd control system. A domain specialist specifies the model of the agents and the underlying navigation approach. The user interacts with the environment by sketching on top of it in real-time. The updated environment is used to find the shortest path for each pedestrian. Agents follow the global path and use local forces to avoid collisions with other agents. Based on Figure 1 in [86].

In practice, some decisions were made about particular pieces of software to create the system. Figure 3.2 gives an overview of these. There are two main modules: a visualisation module created by making use of *Unreal Engine* and a simulation module based on the *FLAMEGPU* framework [148], which handles the agent and behaviour specifications. The main modules communicate with each other through a CPU-based shared memory segment. The data required by each module is shown in Figure 3.2. The agent data used in the *FLAMEGPU* framework must be available to the visualisation module running on the CPU, which, in turn, must send sketched updates to the environment back to *FLAMEGPU* to influence the simulation running on the GPU. The system diagram slightly changes depending on the navigation method. The navmesh approach uses the tool *Recast* to generate and update the navmesh. This tool was embedded in the Unreal project. The grid-based method does not use any open-source tool to generate the grid.

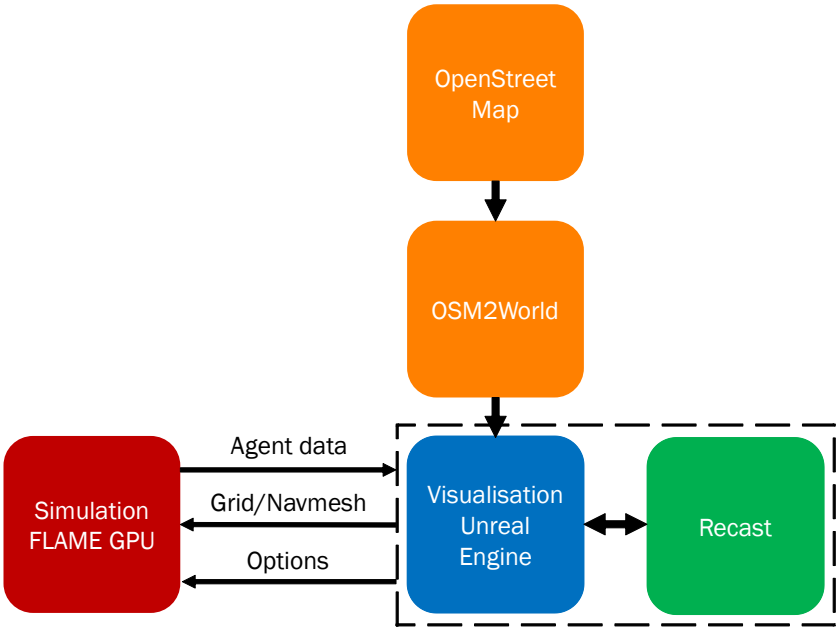


Figure 3.2: System diagram. The Recast module is only used for the navmesh approach.

3.1.1 Environment

The first step is to create an environment to run the pedestrian simulation. The environment could have been a simple scene created with blocks, such as the one shown in Section 3.2, but the idea was to create a more complex and realistic simulation. Therefore, the tool *OpenStreetMap*[1] was used to obtain the data of a real city. The area selected for the simulation is part of the city centre of Sheffield, UK. Figure 3.3 shows the area in *OpenStreetMap* together with the final 3D model of the environment. The tool *OSM2World*[2] was used to convert the *OpenStreetMap* data into a 3D model before importing it into the game engine. Some modifications were made to the model prior to the import: imperfections on the ground were removed, tree models were substituted with a new 3D model, and a few materials were replaced.

3.1.2 Character Models

The pedestrians are rendered as instances of eight base character models. The advantage of using instances is that all of them are drawn in a single draw call. Five different outfits for each model were created with varying hair and clothing colour. This gives a total of 40 models to produce a more heterogeneous crowd. The characters were created with the free online tool *Autodesk Character Generator*[3]. Unreal Engine does not allow the instance creation of models with skeleton and animation. Therefore, this approach could not be used to render the crowd. The solution to this problem was to remove the skeleton of the models and create a

[1] https://www.openstreetmap.org/
[2] http://osm2world.org/
[3] https://charactergenerator.autodesk.com/

(a) (b)

Figure 3.3: (a) Map of the selected area in *OpenStreetMap*. (b) Part of the final 3D environment in Unreal Engine.

keyframe vertex walking animation using the software *Autodesk 3ds Max*[4] (Figure 3.4). Once the animation was ready, an Unreal script for *Autodesk 3ds Max* was employed to convert the animation into a texture. This texture is applied to the Unreal material used in the static meshes of the characters to animate the pedestrians. Since the animation is embedded in the material, it could not be stopped when agents were not moving. The issue was addressed by creating an idle animation, following the same process, and blending both animations based on the agent speed.

3.1.3 Agent model

The simulation module was developed using the FLAME GPU framework, and it is based on the work done by Karmakharm et al. [90]. An adapted version of this work was created to meet the requirements of the system. The FLAME GPU simulation requires an *xml* file to define the agent variables and functions. The agent-based simulation uses the social forces model [58] to determine the movement of the agents. This is a relatively simple model but can be implemented on GPUs and it is sufficient to support the combined sketching work and grid/navmesh use. More complex agent models could be used. The agent motion is the result of the weighted sum of three forces.: (i) The pedestrian avoidance force for inter-agent collision avoidance. This is computed taking into account the position and velocity of nearby agents; (ii) The collision force used to prevent agents colliding with the environment; (iii) The goal force to guide agents to their destination. The calculation of the collision and goal forces differs depending on the use of grid or navmesh. The force weights and some constant parameters (e.g. agent radius) were tuned by trial and error and are not modified by user inputs. Sections 3.2 and 3.3 explain how these forces are calculated for each navigation method.

[4] http://www.autodesk.co.uk/products/3ds-max/overview

Figure 3.4: Frames of the character walking animation.

3.1.4 Sketching

The first step to sketch on top of the environment is to deproject the mouse screen position to the 3D world. Then, this position must be projected to the plane where the simulation takes place. To do this, Unreal Engine provides a function to transform the mouse location from 2D screen coordinates into 3D world coordinates. This function returns the position and the direction of the cursor. Then, a line trace is performed, starting from the cursor location in world coordinates. This information is used to build a line and check the intersection with the environment (Figure 3.5). The system performs this task every frame if the left mouse button is being pressed. The number of points tested depends on how fast the user sketches the line.

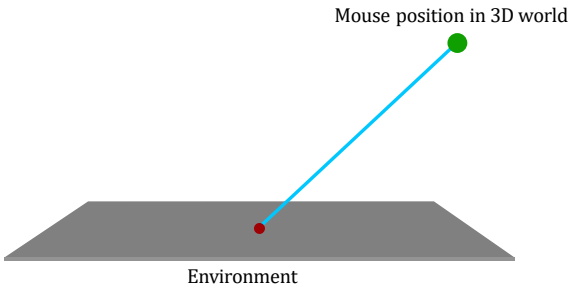
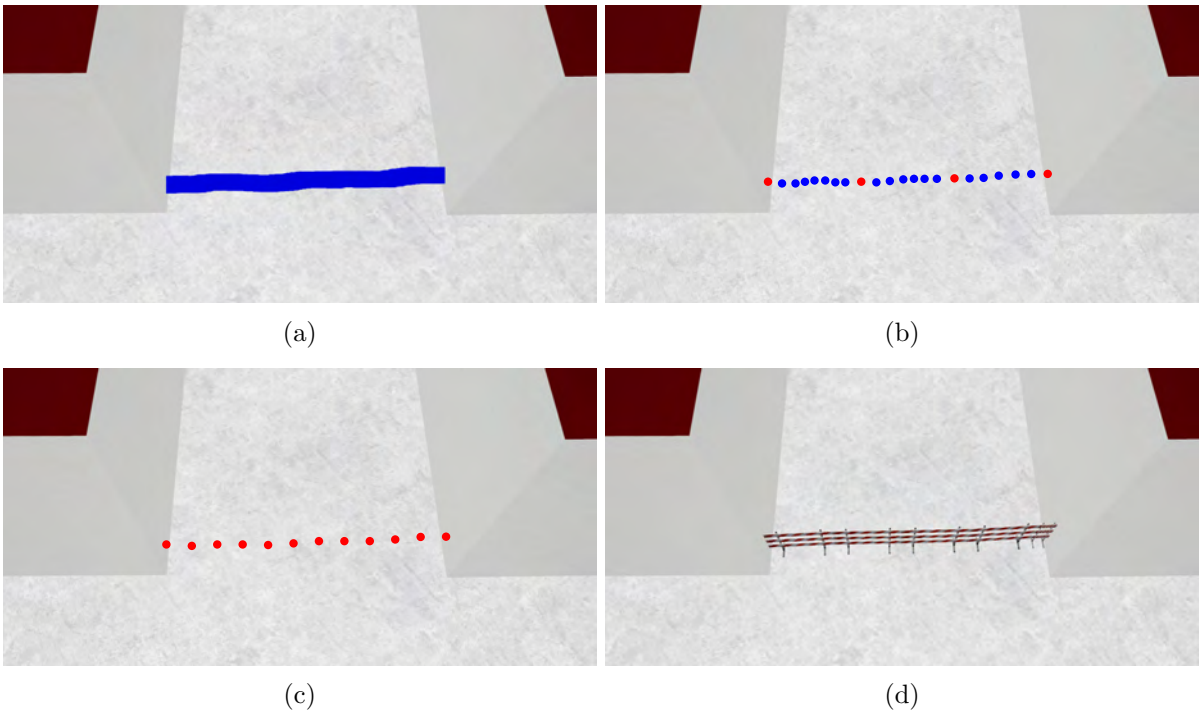


Figure 3.5: Line trace from mouse position in 3D world coordinates where collision point is indicated in red.

The game engine offers a line trace function that returns the hit object and the coordinates of the collision. If the ground plane is hit, the impact coordinates are stored in an array that

represents the line sketched. The point sequence must be sampled to reduce noise and ensure a minimum number of equidistant points. This sampling can be done during the sketch by eliminating points within a specific distance between previous samples, or after completing the stroke in several ways [126]. An example is polyline approximation. This method is used in the Teddy [68] interface; a polygon is formed from the sketch by connecting the first and last point, then the polygon is sampled to have edges of the same size. Other methods include keeping every n_{th} point and curve fitting. The method used in this research is polyline approximation.

The user stroke is sampled to create line segments of equal size. The distance between two consecutive points must not be greater than the specified size (20 units in Unreal). This gap between captured points might be large for quick strokes. Therefore, it is necessary to add twenty points between each pair of consecutive captured points (Figure 3.6). The new point sequence now can be sampled at every 20.0 units to get the starting and ending point of every line segment.



(a) (b)

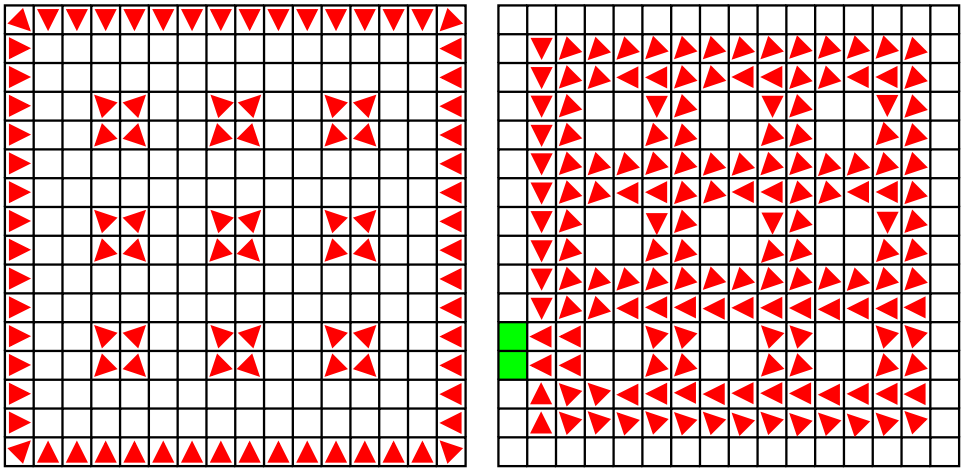
(c) (d)

Figure 3.6: (a) Line sketeched by the user. (b) Red points are captured from the user sketch and blue points are automatically added by the system. For this illustration, less than twenty points were added between each pair of samples. (c) New equidistant points obtained. (d) Final barrier created from the sketch.

This section described the components that are used in both the grid and the navmesh approach. The following sections explain in detail the work done for each navigation method.

3.2 The grid-based approach

The objective of the system is to allow real-time user control of the movement of the agents in an easy and intuitive manner. In this approach, the user can specify the pedestrian spawn and exit locations, create barriers to block pedestrian movement and force pedestrians to follow a path by sketching lines in the environment. All the strokes update the underlying grid in real-time affecting the movement of the agents. Two types of map are obtained from the grid: collision and navigation (Figure 3.7). These maps guide pedestrians to their destination.

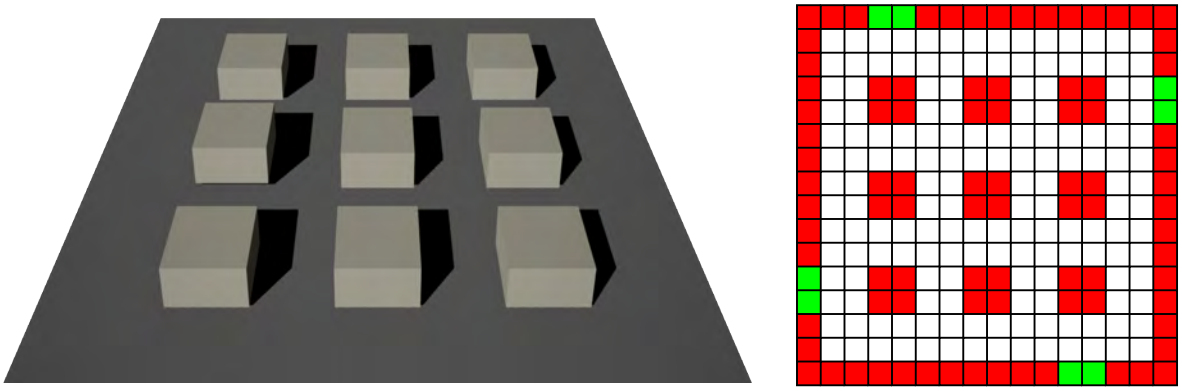


(a) (b)

Figure 3.7: Collision (a) and Navigation (b) maps obtained from the environment shown in Figure 3.8.

3.2.1 Environment Representation

The environment grid is obtained by getting the z value of the environment at the centre of each cell. A threshold height determines if the cell contains an obstacle or a walkable area. This method allows the use of 3D environment models since the grid can be quickly obtained before starting the simulation. The user can define the position of the entrances/exits by sketching lines on the environment. The stroke position is mapped to a cell of the grid and marked as an entrance or exit. Figure 3.8 shows a simple environment consisting of blocks and its grid representation. The grid shows obstacles in red, entrances/exits in green and walkable cells in white. Edges of the environment are considered obstacles to avoid pedestrians walking out of bounds. Figure 3.9 shows a top view of the final environment with a low-resolution grid to illustrate the idea of the grid. The resolution of the grid used for the simulation is much higher to represent the environment accurately.



(a) (b)

Figure 3.8: (a) Toy environment formed by 9 blocks. (b) Grid representing the environment. Obstacles, exits and paths are represented by red, green and white cells respectively.

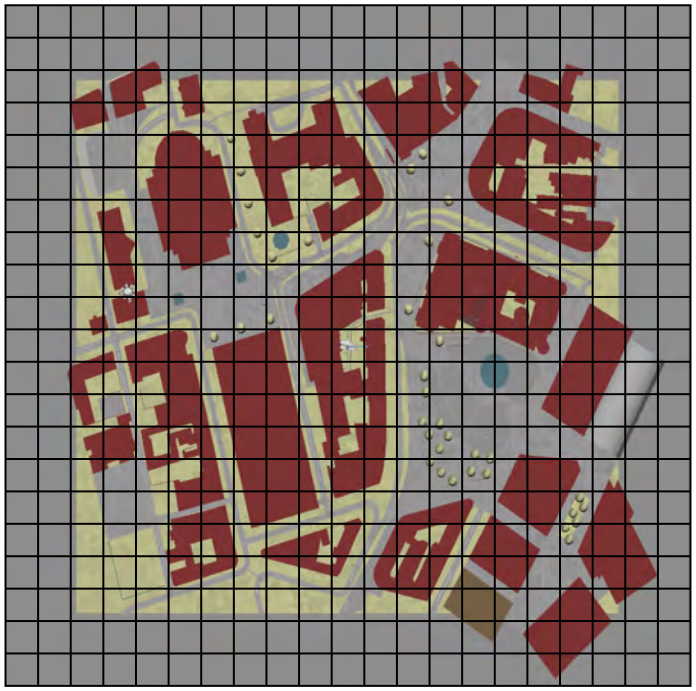


Figure 3.9: Top view of the environment divided into a low resolution grid for illustration purposes.

Collision map

This map is represented by a 2D array where each cell contains x and y components of the repulsive force of the obstacles to push away pedestrians. There are two types of objects for this map: (i) static, which will never change throughout the simulation, and (ii) dynamic, which could be added, cut or deleted by the user at any time. The collision map is initialised with the static objects of the environment. Empty cells are assigned with '0' whereas obstacles are marked with '-1'. Dynamic obstacles update the collision grid while the simulation is running. The last step is to compute the x and y force components of every cell with a '-1'. Since the desired force is repulsive, its direction must be pointing to cells with no obstacles. For each marked cell, its eight neighbours are analysed, and the normalised sum of forces of the empty neighbours is used as the resultant force component. Normalising the result means that all the occupied cells generate the same force strength. This could be improved by adding weights to differentiate types of obstacles. For example, the repulsive force exerted by buildings should be stronger to avoid crowd walking through them (See section 3.2.5). Figure 3.10 shows an example of an obstacle cell (the blue cell). Adjacent walkable cells are assigned with a normalised force based on their position relative to the processed cell. For instance, the top neighbour receives a force of (0,1). In this case, the resulting force is the same as the top-left neighbour after adding all the forces and getting its unit vector.

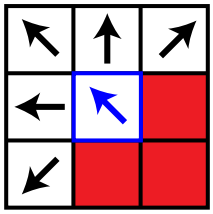


Figure 3.10: Resulting repulsive force (in blue) of the cell in the centre. Black arrows represent the direction of empty neighbouring cells with respect of the processed cell, red cells represent obstacles.

Figure 3.11 shows the steps to produce the final collision map for the environment shown in Figure 3.8. First, the environment grid is computed. Second, cells are assigned with a value depending on their state (obstacle or empty). Last, a repulsive force is calculated for each obstacle cell based on its neighbours.

Navigation map

Navigation maps are used to guide the pedestrians from every position in the environment towards their assigned exit following the shortest path. One map is generated per exit. Every map is represented by a 2D array containing a force pointing towards the specified exit in every walkable area cell. The array is initialised with '0', and obstacle cells are filled with '-1' following the same procedure as the collision map. The cells corresponding to the exits are set to a '-2' value. A wavefront propagation algorithm is used to fill the remaining reachable cells. This algorithm works as follows:

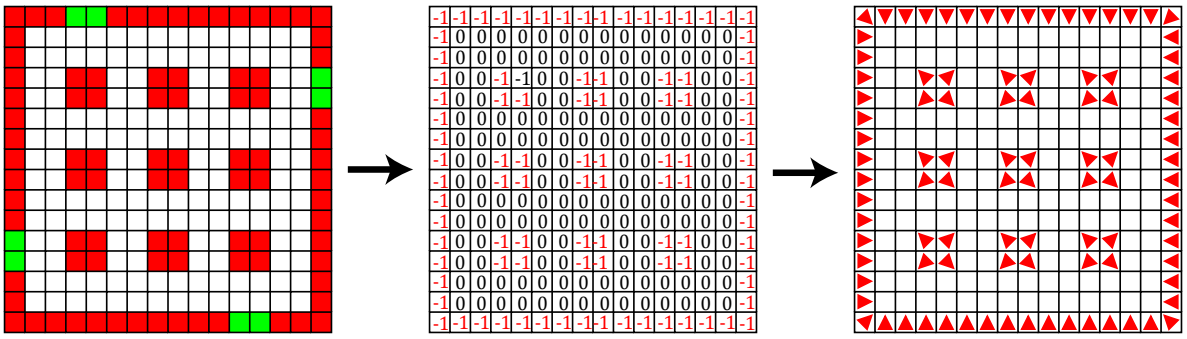


Figure 3.11: (Left) Initial grid obtained from the environment. (Centre) Two-dimensional array where '-1' indicates obstacles and '0' walkable areas. (Right) Final collision map after computing the force of each cell.

- Adjacent cells to an exit are set to '1'.
- The immediate neighbouring cells (only the four cardinal directions) that are not exits or obstacle are set to $currentvalue + 1$.
- The second step is repeated for every visited cell until the array is filled.
- Walkable cells surrounded by obstacles are not modified by the algorithm; therefore, no force is assigned.

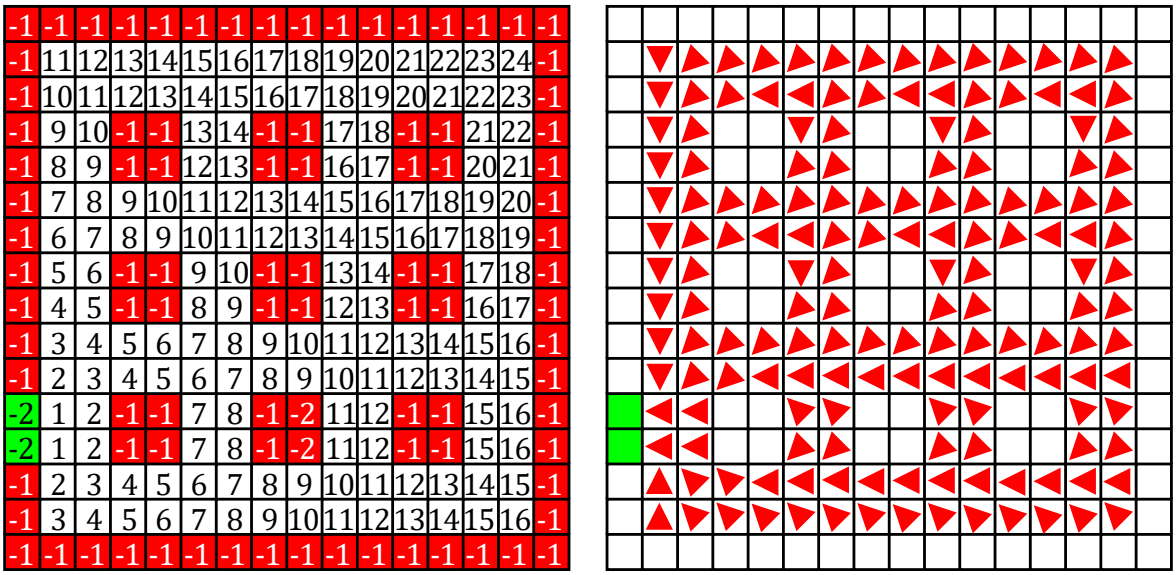
The final step is to compute the x and y force components of every walkable cell. For each element, the values of the neighbouring cells are compared. The final force is in the direction of the lowest value cell. An example of a wavefront propagation algorithm and a navigation map are shown in Figure 3.12.

The shortest path, created by the navigation maps, produces non-realistic behaviour. This issue is more evident in open areas. In Figure 3.14(a), it can be seen that all the cells are pointing to the middle line, which is the shortest path. This navigation grid produces unnatural pedestrian flows since agents will walk in a single line. This problem is addressed by applying a smoothing algorithm to the navigation maps in order to create more realistic paths.

The main idea of the algorithm (Algorithm 1) is to follow the shortest path from each cell to the goal, pushing the visited cells into a stack. The sum of forces is calculated in reverse order from the last element to the initial cell of the stack. The algorithm is terminated early if any cell neighbour is an obstacle. Figure 3.13 illustrates the steps of the algorithm. The resulting grid (Figure 3.14(b)) forces pedestrians to spread along the path rather than following the same line.

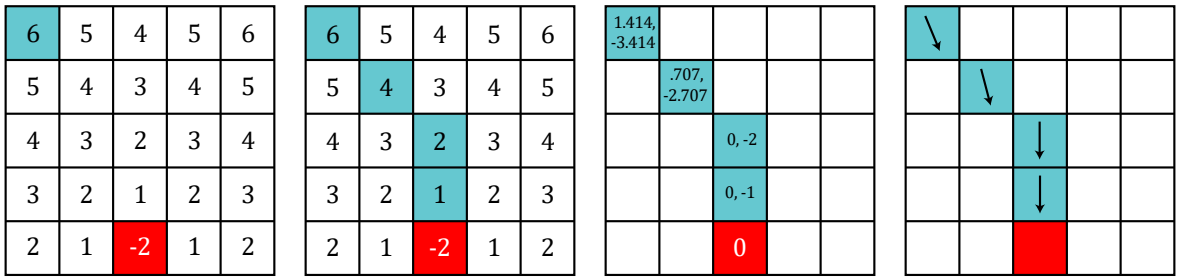
3.2.2 Agent motion

As mentioned earlier, the movement of the agents is driven by three forces: navigation, collision and pedestrian avoidance. The first two forces are based on the collision and navigation maps, and the location of the agents. Figure 3.15 shows the set of maps created for the toy environment



(a) (b)

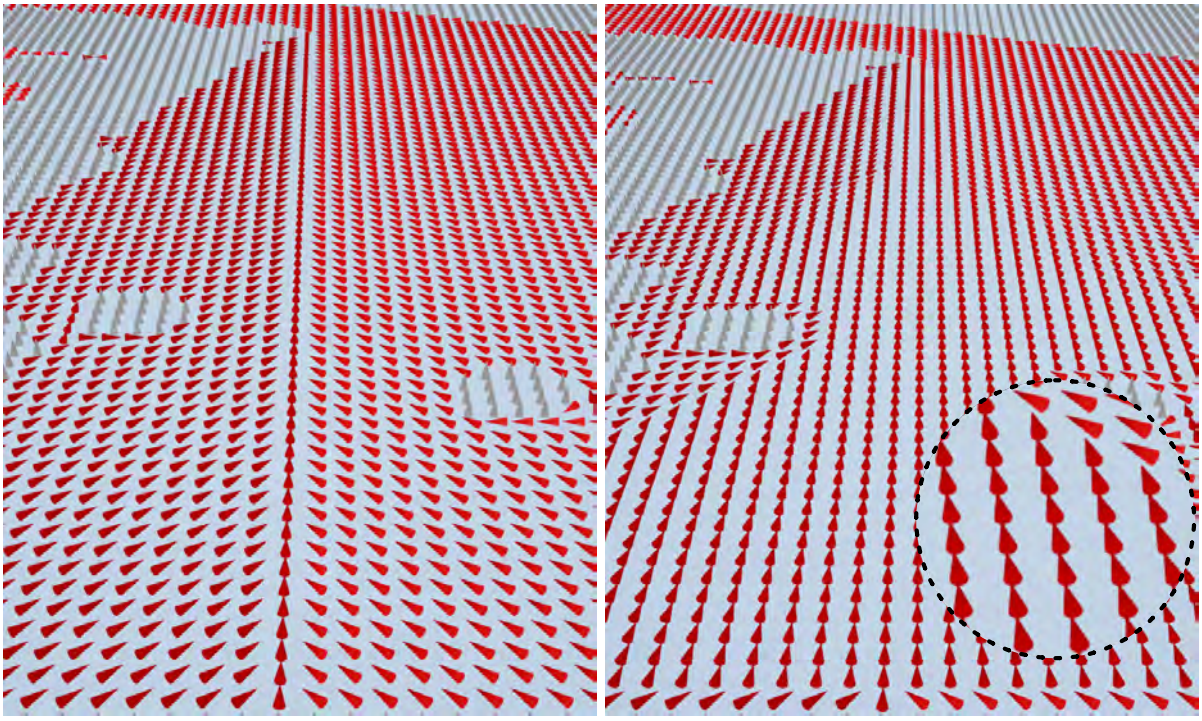
Figure 3.12: (a) 2D array after applying wavefront propagation algorithm. Red, green and white cells represent obstacles, exits and walkable areas respectively. The number in each cell is the distance to the exit. (b) Final navigation map, where red arrows represent the force directions.



(a) (b) (c) (d)

Figure 3.13: (a) Initial grid after applying the wavefront algorithm. The numbers indicate the distance to the goal; the blue colour indicates the starting point and red colour the exit. (b) The shortest path is indicated in blue. (c) The sum of the forces is calculated in reverse order for each blue cell, starting from the goal. The forces are divided into x and y components. (d) The resulting force is computed.

with four exits shown in Figure 3.8. The agent positions are mapped to corresponding cells of the two maps to get the collision and navigation forces. The pedestrian avoidance force is computed using the position and velocity of each agent within a certain radius. The resulting force that determines the movement of the agents is calculated as a weighted sum of the three forces. Pedestrians are spawned at every entrance/exit location and are assigned a random exit.



(a) (b)

Figure 3.14: (a) Navigation map before applying the smoothing algorithm. (b) Same grid after applying the algorithm. The dotted circle shows a zoomed in area to highlight the direction of the vector field.

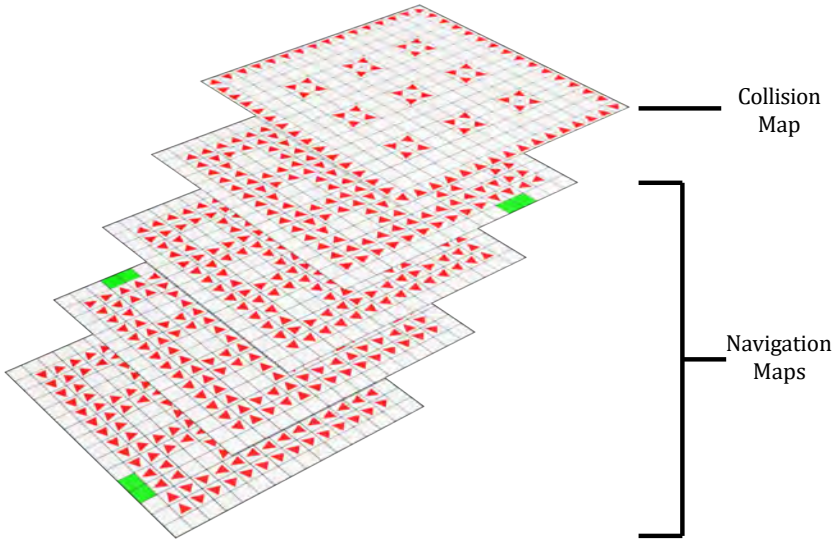


Figure 3.15: Grids generated for a simple, illustrative environment with four exits. There is one common collision map and one navigation map per exit. Spawn and exit points are marked in green.

Algorithm 1 Smoothing algorithm

```
for all cells do
    ForceDirection[cell] ← 0
end for
for all walkable cells do
    enqueue(cell)
    push(cell)
    while queue not empty do
        currentCell ← dequeue()
        minimum ← getLowestNeighbour(currentCell)
        if minimum ≠ exit and minimum ≠ obstacle then
            enqueue(minimum)
            push(minimum)
        end if
    end while
    while stack not empty do
        currentCell ← pop()
        minimum ← getLowestNeighbour(currentCell)
        if ForceDirection[minimum] ≠ 0 then
            ForceDirection[currentCell] += ForceDirection[minimum]
        end if
        direction ← getDirection(minimum)
        ForceDirection[currentCell] += direction
    end while
    FinalForce[cell] ← getForce(ForceDirection[cell])
end for
```

3.2.3 Sketching

The system allows users to specify entrance/exit locations, create barriers to block pedestrian movement and force pedestrians to follow a path by sketching lines in the environment. After sketching, the information for newly created barriers and flow lines is communicated to the simulation module through the shared memory segment residing on the CPU (Figure 3.1). From *Unreal*, sketched data is copied to the shared memory segment. The simulation module reads this data and uses it to recompute the maps, which are then copied to the GPU to process the next iteration of the simulation. After the iteration, the simulation module copies the pedestrian positions back to the shared memory for *Unreal* to use in visualisation. Figure 3.16 shows the menu interface provided to the user. The following list briefly describes the functionality of every option:

- Barrier: Sketch a line on the floor plane to build a barrier from it.
- Arrow: Sketch a line on the floor plane to spawn a flow line in the environment.
- Delete: Select a barrier and delete it.
- Cut: Cut part of existing barriers.
- Undo: Undo the last action.
- Exit Probability: Modify the probability of a pedestrian walking towards a specific exit.
- Camera Speed: Adjust the camera movement speed.
- Save: Save the current environment and barriers to a file.
- Quit: Exit the simulation.

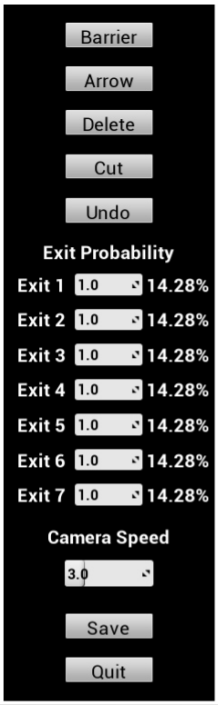


Figure 3.16: Main grid menu.

Barriers

The user can create, cut and delete barriers using the mouse. The *Barrier* button of the menu enables line sketching by holding down the left mouse button and dragging the mouse. Once the sketch is completed, releasing the mouse button will spawn a barrier replacing the drawn line. The user input is mapped to the grid. The affected cells are marked as obstacles ('-1'), and all the maps are recalculated. This process is shown in Figure 3.17. The first image shows a line sketched by the user, followed by the barrier created after completing the stroke. Figure 3.17c shows the updated collision map. Figure 3.17d and 3.17e show pedestrians walking through a gap in the barrier and the corresponding collision map.

Flow Lines

The flow of pedestrians can also be controlled by drawing flow lines. The *Arrow* button enables this functionality. The sketching and sampling process is identical to the barrier creation. An arrow is drawn in the environment, pointing towards the direction of the user sketch (see Figure 3.18b). This flow line is mapped to all the navigation maps, replacing the previous force values of the involved cells. An alternative would be to blend the new values with the existing values. However, replacement avoids the problem where overlapping opposing arrows could cancel out their respective forces, resulting in a null zone of no movement. The width of influence of the arrows is set to three cells and is not currently user-configurable.

A potential problem may occur when a flow line force and a neighbouring cell force of the navigation map are completely or nearly opposite. A pedestrian walking in that area could

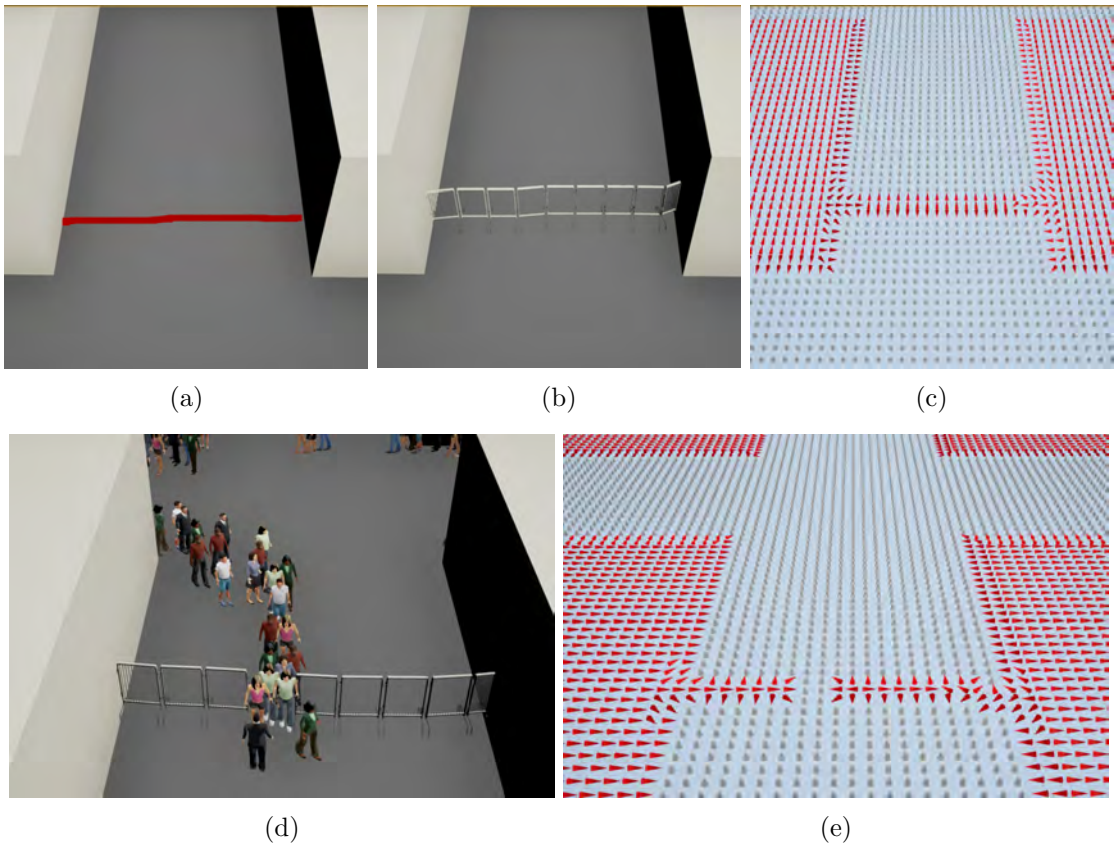
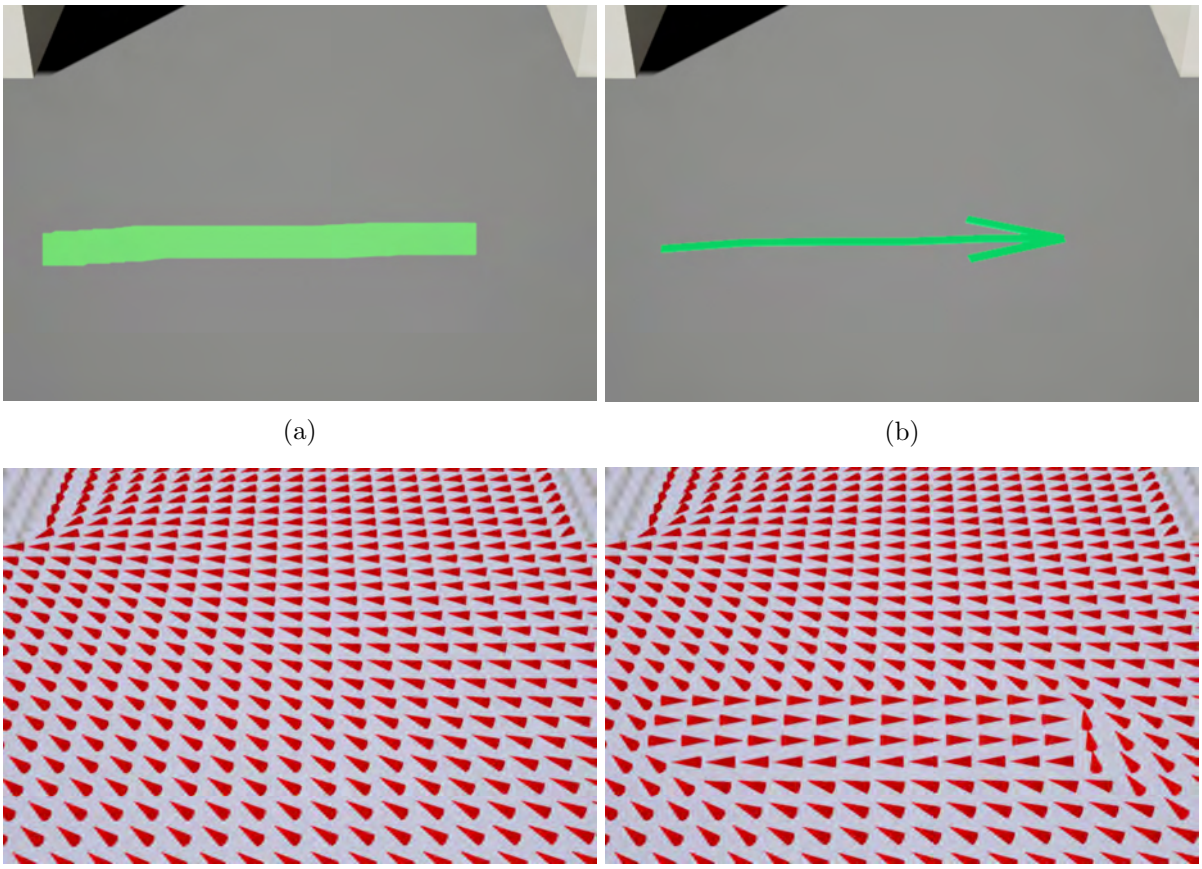


Figure 3.17: (a) Line sketched by the user. (b) Barrier spawned in the same place after the stroke. (c) Collision map updated with the new obstacle. (d) Crowd walking through a barrier gap. (e) Collision map reflecting the cut barrier.

become 'trapped' by those two forces, circulating in a small area. The issue is addressed by considering the opposite flow line cells as obstacles. The navigation map is recalculated, avoiding the opposing arrow (see Figure 3.18d). As a result, pedestrians avoid walking into the cells with opposite direction. In densely crowded environments, agents might be pushed into these undesired flow lines. In this case, agents will follow the arrow and then retake their original path, walking around the flow line, to their destination. This is a simple solution for a complicated problem and can cause sharp turns. Patil *et al* [132] solved this problem by assigning a cost to each cell. This cost depends on the direction in which the cell is traversed respecting the sketched flow line.

3.2.4 Results

The simulation was tested on a PC with the following specifications: Intel Core i5 6500 S Quad Core 3.2GHz 3.6GHz Turbo 6MB Cache, 2x Corsair 8GB Module DDR4 3000Mhz, and NVIDIA GeForce GTX 1060 SC Gaming 6GB GDDR5 1280 Core VR Ready. The performance of crowds with different sizes is listed in Table 3.1. Figure 3.19 shows a simulation running of a 50,000 pedestrian crowd at 15 frames per second.



(a) (b)

(c) (d)

Figure 3.18: (a) Line sketched by the user and (b) arrow created when finishing the sketch. Navigation map (c) before and (d) after the flow line.

The system provides an intuitive graphical way to interact with the simulation by modifying the environment and influencing the pedestrian movement. This is illustrated using a range of scenarios: path control using barriers (Figure 3.20), controlled lane formation (Figure 3.21) and the use of turnstiles to control movement (Figure 3.22). A video of the system is available at https://tinyurl.com/y6du8a8t. Figure 3.20 shows pedestrians adjusting their path to avoid the barriers created by the user. A barrier is used to block off access to one corridor, and multiple barriers are used to produce a snake of movement for a group of pedestrians.

Figure 3.21 shows lane formation control. Such motion can be observed in real crowds, for example, when groups of pedestrians walk in opposite directions at road crossing points. This behaviour can be simulated by sketching opposite arrows next to each other. In Figure 3.21, multiple lanes are created in the same corridor, and pedestrians avoid collisions with pedestrians walking in a different direction. While lane formation can emerge in agent-based simulations, the sketch-based system offers easy control over where it occurs.

Figure 3.22 shows the use of barriers, cuts and arrows to create turnstile-like behaviour, as might be seen at the entrance/exit of a train station. A barrier is created to block the path, and two holes are cut to allow pedestrian flow. Pedestrians trying to walk through a narrow

No. of Agents	Frames per second
1,000	90-91
5,000	82-84
10,000	49-50
20,000	26-27
50,000	14-16

Table 3.1: System performance for multiple crowd sizes.

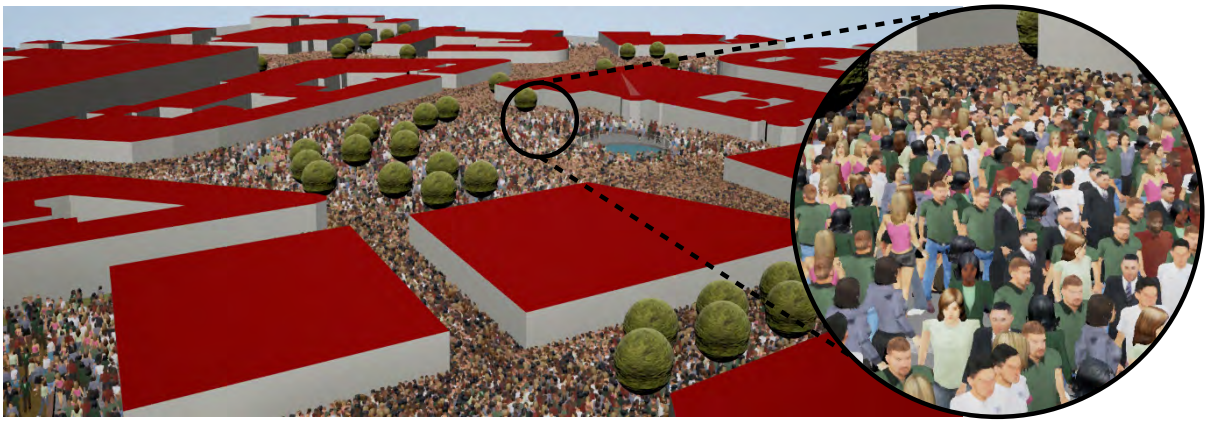


Figure 3.19: Simulation with 50,000 pedestrians running at 15 frames per second.

Figure 3.20: Pedestrians following the path created by barriers.

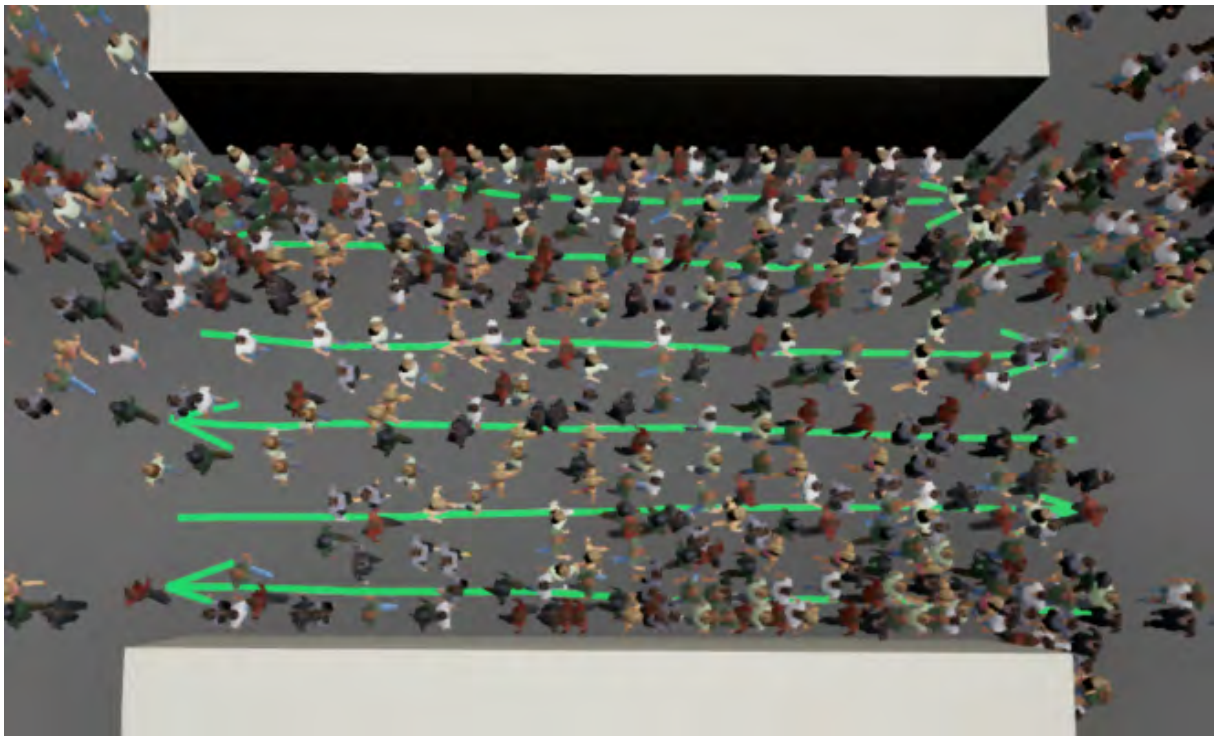


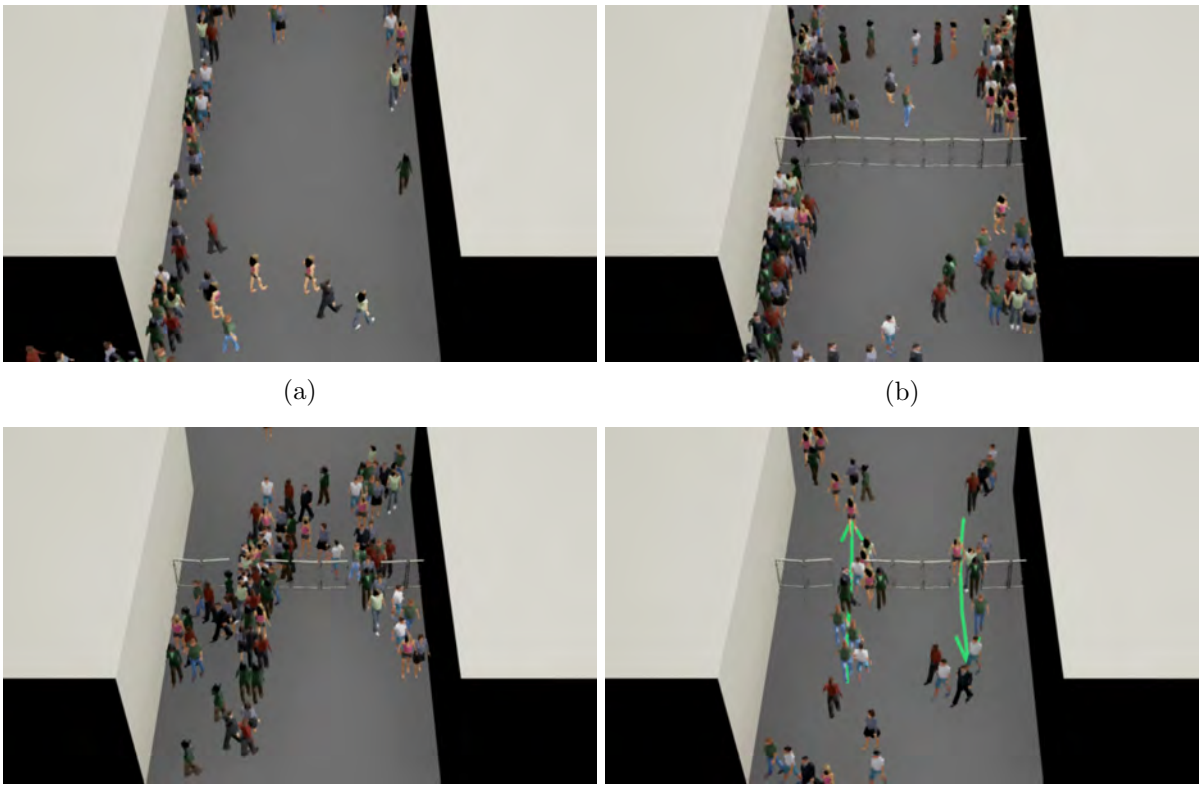
Figure 3.21: Lane formation behaviour simulated with multiple flow lines.

space in opposing directions cause congestion for each turnstile, as illustrated in Figure 3.22c. To address this issue opposite arrows are drawn, one in each gap, to force pedestrians to move in the specified direction. A similar scenario is shown in Patil *et al*'s work [132]. However, a predefined environment is needed in their work, whereas in this system the entire scenario can be recreated by real-time sketching at any position in any environment.

This sketching approach can be compared to other research work. Jin *et al*'s work [73] uses flow fields to control multiple crowds, which can be updated in real-time by the user adding or deleting 'anchor points' with associated direction. However, as the number of points increases, the generation of the vector fields by radial basis functions based vector interpolation becomes more expensive, having an impact on the simulation performance. In this research, the number of arrows does not affect the performance since only the existing grid forces are altered. Oshita and Oqiwara's work [129] uses 'guiding paths' for pedestrians, but does not allow the user to update these in real-time. Also, neither of these approaches has the feature of adding obstacles to modify the environment.

3.2.5 Limitations

The sketching approach is an intuitive way to control certain aspects of the simulation. However, the grid-based navigation approach also has some issues and disadvantages. First, a serious drawback is evident in dense crowds where the sum of repulsive forces between agents may result in a value greater than the force exerted by obstacles in the environment. This problem leads to pedestrians walking through buildings, as shown in Figure 3.23. Second, an issue with



(a) (b)

(c) (d)

Figure 3.22: Simulation of turnstile behaviour. (a) Pedestrians moving towards their assigned exit. (b) Agents changing direction after a barrier was created. (c) Congestion created due to opposing agent flow in small gaps. (d) Free pedestrian flow following the direction specified by the flow lines.

the navigation map is homogeneity since all the agent walking towards the same exit will follow the same path. Third is poor scalability – the resolution of the grid has to be increased for larger environments for a more accurate representation. High resolutions lead to increased memory usage and more computation time to generate the navigation maps. Fourth, modifying the environment updates the underlying grid that controls all the pedestrians. Therefore, drawing guiding paths for individual agents or small groups of agents is not possible. Last, pedestrians trapped between obstacles do not move since they are not able to find a way to their destination (Figure 3.24). The last two limitations are not specific to the grid but to the sketch approach. Sketching on top of the environment affects the entire crowd. Creating a flow line for a specific group of agents would require an extra copy of the environment specific to the group. The issue of trapping pedestrians between obstacles exists since users can freely sketch elements. The system should check if new elements cause any problems to the existing crowd before updating the environment. This would involve ensuring, after each sketch, that there is a clear path from every polygon to all the exits. Chapter 4 explores further uses of sketching to control only a group of agents.

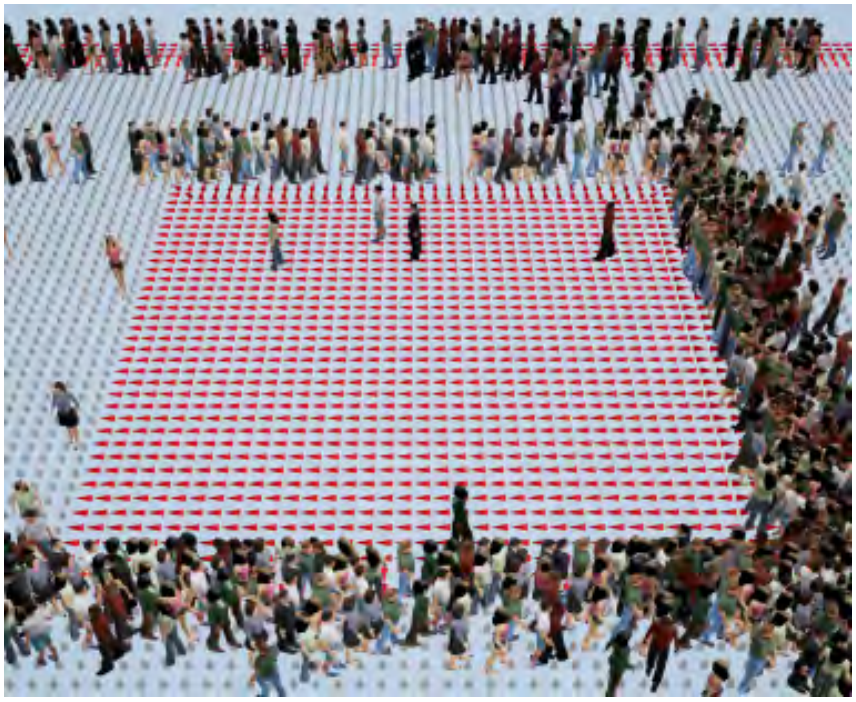


Figure 3.23: Pedestrians walking through buildings.

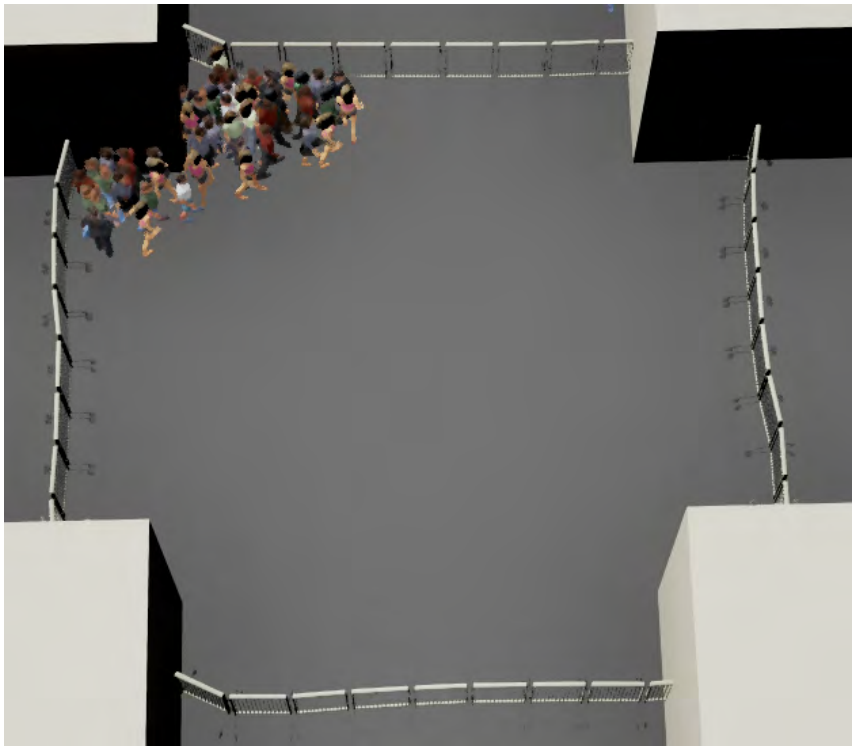


Figure 3.24: Pedestrians trapped between barriers.

3.3 Navmesh

A navmesh is a mesh of polygons that represent the walkable areas of an environment (see Section 2.1.3 for details). The polygons must be convex so agents can move freely within the polygon. An advantage of navmeshes over grids is that polygons can cover larger areas than cells. This might reduce the memory used and the pathfinding time.

Hughes et al. [66] presented an approach where a navmesh is generated by sketching the boundaries of non-accessible areas. Additionally, users can sketch waypoints, paths and behaviours. However, the navmesh is not updated in real-time. This research presents three novel contributions. First, sketching is used to update a navmesh in real-time. This includes the ability to draw barriers, unlike previous work where a list of points was used to add an obstacle to a navmesh [78], which is less intuitive for the user. Second, flow lines can be sketched, and the cost of traversing each flow line can be individually changed. Third, areas can be sketched onto an environment, similar to [66], but with explicit control being given over the percentage of agents visiting each (waypoint) area. This section covers the implementation of these three contributions. Chapter 4 build on the use of this data structure to add more complex features and identify problems of modifying simulation environments in real-time.

3.3.1 Environment Representation

The underlying navmesh used to represent the environment and determine the movement of the agents is created with *Recast*, which is an open-source tool used in games to automatically create a navmesh from a 3D environment. The tool includes *Detour*, which is a toolkit for navmesh path planning using the $A*$ algorithm. *Recast* divides the environment into tiles (i.e. a grid) and then creates the navmesh for each tile individually to form the polygons representing walkable areas. The polygons of adjacent tiles are connected to allow movement between tiles. This tiled approach permits the real-time update of individual tiles rather than the entire navmesh. Figure 3.25 shows a top view of the environment on the left, and the navmesh created for the highlighted area on the right. Later updates are only made in affected tiles. This feature facilitates the real-time modification of the mesh. *Recast* follows several steps to create a navmesh from input geometry:

- Voxelisation: The first step is to discard triangles based on the maximum slope defined by the user. The remaining triangles are rasterised into a grid.
- Filtering: Voxels that are not walkable are discarded. This is determined by agent parameters such as height and radius.
- Partition: Walkable surfaces are partitioned into simpler regions for easier triangulation.
- Identify contours of the regions.
- Build polygons to create the mesh representing the walkable surfaces.
- Create a detailed mesh to obtain the polygon heights.

(a) (b)

Figure 3.25: (a) The tiled navmesh created with Recast and displayed in Unreal Engine. The underlying square tile pattern is shown, as well as the polygons created to connect different parts of the environment such as buildings and trees. (b) The environment where the simulation runs. The red rectangle highlights the area shown in (a).

The *Recast* software was modified to support sketching and to update the navmesh according to user actions. After every navmesh change, the shortest path from every polygon to the target is recalculated. The navmesh is represented by a structure storing the polygon data (see code snippet below). The structure includes the number of vertices, edges, neighbours and the route of the shortest path to every exit. This information is sent to the simulation module through the shared memory segment described in Section 3.1 and Figure 3.2.

```
1  struct poly
2  {
3    int count;
4    int *vertexCount;
5    int *first_vertex_index;
6    int **edges;
7    int **neighbours;
8    int **routes;
9  };
10
11 struct vertex
12 {
13   int count;
14   glm::vec3 *vertices;
15 };
```

3.3.2 Pathfinding

A navmesh can be represented by a graph. This graph is formed by nodes (polygons) and the edges shared by adjacent polygons. Each node contains a list of neighbouring polygons and the

connecting edges. The navmesh includes all the walkable areas of the environment. Some of these regions might not be reachable; thus, the polygons representing these disconnected zones are discarded to accelerate the pathfinding process and reduce the graph size. The resulting graph is used to calculate the shortest path from every polygon to all the exits and areas. The $A*$ algorithm is used to compute the shortest path - this uses a heuristic value to guide the search for better performance. A route is computed for each exit and area. To create the route, every polygon stores the adjacent polygon leading to the corresponding target. In this manner, agent movement can be calculated knowing the current polygon and the assigned exit/area of the agents. A grid search is used to find the polygon faster. Every polygon is stored in a cell of the grid based on the vertices of its bounding box. The agent location is then used to obtain the grid cell and to test containment in every polygon of that cell.

One approach to generate a path is by following the middle point of the edges connecting the polygons of the shortest route. However, this would produce unrealistic paths where pedestrians would move along a zigzag course. This problem can be solved by smoothing the resulting path. The algorithm employed to achieve this is called *The Simple Stupid Funnel Algorithm* [31]. This technique finds the corners of the path staying inside the polygons found by the A* route. Figure 3.26 shows the navmesh of an environment area with three exits in red. The lines starting from the middle of the area indicate the shortest path found from that point to every exit.

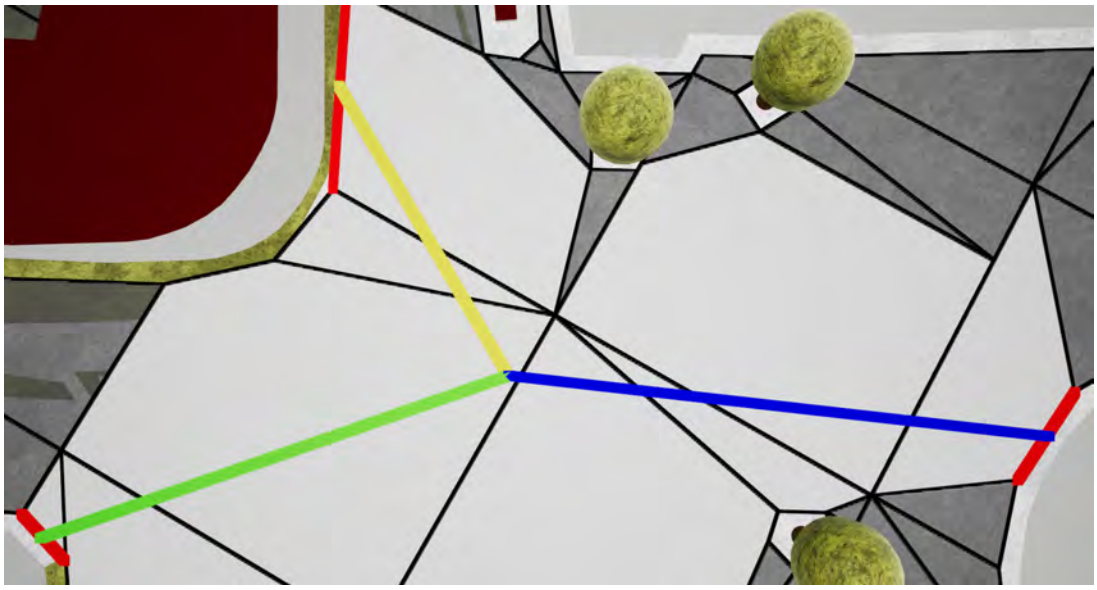


Figure 3.26: Part of the environment with three exits in red. The shortest path from the centre to each exit is represented by the coloured lines. The polygons crossed by the paths are highlighted in white.

3.3.3 Agent motion

The simulation again uses the social forces model to control the agent movement. The forces are: (i) The pedestrian avoidance force for inter-agent collision avoidance. (ii) The collision

force used to prevent agents from colliding with the environment. The polygon edges without neighbours exert a repulsive force on agents depending on the distance; (iii) The goal force to guide agents to their destination. This force is obtained from the shortest path found by the A* and the funnel algorithm. The parameters of the social forces model were tuned by trial and error and cannot be modified by the users.

The FLAMEGPU module needs to store information about every agent. This data includes id, position, destination, storyboard followed, among others. The full list of agent properties is shown in the listing below. Not all this information is required by Unreal to render the crowd; it only uses a random texture, id, position, velocity to obtain the rotation, and speed.

```
struct agent
{
    int id;
    int texture;
    int navmesh;
    int entrance;
    int exit;
    int story;
    int group;
    int area;
    int behaviour;
    float x;
    float y;
    float velX;
    float velY;
    int currentPoly;
    int nextPoly;
    int storyPosition;
    float speed;
    float currentSpeed;
    float tempSpeed;
};
```

3.3.4 Sketching

The interface, implemented in *Unreal*, allows the user to perform a series of actions by sketching or clicking in the environment. These actions include definition of agent spawn and goal locations, sketching obstacles to alter the crowd movement, creation of flow lines to guide the motion of the agents, drawing areas to create waypoints, and definition of journeys via storyboards. Figure 3.27 shows the menu interface of the navmesh-based system. The options are described in the following list:

- Select: Click on elements previously created to open a configuration window.
- Entrance/Exit: Select a polygon edge to create an entrance/exit.
- Barrier: Sketch a line to create a barrier.
- Flow line: Sketch a line to create a flow line.
- Area: Sketch a shape on the environment to create an area.
- Storyboard: Opens a menu to create/edit storyboards. Chapter 4.
- Group: Sketch a shape to select a group of agents and create a storyboard specific to the group. Section 4.1.
- Show timeline: Open timeline interface to create events at specific times. Section 4.2.
- Show navmesh: Show/hide navmesh.
- Show info: Open information panel.
- Simulation Speed: Adjust the simulation speed.
- Camera Speed: Adjust the camera movement speed.
- Start: Start simulation.
- Quit: Quit simulation.

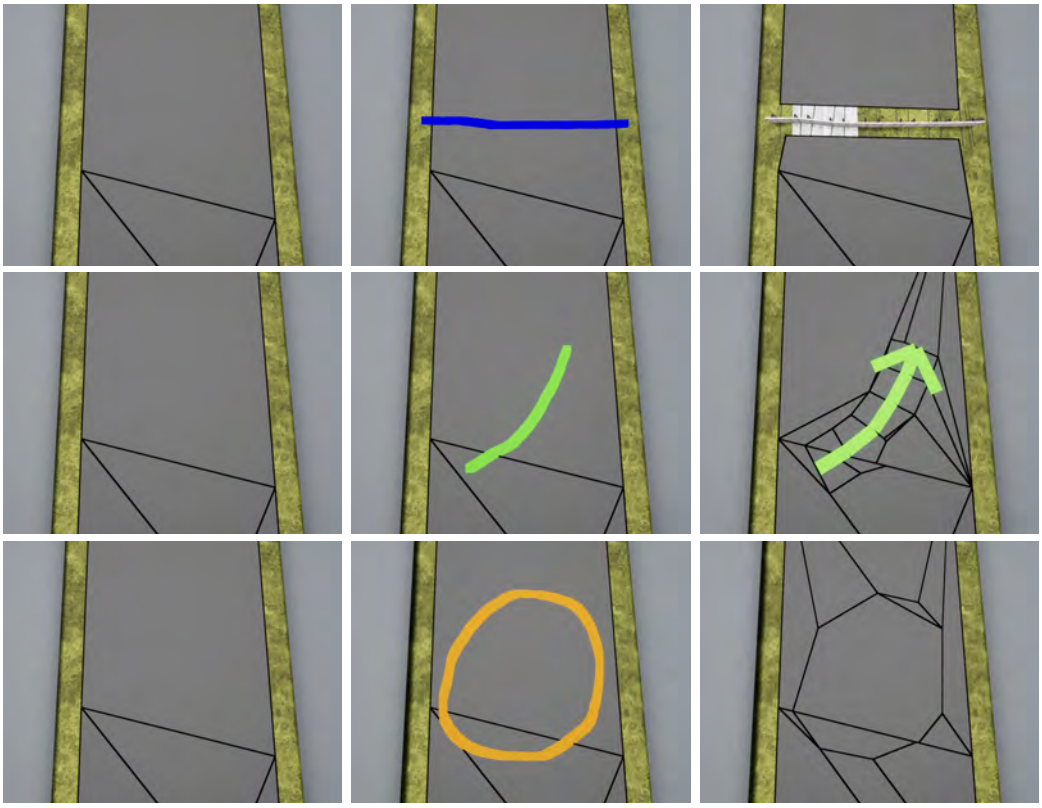
Figure 3.27: Main navmesh menu.

The first step to update the navmesh via sketching is to capture the user sketch and sample the line into equidistant points. Each sequence of points can represent an obstacle, a flow line or an area edge. Then, the line is mapped to the navmesh. This is achieved by marking the area covered by the sketch in the navmesh. These regions are given an id to differentiate among obstacles, flow lines and areas. The tiles affected by the user sketch are identified, and the navmesh of these tiles is rebuilt with the new information. Figure 3.28 shows the sketching process. Column (a) shows the initial navmesh, column (b) illustrates the three user strokes. Each colour line represents a different element: blue for barriers, green for flowlines and orange for areas. The elements created and updated navmesh are shown in column (c).

Entrances and Exits

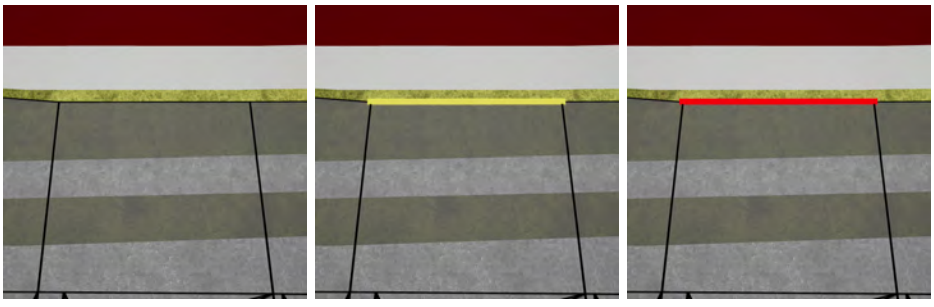
The entrances and exits are created by selecting a polygon edge with no neighbours. These locations define the spawning position of the agents and also serve as goals. Figure 3.29 illustrates the creation of an entrance (yellow) and an exit (red). The user can set the number of pedestrians to be spawned, the emission rate and the exit probability per entrance.

Entrances and exits store information needed for agent navigation. This data includes



(a) (b) (c)

Figure 3.28: (a) Original navmesh. (b) User-sketched lines. (c) Updated navmesh and the elements created: barrier, flowline and area.



(a) (b) (c)

Figure 3.29: (a) Original navmesh. (b) Yellow entrance created on the polygon edge selected. (c) Red exit created on the polygon edge selected.

the number of entrances/exits, state (open/close), the index of the polygon and the edge. Entrances also store the number of storyboards assigned. The code snippet below illustrates this information.

```
struct entrance
{
  int count;
  int *noStoryboards;
  int *state;
  int *poly;
  int *edge;
};

struct exit
{
  int count;
  int *state;
  int *poly;
  int *edge;
};
```

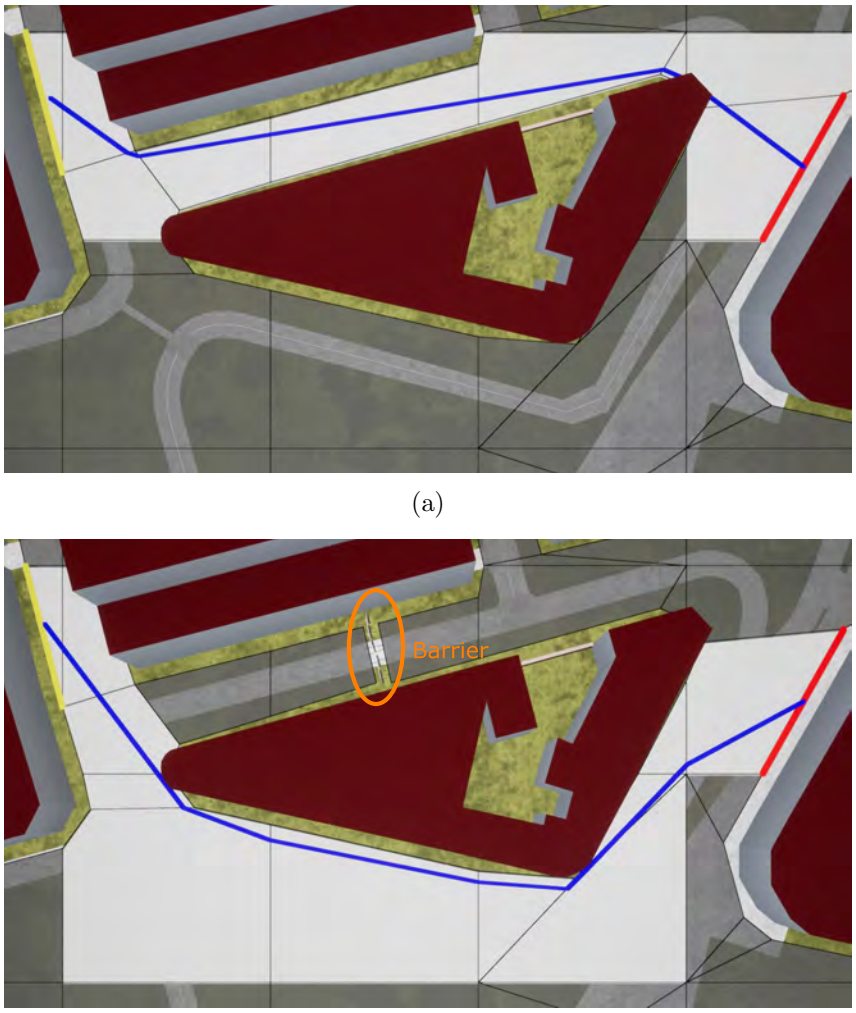
Barriers

The barrier obstacles are created by marking the affected navmesh area as null. A null area cannot be crossed and is not used in navigation computation. The process is made efficient by using the tiles that the relevant navmesh area overlaps. Each overlapped tile is divided into an integer grid of voxels, the size of which can be controlled by a *Recast* parameter. Every voxel in the grid is tested to determine if it lies within the sketched obstacle region, at which point it is marked as empty. (The use of integer grid coordinates speeds up the calculation process, but has implications when sketching flow lines that cross tile boundaries—see next section.) Using this information, the contours of the updated walkable areas inside the affected tiles are calculated, and these are used to re-triangulate this area to obtain the new polygons of the navmesh. Figure 3.28 shows the process of producing a barrier by sketching a line—navmesh polygons are generated on both sides of the barrier. Figure 3.30 shows how the addition of a barrier changes the shortest path from the starting point (yellow entrance) to the goal (red exit).

Flow lines

A similar procedure to that for barriers is used to create flow lines. The main difference is that the area is given an id to identify it as a flow line, rather than labelled as null. The sketched flow line is divided into a set of polygons which are traversable only in the direction of the sketch (Figure 3.28). Currently, the size of these polygons is not configurable by the user. The cost of traversing the flow lines can be changed individually—a higher value means that a flow line is more likely to attract agents from the surrounding area.

The addition of flow lines converts the navmesh in the flow line area into a directed graph which means that adjacent polygons are not necessarily connected for navigation purposes. Therefore, agents inside flow lines must follow the complete flow line until the end of it is reached.



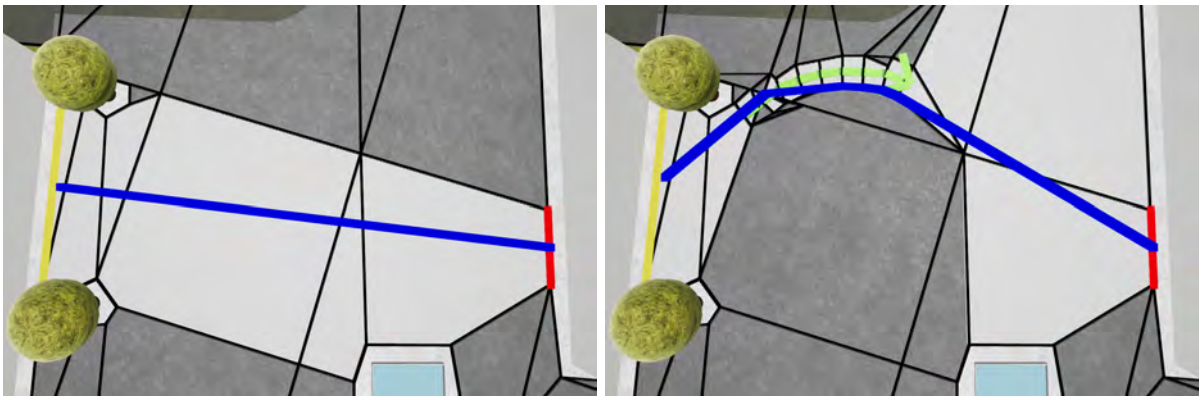
(a)

(b)

Figure 3.30: (a) Shortest path in blue from the entrance (yellow) to the exit (red). (b) A barrier has been added and the shortest path is recalculated.

The routes to areas and exits are recalculated when the cost of the flow lines is changed. Figure 3.31 shows how flow lines impact the path followed by agents. The new path is no longer the shortest, but the preferred path since flow lines attract pedestrians.

Sketching flow lines that cross tile boundaries complicates the process of creating a new navmesh. These situations occur when the same flow line area is mapped to two abutting tiles, as shown in Figure 3.32. Numerical conversion issues can result in a misalignment between adjacent flow line polygons. When mapping the sketched flow line to the tile, a conversion from floating-point (line coordinates) to integer (tile coordinates) is performed, and this conversion may produce a misalignment. This problem generates small gaps where pedestrians can 'escape' from the flow line. In Figure 3.32, pedestrians would be able to leave the flow line early. The problem was solved by modifying the adjacency conditions of polygons from different tiles. Regular polygons from one tile are connected to the flow line polygons of the adjacent tile, but



(a) (b)

Figure 3.31: (a) Shortest path in blue from the entrance (yellow) to the exit (red). (b) A flow line has been added and the path is recalculated.

not vice-versa.

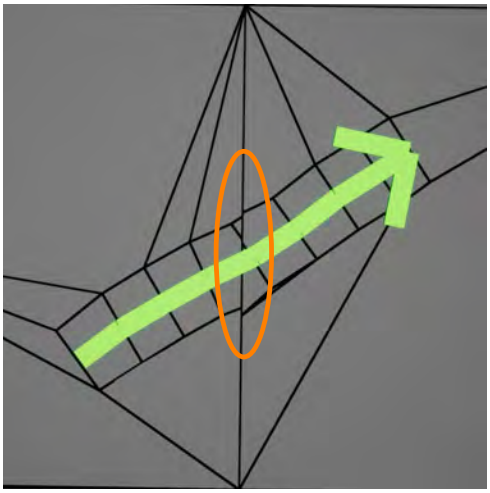


Figure 3.32: Flow line polygons misaligned when crossing tile boundaries.

It is possible to overlap flow lines, and pedestrians are free to move between flow lines at a crossing point. In Figure 3.33, pedestrians follow their specific flow line without having problems at the crossing point. The weight of the last sketched flow line will be assigned to the intersection polygon. Pedestrians may be forced out of a flow line when it is crowded; however, they will try to re-enter unless the shortest path is found from their new navmesh polygon to their destination.

Areas and behaviours

The user can create areas in the environment that serve as waypoints. The shape of the sketched polygon is mapped to the navmesh, and the new area is re-triangulated to eliminate concave polygons. The first and last point of the sampled line are connected to ensure a closed polygon.

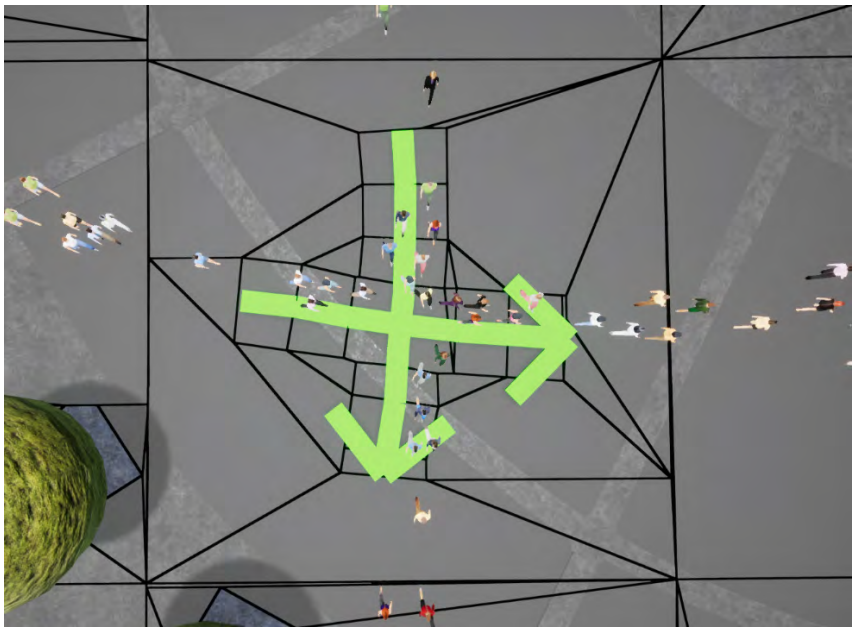


Figure 3.33: Pedestrians following overlapped flow lines.

The shortest path from each navmesh polygon to the area is then computed. In addition, the interface offers the ability to set the percentage of agents that will visit each area.

Users can select a behaviour that affects agents when they enter the area. Currently, two behaviours are implemented: (i) Wandering – when an agent reaches an area, it moves in random directions within the area for a predefined amount of time, before continuing on its journey; (ii) Waiting – agents wait in the centre of the area for a predefined amount of time, before continuing on their journey. While the idea of areas and behaviours has been previously suggested [66], the main difference in the approach in this thesis is the ability to set the percentage of agents visiting and to assign the areas to a storyboard (see Chapter 4).

3.3.5 Results

A set of scenarios were simulated to demonstrate the system's functionality. Chapter 4 shows more complex result scenarios such as marshalling a crowd through a city centre. A video of the system in action is available at https://tinyurl.com/y4qfkqb2. The scenarios were run on the PC described in Section 3.2.4. The performance of the system is shown in Table 3.2.

No. Agents	fps
1k	118-119
5k	86-87
10k	50-52
20k	24-26

Table 3.2: System performance with multiple crowd sizes.

The first scenario is the creation of a queue that simulates the entrance to a venue. Figure

3.34 shows a snake-like corridor created by sketching barriers.

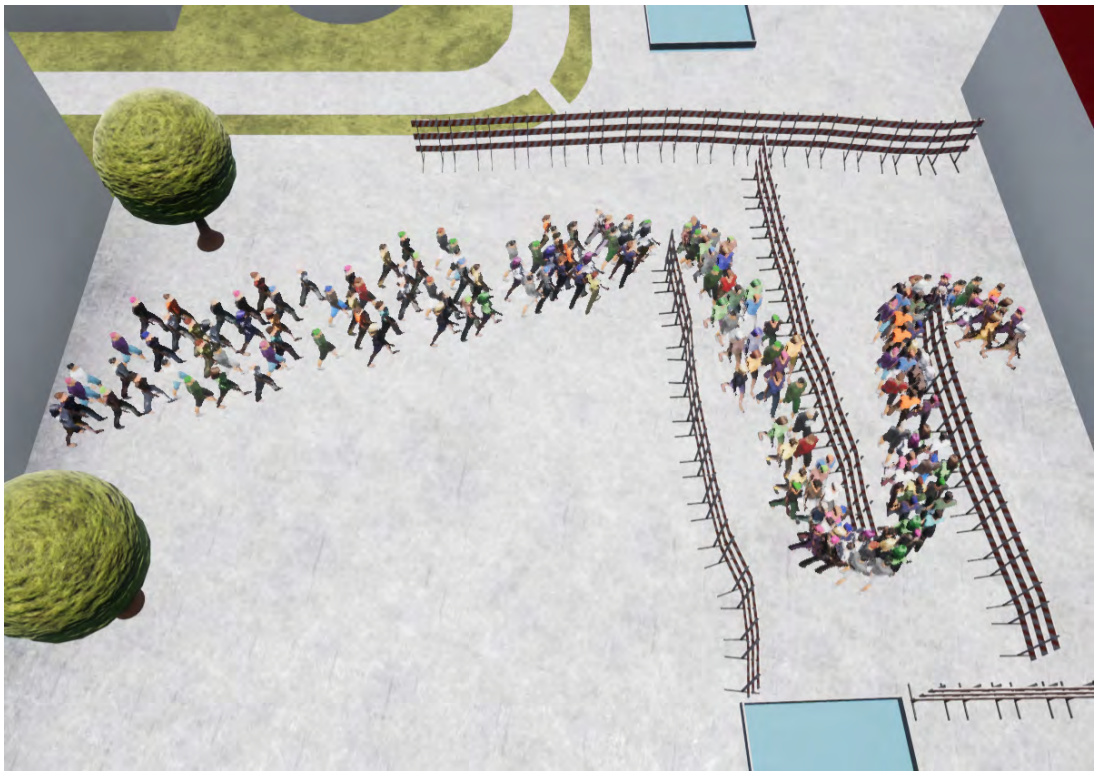
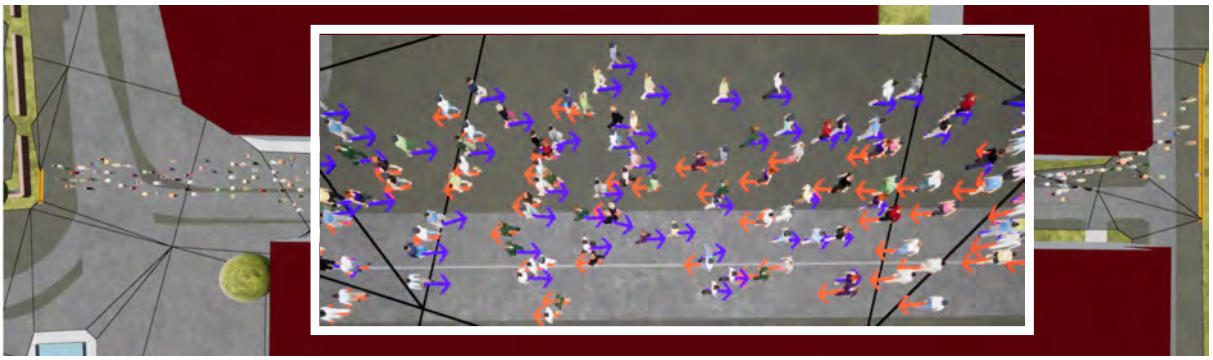


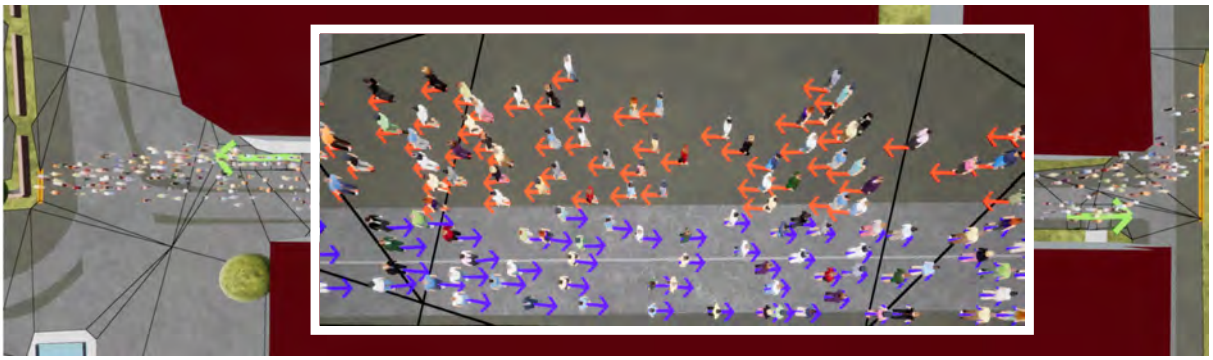
Figure 3.34: Pedestrians adjusting their path after the user created a corridor by sketching barriers.

A more illustrative example is shown in Figure 3.35, where two crowds move in opposite directions in a corridor. Multiple lanes are formed when both crowds meet, as seen in Figure 3.35a. The number of lanes can be reduced by sketching one flow line at each side of the corridor, towards the ends of the corridors (Figure 3.35b). These flow lines attract pedestrians moving in the same direction finally producing only two lanes.

Figure 3.36 shows the use of areas, which can be used as waypoints on a journey for a range of scenarios, e.g. where some pedestrians have to queue at a ticket machine before continuing in a train station or perhaps where pedestrians are stopping to watch some kind of street entertainment before continuing. In this example, the percentage of agents visiting the area, which is user-controllable, is altered from 50% in Figure 3.36a to 90% in Figure 3.36b.

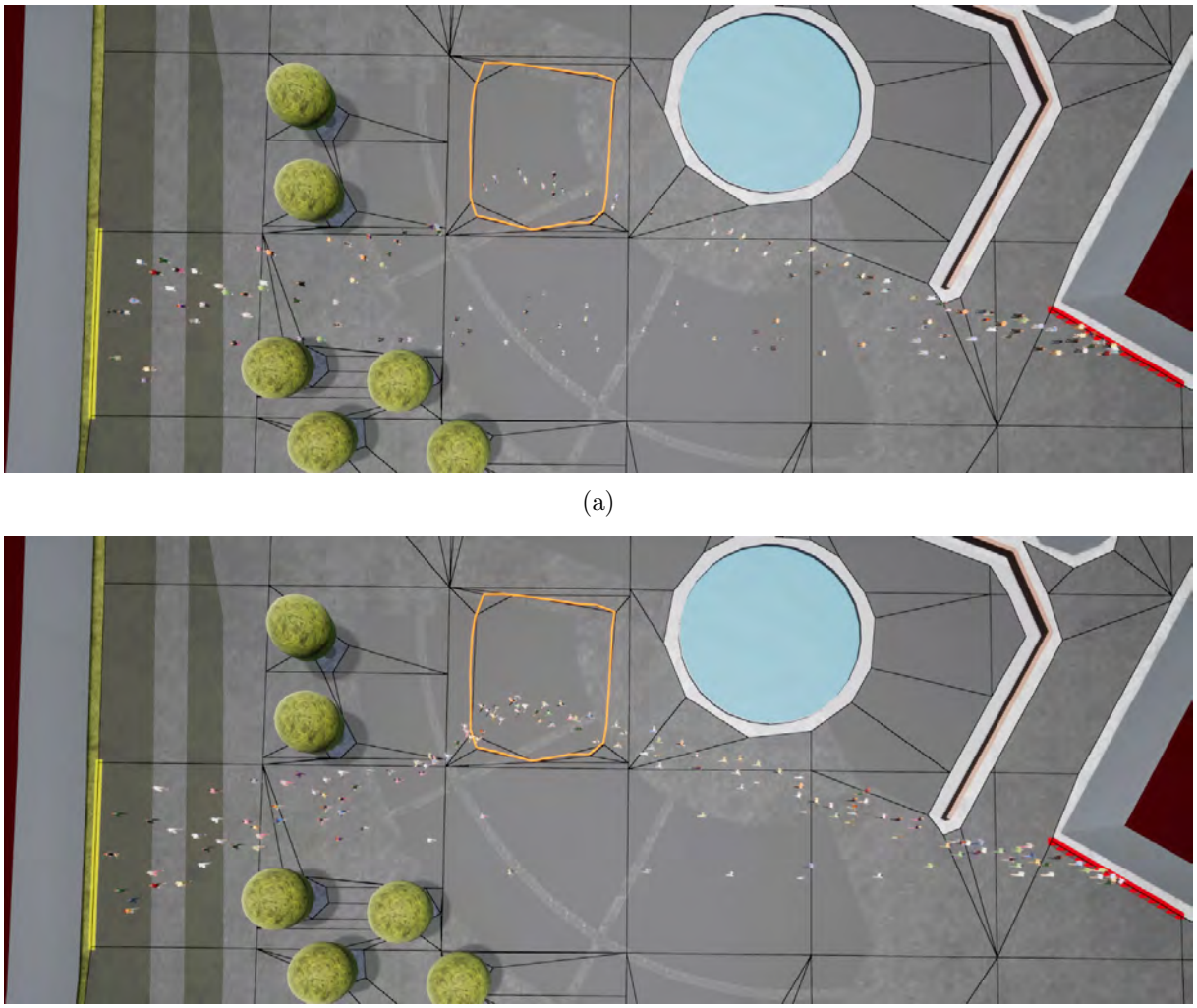


(a)



(b)

Figure 3.35: Pedestrians walking in opposite directions inside a corridor. (a) Multiple lanes are formed when the crowds meet. (b) Two flow lines are sketched, one at each end of the corridor, to control the number of lanes.



(a)

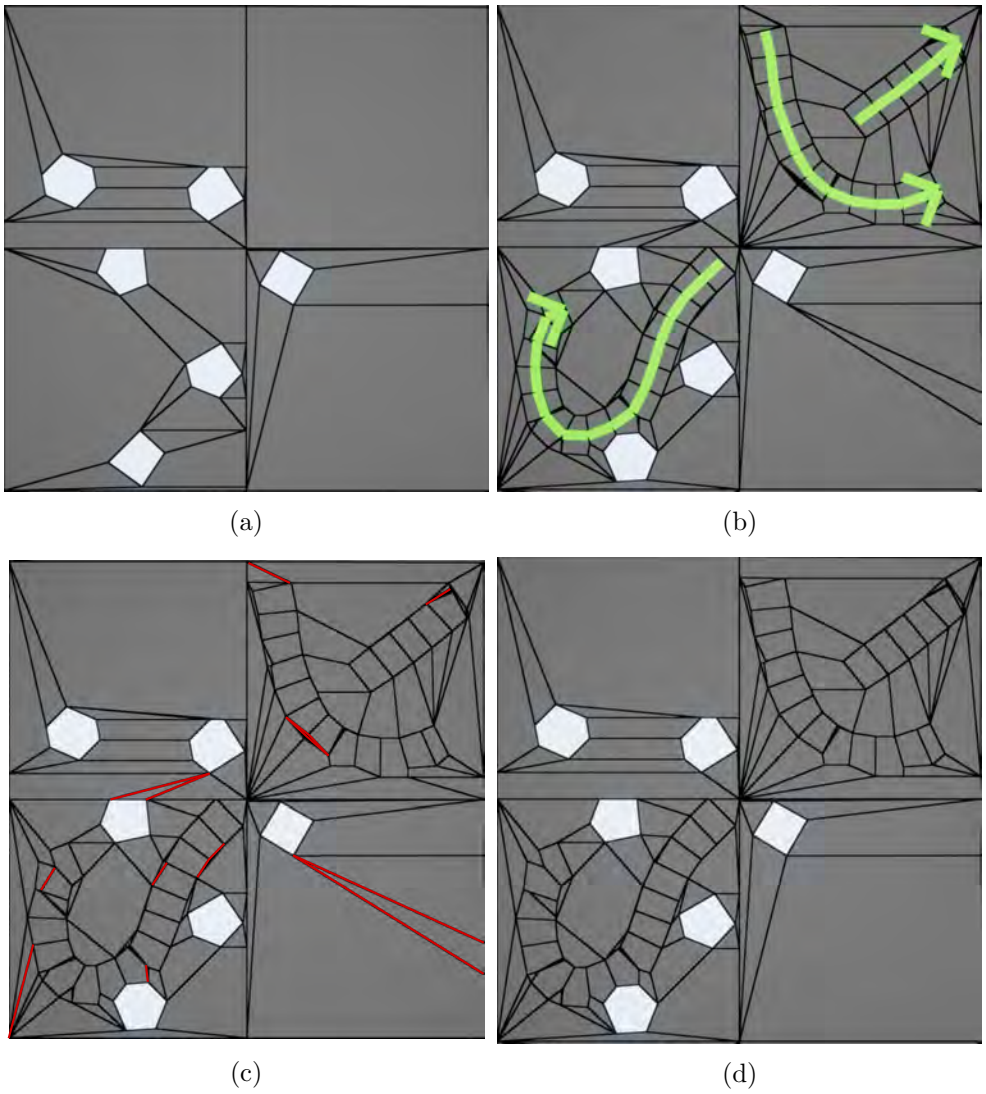
(b)

Figure 3.36: Pedestrians walking from the yellow entrance to the red exit via the same area (represented by the orange rectangle) with two different user-controllable area visit percentages: (a) 50%; (b) 90%.

Limitations

The navmesh is a better alternative for navigation since it represents the environment more accurately and requires less memory than the grid-based approach, although computation time is similar for both approaches. However, the navmesh approach has some limitations. First, the initial construction time (which can be done in an offline step) can be long, depending on the complexity of the environment. Second, as environments become more complex, computation times can become more of a problem. Complex environments require more convex polygons to cover all the walkable surfaces. Therefore, the creation of the navmesh and the pathfinding algorithm are more computationally expensive. The addition of each barrier, flow line or area increases the number of polygons in the mesh.

Another issue is that sketching flow lines produces some long, thin and sometimes unnecessary polygons that could be merged into a single polygon. To address this, the triangulation algorithm used by *Recast* was improved to merge some redundant polygons to reduce the pathfinding time. Figure 3.37 shows the result of the new algorithm in four tiles of the navmesh after sketching three flow lines. The original tiles and the flow lines created by the user are shown in Figure 3.37a and Figure 3.37b, respectively. Figure 3.37c shows the 148 polygons created with *Recast's* original algorithm. With the improved algorithm, this number is reduced to 130, as illustrated in Figure 3.37d. Typically, an initial navmesh does not have many tiles with complex areas or polygons. In these cases, the polygon reduction is not considerable, but increases as the environment gets more complex with barriers and flow lines.



(a) (b)

(c) (d)

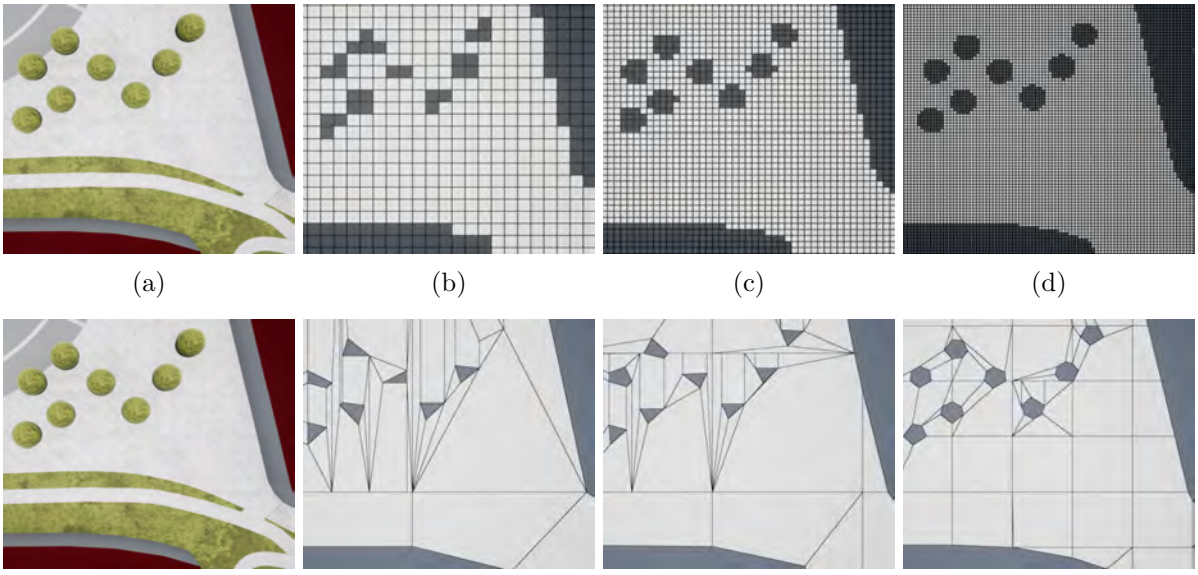
Figure 3.37: (a) Four tiles of the navmesh. (b) The same four tiles with three flow lines sketched by the user. (c) *Recast's* polygon merge algorithm creates 148 polygons. Edges that are eliminated by the improved algorithm are highlighted in red. (d) The 130 polygons resulting from the improved algorithm.

3.4 Grid vs Navmesh

This section compares the grid-based approach and the tiled navmesh navigation of this work using three criteria: environment representation and sketch accuracy, memory usage and computation time. The environment shown in Figure 3.25b is used as the scenario.

3.4.1 Environment representation and sketch accuracy

The representation accuracy of the grid depends on its resolution. Low-resolution grids use bigger cells which cover larger areas of the environment, e.g. walkable surfaces and obstacles. However, a cell can only be marked as empty or occupied, leading to a misrepresentation of the environment. This issue is evident in objects with circular shapes and when straight walls are not aligned with the grid, as shown in the top row of Figure 3.38, and also when curves and lines are sketched at angles to the underlying grid. More accuracy can be obtained by increasing the resolution of the grid, but increasing the resolution has an impact on memory usage and computation time.

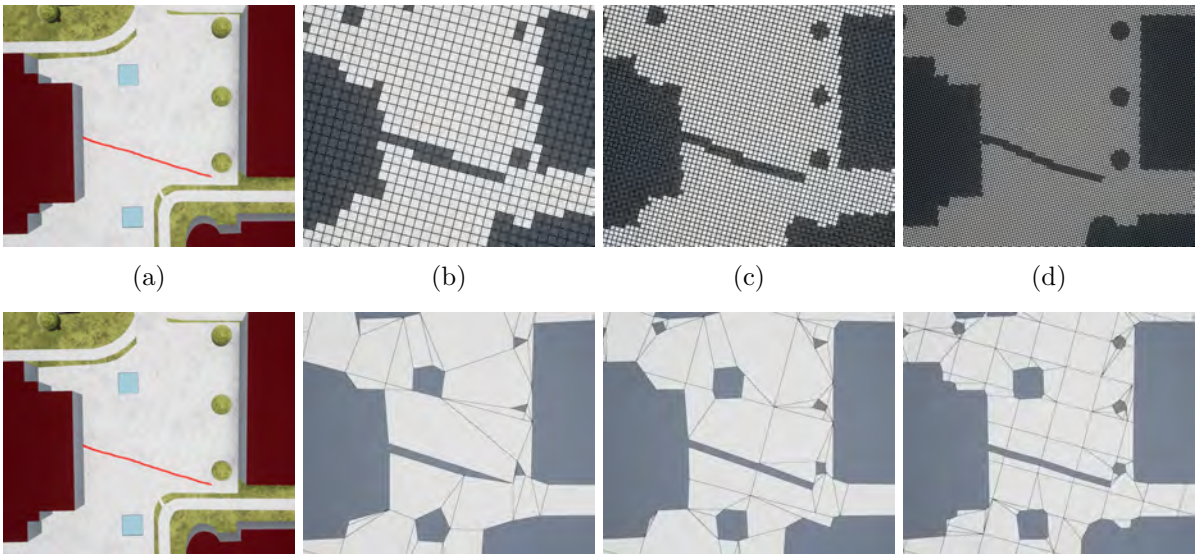


(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 3.38: Grid and navmesh representation of the environment. Top row shows three grids with different resolutions: (a) original environment, (b) 128x128, (c) 256x256 and (d) 512x512. Bottom row shows three navmeshes with different voxel size: (e) original environment, (f) 0.5, (g) 0.25 and (h) 0.1.

The navmesh provides a more accurate representation of the environment since it is based on the geometry of the model. Reducing the voxel size results in a better representation but also increases the number of tiles and polygons. An advantage of the navmesh over the grid is that polygons can cover larger areas compared to the cells of the grid; therefore, fewer polygons are required to cover the entire walkable surface. The bottom row of Figure 3.38 shows the same zoomed area of the environment represented by navmeshes with different voxel size. As the



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 3.39: Grid and navmesh sketch accuracy. Top row shows three grids with different resolutions: (a) original environment, (b) 128x128, (c) 256x256 and (d) 512x512. Bottom row shows three navmeshes with different voxel size: (e) original environment, (f) 0.5, (g) 0.25 and (h) 0.1.

voxel size is reduced, the number of polygons increases and the accuracy of the representation improves. Sketch precision is also better for the navmesh approach since the navmesh does not require a small voxel size to represent a line in a reasonably accurate way. In contrast, the grid approach must increase the resolution of the grid. This can be observed in Figure 3.39, a voxel size of 0.25 is enough to represent the sketched line, whereas the highest resolution grid still shows some imperfections.

In both the grid-based approach and the navmesh-based approach, increasing the accuracy of the representation impacts on the memory used and the computation efficiency. Larger environments require more data to be accurately represented; therefore, a trade-off has to be made between environment representation, memory usage and computation time.

3.4.2 Memory usage

A grid is represented by a two-dimensional array with each element set to empty or occupied. However, an additional structure is required to store the shortest paths to the goals. These paths are stored in the navigation maps (as in [90]). The maps are grids where every cell stores a force directing agents to their target. Table 3.3 shows the memory used by grids of different sizes with multiple exits. Figure 3.40 illustrates that the memory used is directly proportional to the grid resolution and the number of exits. When the number of cells is quadrupled, the memory usage is also increased fourfold. The grid approach has poor scalability.

The navmesh is represented by a structure that stores tiles, polygons, vertices and polygon adjacency, as well as other information required to find paths between polygons. In addition, an

Table 3.3: Memory used by the grid approach with three different sizes.

			Structure memory (kB)			
			No. exits			
Grid size	No. cells	Grid memory (kB)	1	3	5	7
128x128	16,384	128	256	512	768	1,024
256x256	65,536	256	1,024	2,048	3,072	4,096
512x512	262,144	512	4,096	8,192	12,288	16,384

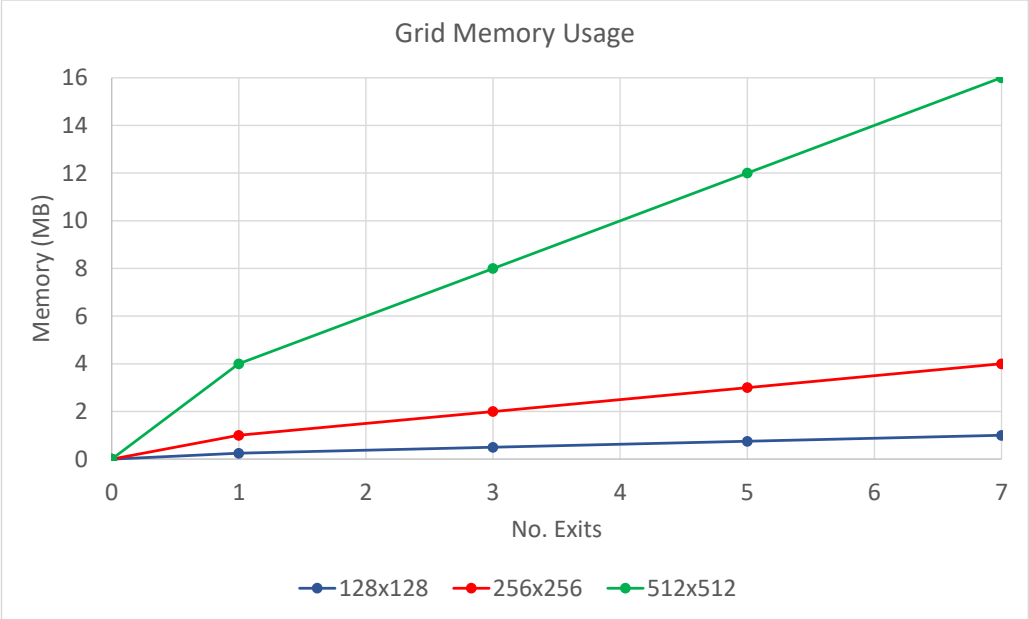


Figure 3.40: Memory used in megabytes (MB) by three grid sizes and multiple exits: 128x128 (blue), 256x256 (red) and 512x512 (green).

extra structure is defined to store the shortest routes, entrances, exits, areas, storyboards and a search grid used to accelerate the polygon search to determine the position of each agent. An advantage of the grid-based approach is that the location of the agent can be directly mapped to a grid cell when determining the next movement. Table 3.4 shows the number of tiles, polygons and vertices generated for three voxel sizes and also the memory used by the navmesh structure. Table 3.5 shows the memory used by the additional structure with different voxel sizes and multiple exits. Similar to the grid, memory usage also increases with the number of tiles, polygons and exits. However, the growth ratio is lower, as shown in Figure 3.41. Since the navmesh approach scales better with the size of the environment, it is a more suitable option for large environments in terms of memory usage.

Table 3.4: Number of elements generated by the navmesh with different voxel size, and memory used.

Voxel size	No. tiles	No. polygons	No. vertices	Memory used (kB)
0.5	44	516	1,363	80.2
0.25	168	911	2,643	163
0.1	921	2,213	7,310	499

Table 3.5: Memory used by the structure storing the shortest paths with multiple exits.

	Structure memory (kB)			
	No. exits			
Voxel size	1	3	5	7
0.5	260	263	265	267
0.25	379	387	386	390
0.1	701	710	719	728

3.4.3 Computation time

To compare the performance of both approaches, the time taken by three functions was measured: construction, update and pathfinding. A grid is built by using vertical raycasting to segment the world into cells. The update time is the amount of time taken to change the values of the grid when the environment is dynamically updated. Pathfinding uses a wavefront propagation algorithm to calculate the distance from each grid cell to the corresponding goal, and the result is smoothed to create more realistic paths. Table 3.6 shows the time taken by these functions for three grids with multiple exits. The update times are similar since only the affected cells are updated. However, the paths must be recalculated, which requires more time as the grid resolution increases.

Table 3.6: Time taken to create and update three grids of different sizes and to find the shortest paths for several goals.

			Pathfinding (s)			
			No. exits			
Size	Build (s)	Update (s)	1	3	5	7
128x128	0.071	0.00003	0.003	0.008	0.019	0.029
256x256	0.279	0.0001	0.031	0.050	0.075	0.135
512x512	0.944	0.0005	0.062	0.251	0.324	0.45

The process of building and updating a navmesh is described in Section 3.3. Table 3.7 shows

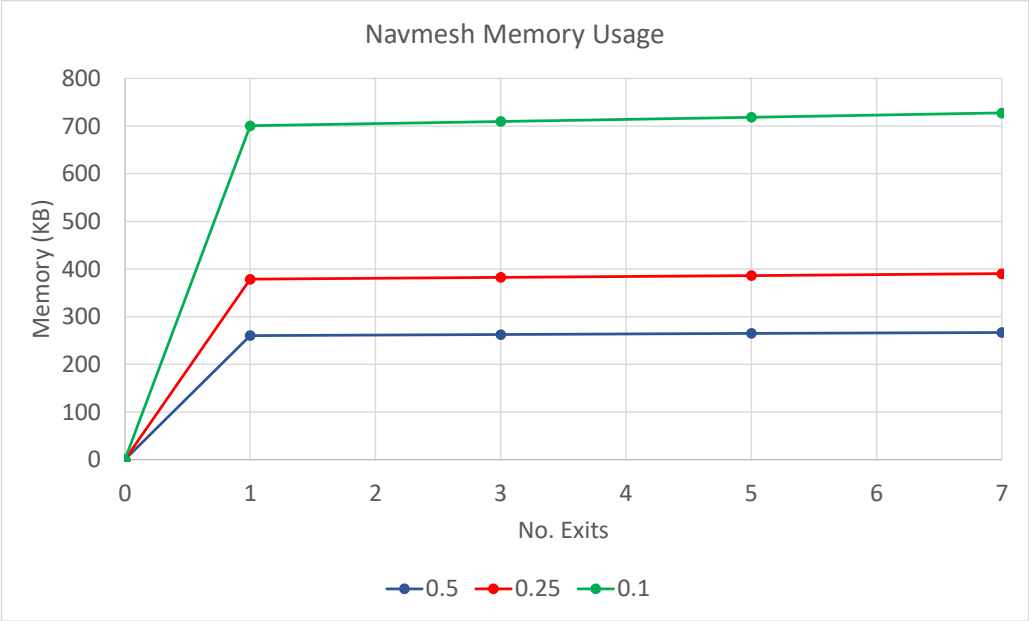


Figure 3.41: Memory used in kilobytes (kB) by three navmeshes with different voxel size and multiple exits: 0.5 (blue), 0.25 (red) and 0.1 (green).

the times for these processes and the pathfinding algorithm. The construction time is an issue for the navmesh approach. For a static environment, the navmesh can be created just once in an offline stage. For a dynamic environment, such as a sketching environment, the navmesh must be recalculated when the environment changes. However, using the tiled approach of *Recast*, where the mesh is updated locally, ameliorates the cost. The update time is similar for each case since the number of voxels per tile is the same.

Table 3.7: Time taken to create and update three navmeshes with different voxel sizes and to find the shortest paths for several goals.

			Pathfinding (s)			
			No. exits			
Size	Build (s)	Update (s)	1	3	5	7
0.5	0.134	0.003	0.003	0.009	0.015	0.024
0.25	0.492	0.002	0.009	0.031	0.054	0.078
0.1	3.433	0.004	0.076	0.253	0.427	0.597

Figure 3.42 plots the time taken by the pathfinding algorithm for different size grids and navmeshes. For the navmesh, the main issue is that this time increases as the number of polygons grows. Sketching barriers, flowlines, and areas creates more polygons, thus exacerbating the problem. Higher computation times could compromise the real-time interaction with the simulation. This may be an issue in complex environments or with a large number of destinations. Future work could look to implement a hierarchical pathfinding algorithm similar to the one

proposed by Pelechano and Fuentes [133]. The search could be divided into two levels: tiles and polygons. First, the path of connected tiles that leads to the goal could be found. This would include the path of polygons within each tile and the tile boundary joins. Thus, when the navmesh of a tile was updated, the path of polygons within the tile would be updated while still matching the boundary joins with abutting tiles.

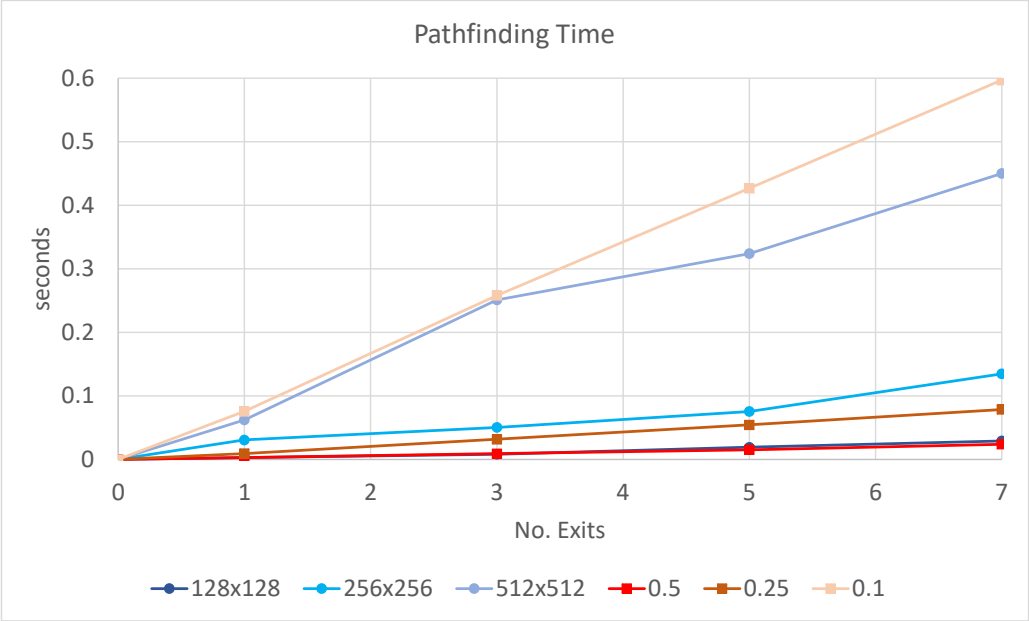


Figure 3.42: Time taken in seconds to calculate the shortest paths in grids and navmeshes with different sizes.

The comparison made in this section is specific to the settings tested. It also depends on the algorithm used to find the shortest path for each method. A more general and fair comparison requires the use of the same algorithm and a time complexity analysis. The time needed by the A* search algorithm depends on the solution depth d, branching factor b and the heuristic function $h(n)$. The worst-case scenario is a heuristic function that returns 0 for every node, this would produce a complexity of $O(b^d)$ [152].

The branching factor b of the navmesh depends on the number of sides of the polygons. This varies between 3 and 6. However, the number of edges does not always correspond to the number of neighbours of each polygon. Taking the average number of sides, the branching factor of the navmesh would 4.5. Regarding the grid, each cell is a square. Therefore, the branching factor is 4. Both navigation structures have a similar b without considering the heuristic function. However, the function prunes away nodes decreasing the branching factor and creating the effective branching factor b^*. An optimal function would produce a value of b^* close to 1. Since the same $h(n)$ is used for the grid and navmesh, it can be assumed that they have a similar b^*. Therefore, their time complexity is $O(b^{*d})$. The depth of the solution d depends on the number of nodes. Taking the examples in Section 3.4.2, the highest grid resolution produced 262,144 nodes. Whereas the navmesh with the smallest voxel size generated 2,213 polygons. Thus the grid would almost always have a larger exponential value d, leading to longer computation times.

A more thorough analysis is required to calculate the exact values of b^* and d for a given set of example scenarios.

3.5 Summary

This section discusses the sketching process, sampling methods, techniques to improve the sketch, and mapping the final stroke to the environment. Then, the section summarises the contributions of the work done on sketching on top of grid and a navmesh. Lastly, the advantages of using a navmesh over a grid are summarised.

Computer-based sketching needs to sample the user input to interpret the stroke. Sampling is used to eliminate noise by reducing the number of points captured. This process can be done on the fly or after completing the stroke [126]. The approach selected to do the sampling is polyline approximation, where the stroke is represented by line segments of the same length. Once the sampling is completed, the sketch can be improved by fitting it to other representations such as Bézier curves and B-splines. A different technique, called Beautification [67], reads the user input and beautifies the stroke taking into account geometric restrictions such as parallelism, perpendicularity and symmetry. A relative straightforward sampling process without post-sampling techniques was used in this research since it was enough to reduce noise and interpret the stroke. However, other approaches could have been used.

After processing the user input, the sketches must be mapped to the data structure that represents the environment. This chapter has investigated two navigation methods: grid (3.2) and navmesh (Section 3.3). A contribution of this work is a grid-based system updated by sketching. The system is related to Patil *et al*'s work [132] work, in that flow lines can be sketched to direct the crowd. However, additional features were included: sketch entrance/exit, sketch barriers, and the use of multiple navigation maps to guide the agents. Another contribution of the chapter, is the use of a navmesh to support the sketch-based control. The navmesh is updated by the user sketches in real-time. This is the main difference with Hughes *et al*'s work, which used sketching to generate the navmesh in an offline stage.

A comparison has been made between grids and navmeshes. The navmesh was the best alternative since it represents the environment more accurately, scales well with the size and complexity of the environment and presented similar path-finding times to the grid approach. Mapping the strokes to the navmesh is also more precise than the grid. The grid resolution is crucial for the mapping; if the cell size is large, then it is highly likely that the cells affected by the sketch also contain open areas. The problem is that a cell can only represent an open area or an obstacle. Mapping strokes to a navmesh also depends on the size of the voxels, but this could be small without having a considerable impact on the number of polygons. This process only consists of marking the affected voxels and recalculating the navmesh depending on the element sketched. A quick

Chapter 4

Extending the sketch-based approach

This chapter presents additional control options for more complex scenarios by extending the sketch-based approach. So far, this research has focused on sketching to create elements that modify the environment, but more advanced control options can be implemented by using and combining these existing elements. Furthermore, this chapter investigates sketching to directly change the behaviour of the agents by defining paths and behaviours. Then, a problem of dynamically updating the environment is discussed, and some solutions are proposed.

The work in this chapter presents four contributions. The first contribution is the creation of storyboards. Users can define the journey of the crowd by clicking on entrances, waypoints and exits to connect them. In contrast, the commercial system MassMotion handles this functionality by creating a 'journey', where users define in a window the origin and destination of the crowd. Any additional actions during the journey are specified in a separate window with more parameters. The second contribution is a timeline interface that allows the simulation of events at different times of the day. A similar idea was proposed as future work in [76], where the idea was to change the density and direction of the crowd to transition between hours of the day. However, no implementation of this idea was found. The third contribution is the creation of a sketch-based storyboard by linking paths and behaviours. These storyboards: are created by sketching lines, defining an exact path, and controlling only a group of agents selected by users. Hughes et al. [66] suggested the idea of linking events in a sequence to create the story of the crowd. However, no further work on this idea was found. The last contribution is the identification of a problem arising from sketching in real-time to modify the environment. Pedestrians gain knowledge immediately, even when they cannot 'see' the changes. This behaviour is unrealistic. Therefore, this chapter proposes some alternatives to address this issue.

Section 4.1 describes the implementation of storyboards, which are used to define the crowd's journey from the entrance to exit. Section 4.2 adds time to the simulation and the ability to create events at any point of the simulation using a timeline interface. This feature allows setting up scenarios simulating different times of the day. Two complex scenarios illustrate the functionality of these features in Section 4.3. Additional non-sketch-based options available in the system are described in Section 4.4. Section 4.5 explores controlling groups of the crowd. This option uses sketching to select a group of agents and define paths and behaviours.

Contrary to the storyboards, using this control directly modifies the agent parameters; thus, it is possible to specify the exact path to be followed. Section 4.6 discusses the problem of dynamic environment changes and proposes some solutions. Lastly, a discussion and summary of the chapter are provided in Section 4.7.

4.1 Storyboards

A storyboard defines the journey of agents throughout the simulation. Storyboards consist of an entrance, optional areas and an exit. Currently, up to ten storyboards can be created per entrance. The user can define the percentage of pedestrians (spawned at the selected entrance) that follow the desired storyboard. A menu displays the existing storyboards and offers editing facilities (Figure 4.1). If no storyboard has been created, agents spawn at every entrance depending on a user-defined emission rate. Similarly, their exit is selected from all the exits based on percentages specified by the user. By creating storyboards, the user defines the exact agent journey.

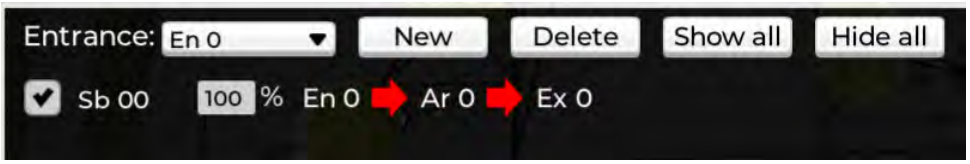


Figure 4.1: Storyboard menu where users can create and delete storyboards for every entrance.

Storyboards are created by clicking on an entrance polygon, then optionally selecting areas to finish the storyboard by choosing an exit. Once the route is completed, the storyboard is displayed, as shown in Figure 4.2. The selected polygons and areas are highlighted, and the indication arrows show the order of the storyboard. These arrows are not the actual path followed by the crowd. Individual pedestrians will calculate their path using the navmesh. The storyboard in Figure 4.2 goes from the left yellow entrance to the area in the middle and finishes in the right red exit.

Every storyboard has an id, stores the number of areas that compose it and the id of these regions, as shown in the following listing. Agents check their entrance and storyboard variables to obtain the route of the corresponding storyboard.

```
1  struct storyboard
2  {
3    int **noAreas;
4    int ***route;
5  };
```

Storyboards offer complete control over the paths to be followed by the crowd. Figure 4.3 shows three storyboards created by the user – different shades of the same colour are used for storyboards that belong to the same entrance, and each entrance is assigned a different colour for its storyboards. The crowd is divided into three equal parts, each following a different storyboard. Two storyboards have an intermediate area, and one directs pedestrians straight

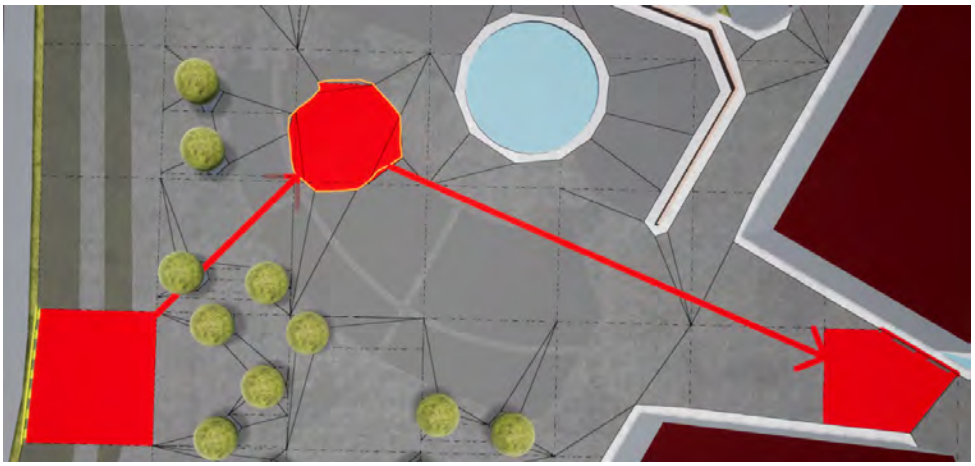


Figure 4.2: Storyboard created by the user. The selected polygons and areas are highlighted in red. The route begins at the left entrance, continues to the area in the middle and ends at the right exit.

from the entrance to the exit. It can be difficult to see the different shades of colour used for storyboards emanating from the same entrance because the system allows up to 10 storyboards from one entrance. This issue is something that still needs further work.

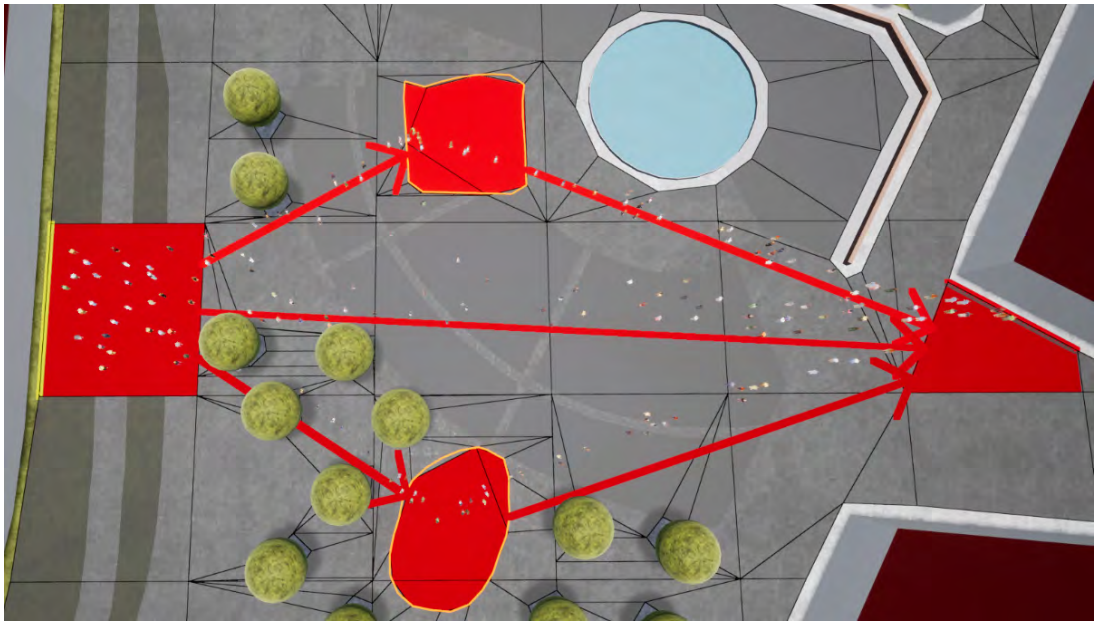


Figure 4.3: Pedestrians divided into three equal groups, each following a different storyboard.

4.2 Timeline

The simulation keeps track of the time, allowing the user to simulate events during a day. Example events include open/close entrances and exits; change the emission rate of an entrance;

create barriers, flow lines, areas and storyboards. In addition, the speed of the simulation can be increased up to 24x to observe a day in an hour. Creating events offers more flexibility and control to create complex scenarios simulating different times of the day, for instance, increase pedestrian flow during peak hours or closing entrances/exits at a specific time.

Using the timeline interface (Figure 4.4b), the previously created elements can be dragged and dropped into the timeline to specify their occurrence time. The duration of the events is modified by re-sizing the elements. The starting time and duration can also be entered manually in a configuration window. This window includes other parameters such as entrance emission rate and state. The simulation continuously checks for events and updates the environment accordingly to influence pedestrian behaviour. The elements that are not added to the timeline are considered permanent as part of the environment.

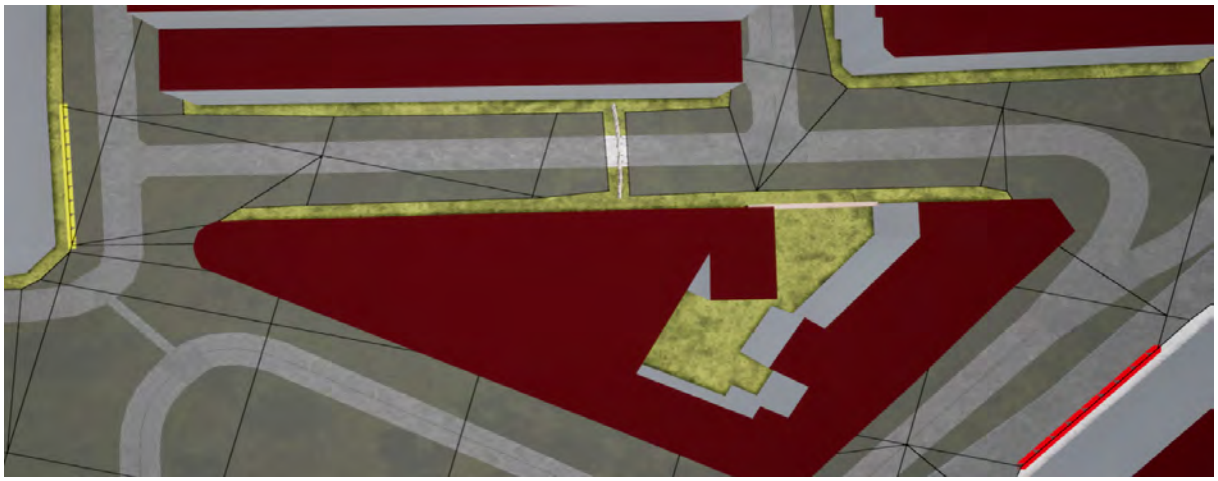
Events can be created, modified and deleted while the simulation is running. The system compares the current time and the time of the created/modified event to do the corresponding action or ignore the event. Figure 4.4 shows a simple example of the timeline interface. In this scenario, only two events are created. First, the emission rate of the yellow entrance is increased. Second, the barrier appears, blocking the path of the pedestrians and forcing them to walk around the building to reach the red exit.

4.3 Storyboard and timeline scenarios

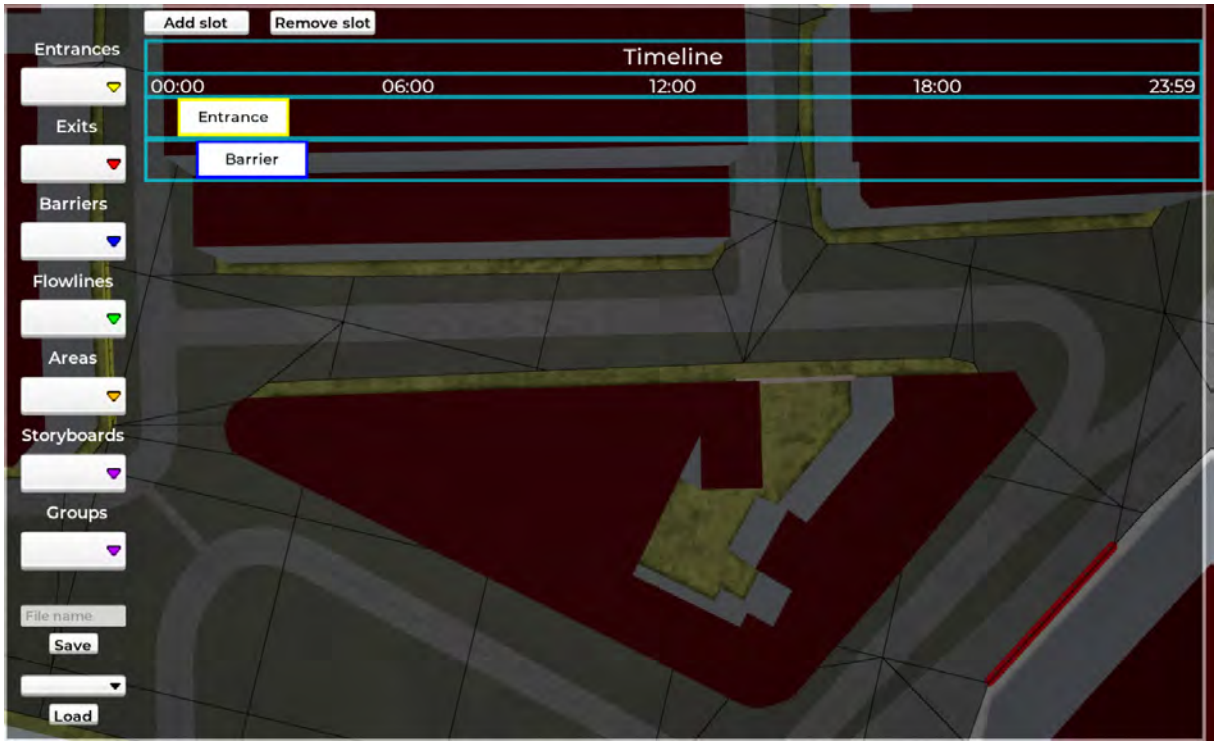
This section shows a couple of complex and practical scenarios using storyboards and the timeline interface. Figure 4.5 shows a scenario with 10 entrances/exits and 70 storyboards. In this example, a day in the city centre is simulated. Five entrances/exits are placed on the outskirts of the city centre, and the other five entrances/exits are located in buildings. This scenario simulates the flow of people going to work or visiting the area. Figure 4.6 shows the timeline for this complex environment. In the example, the emission rate of the named entrances is changed throughout the day to simulate two peak travel periods, as people travel to a city centre in the morning and then leave it in the evening. For example, the “Division” entrance has a high flow from 7-9am, then a low flow until 4pm. These people follow one of 5 storyboards to different buildings. For the building ”JLewis”, there is a low leaving rate from 10am-5pm, followed by a high leaving rate from 5-7pm. These people follow storyboards to exit routes (which are equivalent to the original entrance routes).

The use of storyboards combined with the timeline gives non-expert users control over the simulation to recreate real scenarios. One issue is that displaying all the storyboards at the same time can cause visual clutter, as in Figure 4.5. To reduce this, users can hide/show storyboards individually or grouped by entrances. An area to investigate in future work would be the use of decision trees in conjunction with storyboards.

The last scenario is a practical application that could be used to marshal a crowd through a city. An example would be a police force controlling a football crowd moving from a railway station to a football ground. A video of the scenario is available at https://tinyurl.com/y5cafnkj. Figure 4.7a shows the starting position of the crowd to be marshalled (Entrance C) and its destination (Exit A). Additional pedestrians are also simulated to walk around the city.



(a)



(b)

Figure 4.4: (a) Simple scenario with one entrance (yellow), one barrier and one exit (red). (b) Timeline simulating two events. The emission rate of the entrance is increased, and the barrier appears shortly after that.

Figure 4.7b highlights the routes, entrances and exits used by the agents that are not part of the marshalled crowd. Without any guidance, the marshalled crowd would follow the path shown in Figure 4.7c, a route which passes through the middle of the potentially congested city centre area. The objective of the scenario is to dynamically control the path of the crowd (shown in

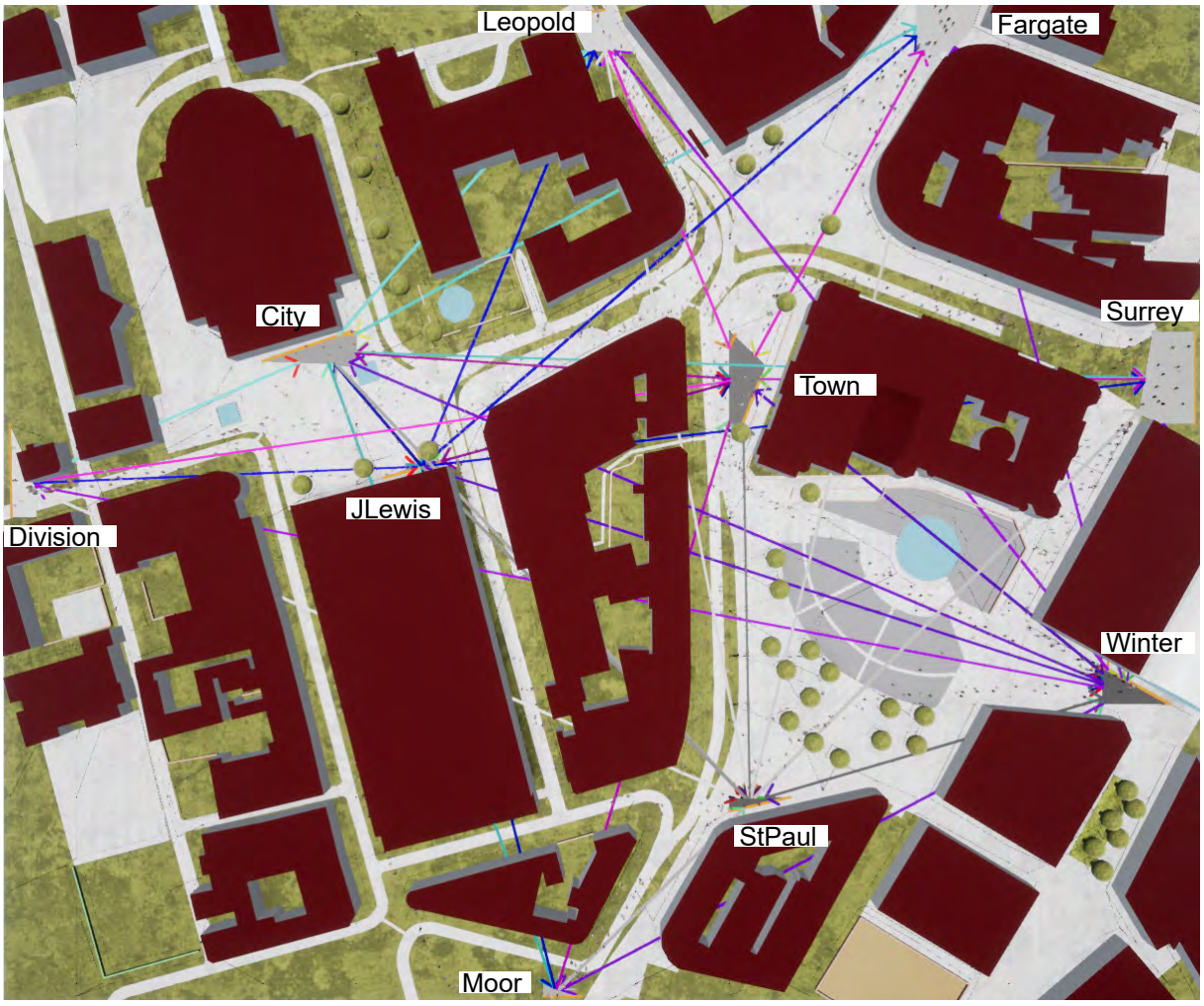


Figure 4.5: Complex scenario simulating a city centre with 10 entrances/exits and 70 storyboards.

Figure 4.7d) by sketching and deleting barriers to avoid congestion.

Particular frames of the simulated scenario are shown in Figure 4.8. At the beginning of the simulation (Figures 4.8a and 4.8b), barriers are sketched (blue lines) to prevent the crowd from going to the area that will be congested. These obstacles also affect pedestrians who are not part of the marshalled crowd. Figure 4.8c shows that pedestrians moving from Entrance A to Exit B need to readjust their path and walk around the buildings on the left to reach their destination. This new route interferes with the path followed by the marshalled crowd. Therefore, it is blocked when the crowd approaches (figure 4.8d). The front part of the crowd arrives at Exit A in figure 4.8e. As the marshalled crowd walks past certain areas, barriers are deleted, and pedestrians can retake their original path to their destination (figure 4.8f).

This scenario shows that it is possible to dynamically add and remove items such as barriers within the environment while a simulation is running. Multiple scenarios could easily be simulated in advance of an operation or tested in faster-than-real-time while a live situation

Figure 4.6: Timeline interface. The elements created by the user can be selected with the drop-down lists on the left. These can be dragged into the timeline to determine the time of the event.

is unfolding based on information feeds from on-the-ground operatives or from CCTV cameras. Challenger et al. [22] recognises that simulation tools are useful for strategic planning, safety and management of crowd events, but emphasises that these tools cannot replicate or predict the real crowd behaviour. Moreover, Challenger et al. [22] identifies a future research opportunity in exploring interactions between groups such as the police and a crowd. The marshalling scenario is a good example of how the sketching approach can be used in the previously mentioned areas. The system might not produce exactly the same result as the real crowd, but it offers the possibility to simulate unexpected events in real-time, which would not be possible in an offline stage. In addition, the barriers could represent a police force interacting and blocking the path of pedestrians that are not part of the marshalled crowd.

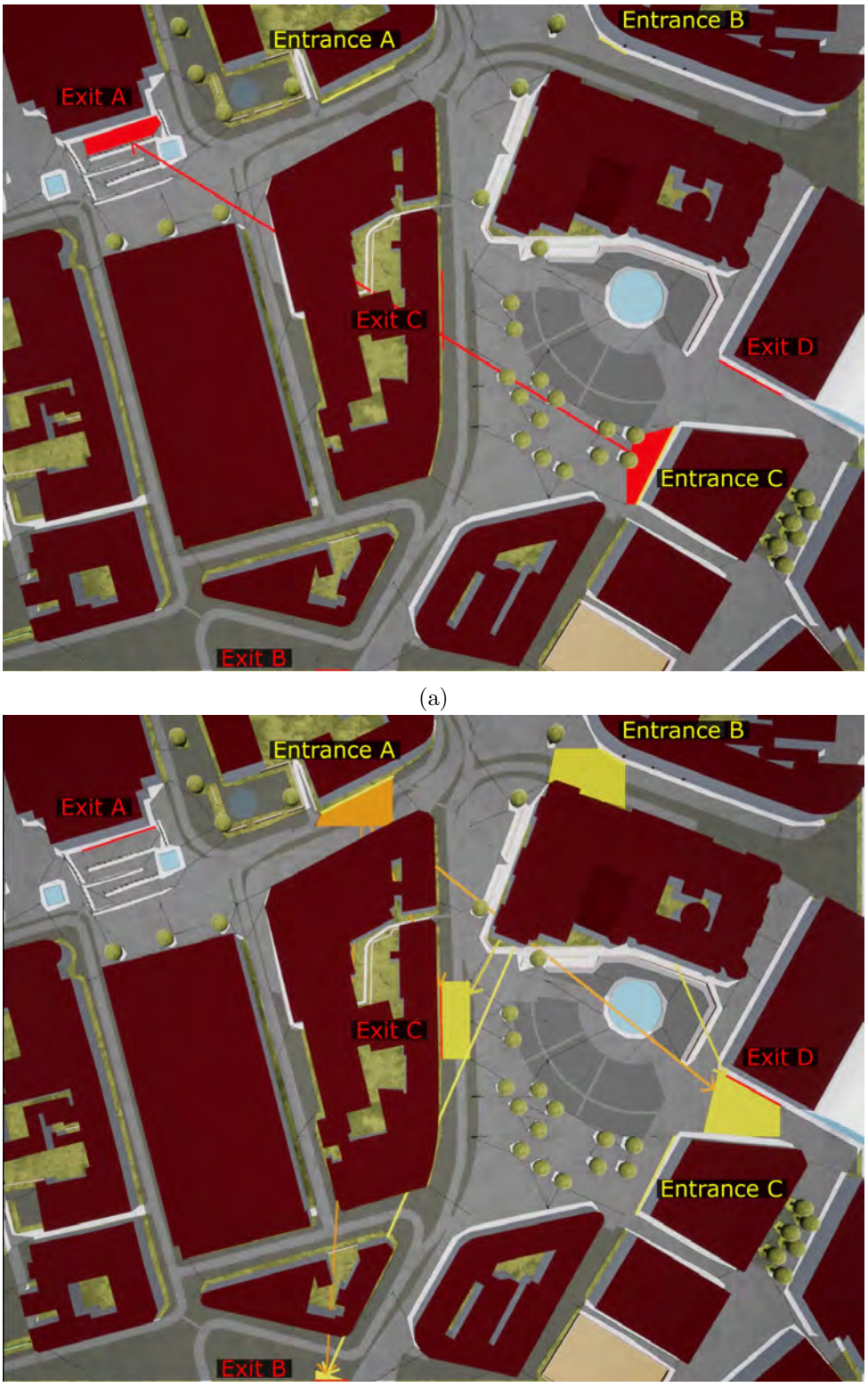
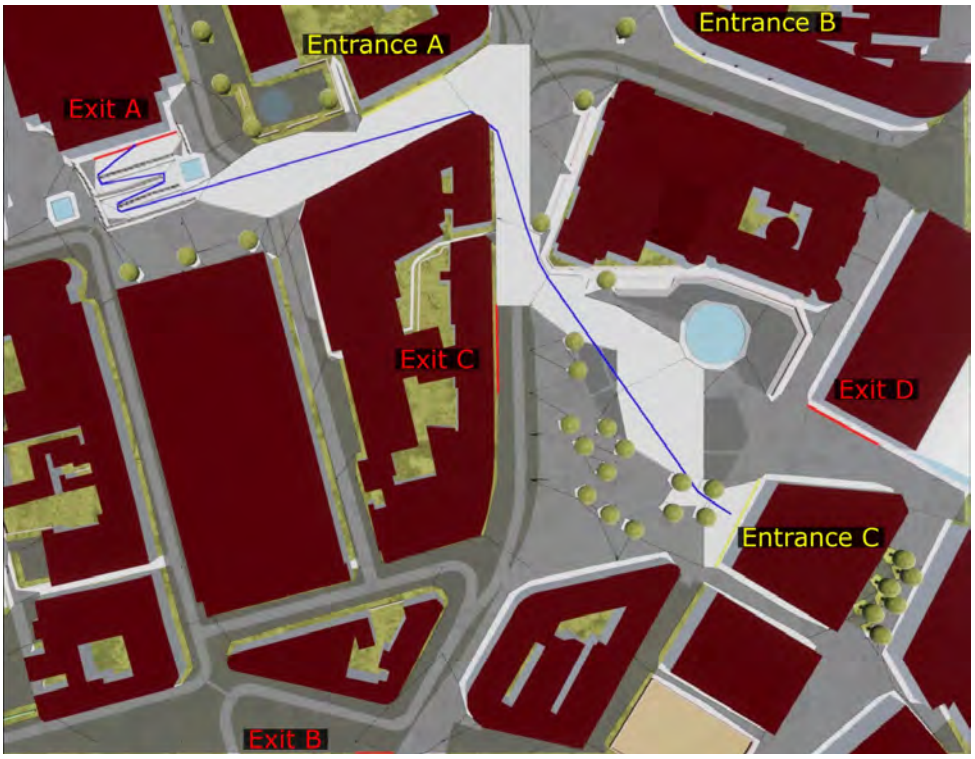
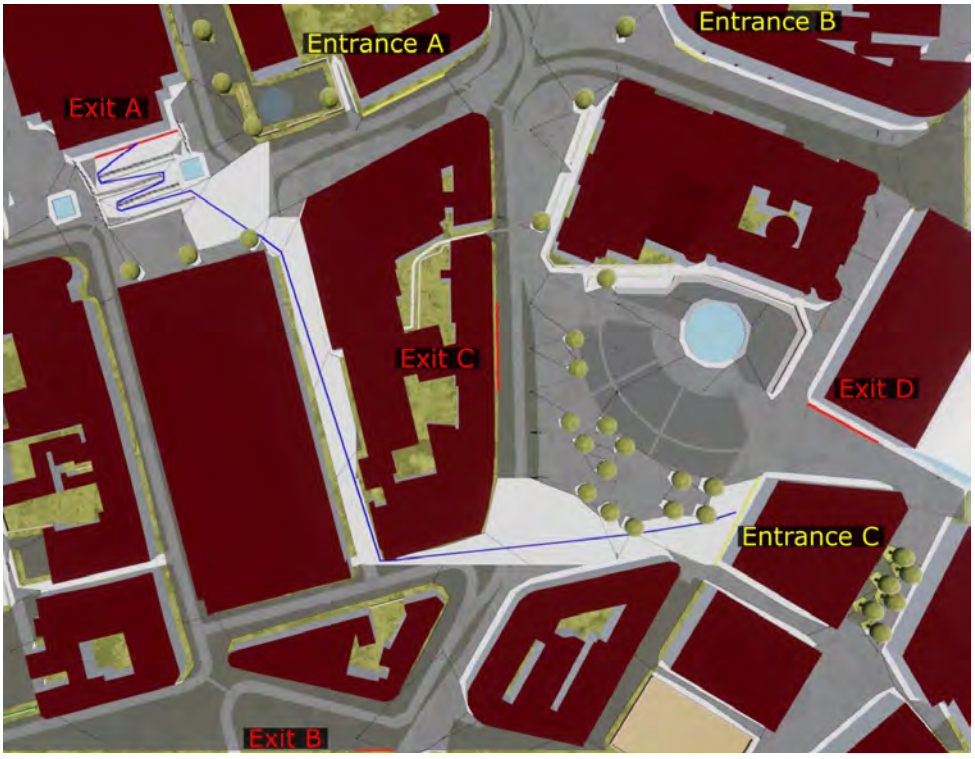


Figure 4.7: Crowd marshalling scenario. (a) The crowd to be marshalled spawns from Entrance C and moves towards Exit A. (b) Pedestrians walking around the city move from Entrances A and B to Exits B, C and D.

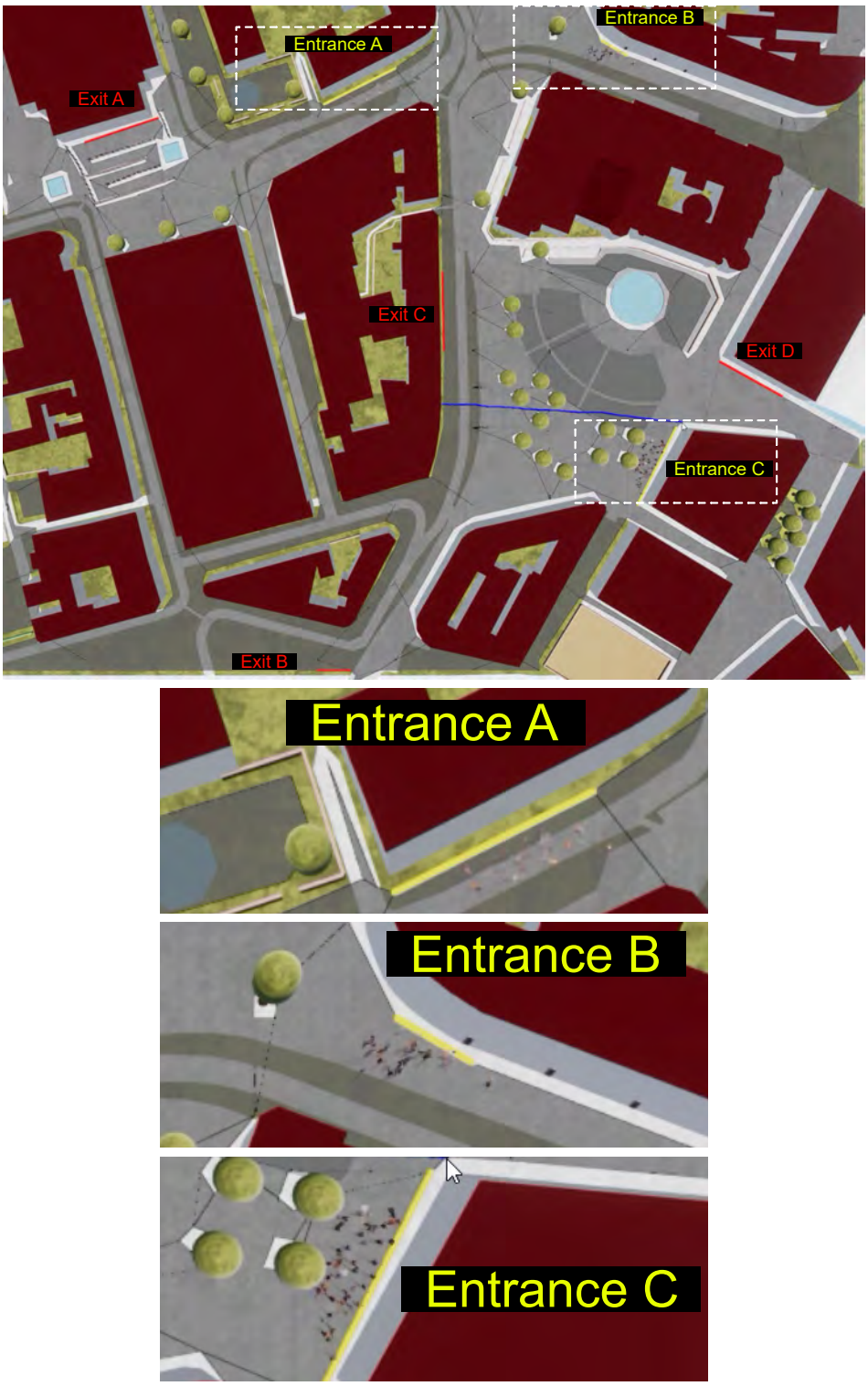


(c) (cont.)



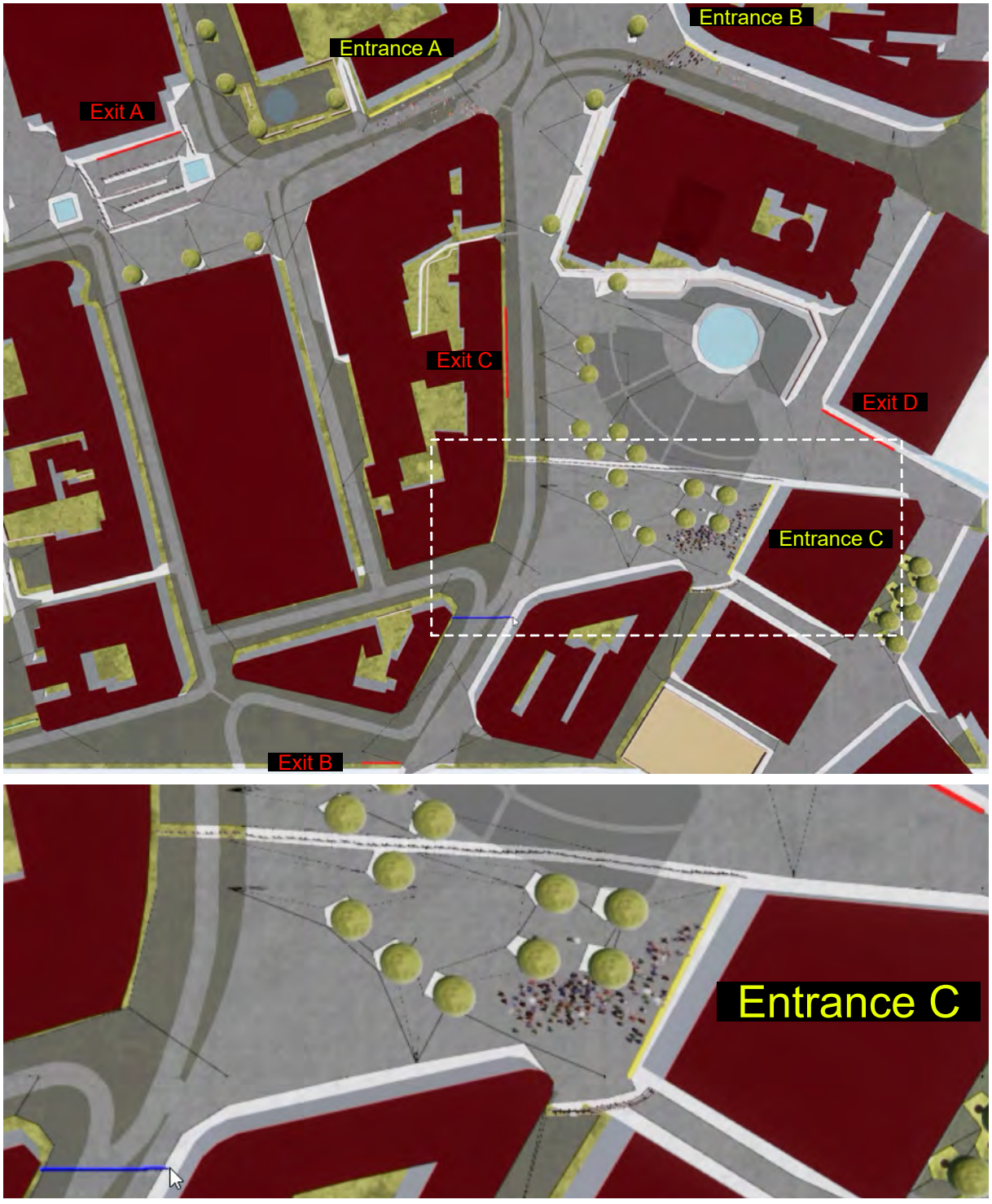
(d) (cont.)

Figure 4.7 (cont.): Crowd marshalling scenario. (c) The shortest path that the crowd would follow without user intervention. (d) The desired path to be followed by the crowd while being marshalled.



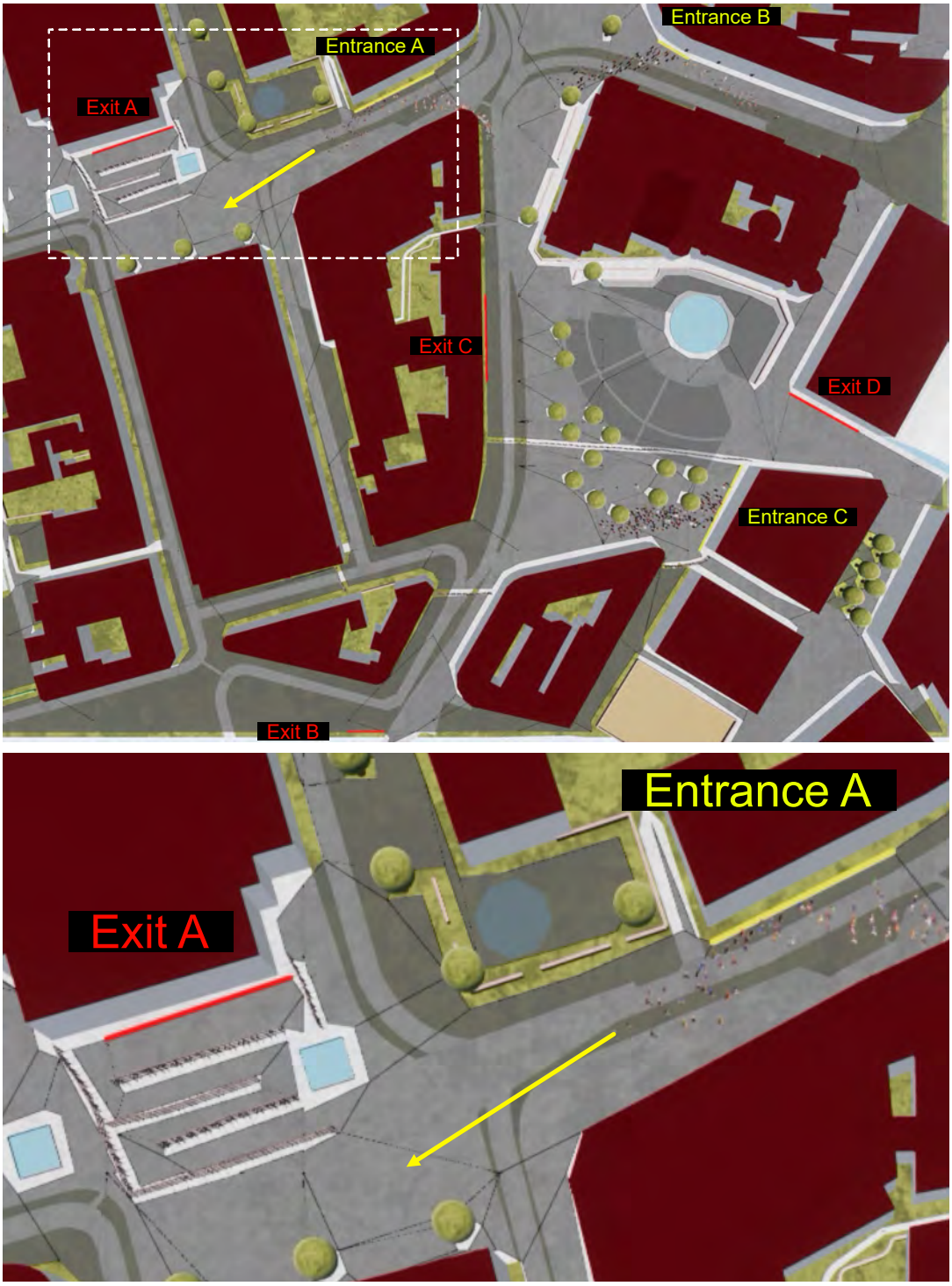
(a)

Figure 4.8: Simulation frames of a crowd being marshalled from Entrance C to Exit A in a city centre. (a) Barrier sketched (blue line) by the user to block the crowd's path and avoid a congested area. Highlighted areas in the environment are enlarged to better show off the crowd.



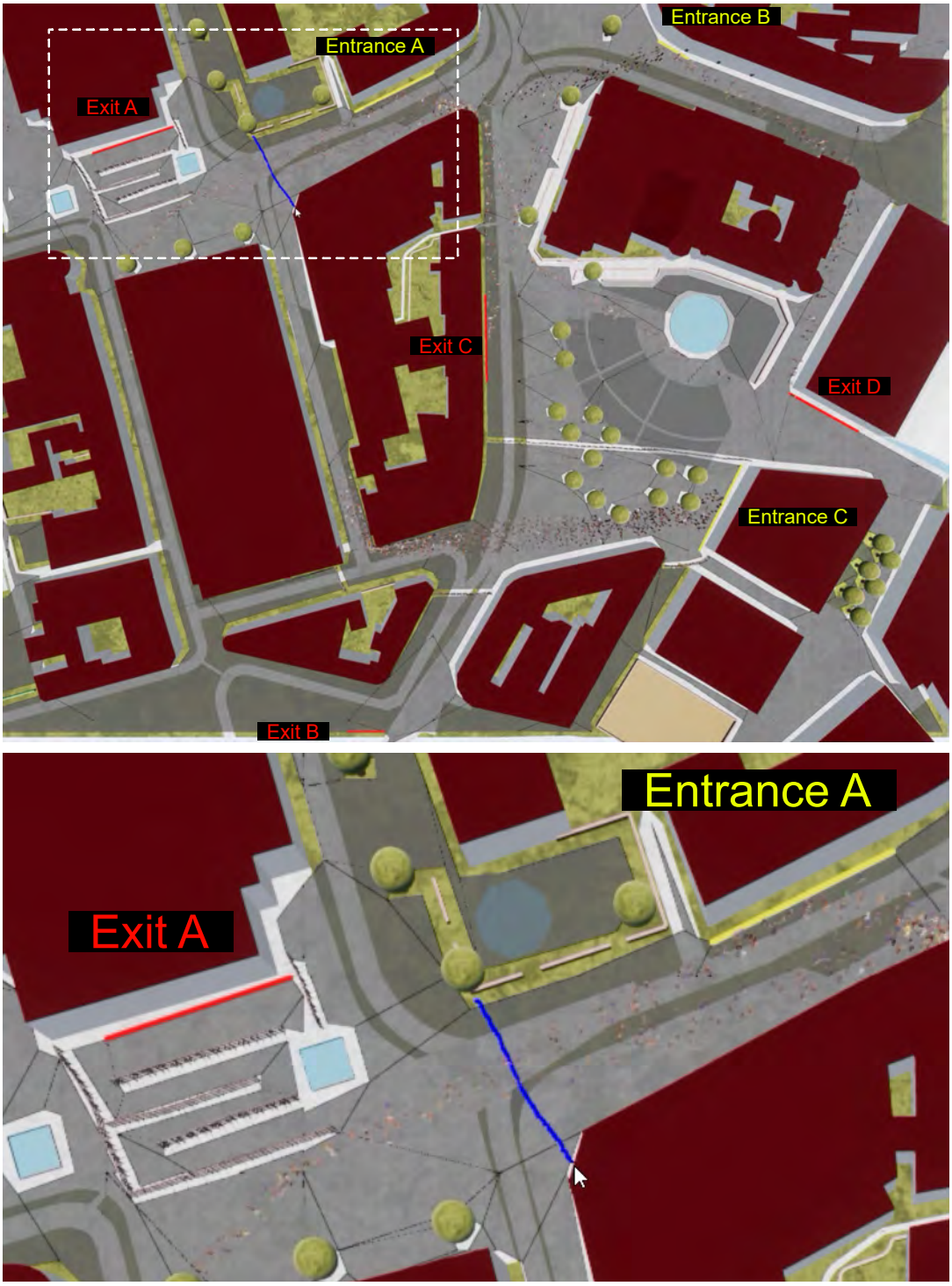
(b) (cont.)

Figure 4.8 (cont.): Simulation frames of a crowd being marshalled from Entrance C to Exit A in a city centre. (b) User continues to create obstacles to define the route to be followed by the crowd. The highlighted area in the environment is enlarged to better show off the barriers and the crowd.



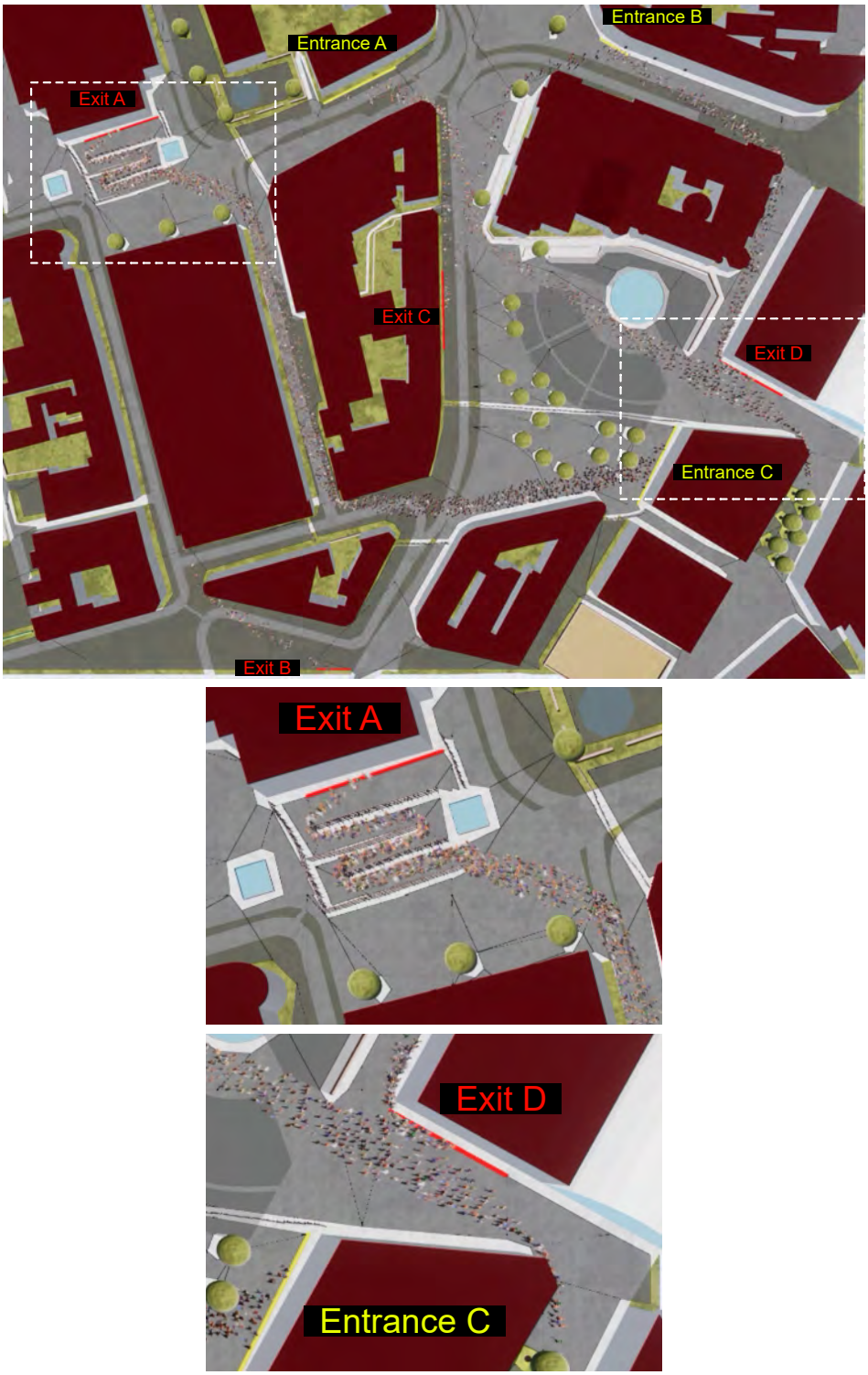
(c) (cont.)

Figure 4.8 (cont.): Simulation frames of a crowd being marshalled from Entrance C to Exit A in a city centre. (c) Pedestrians moving from Entrance A to Exit B recalculate their shortest path, walking around the buildings on the left. The highlighted area in the environment is enlarged to better show off the Entrance A crowd and the new path.



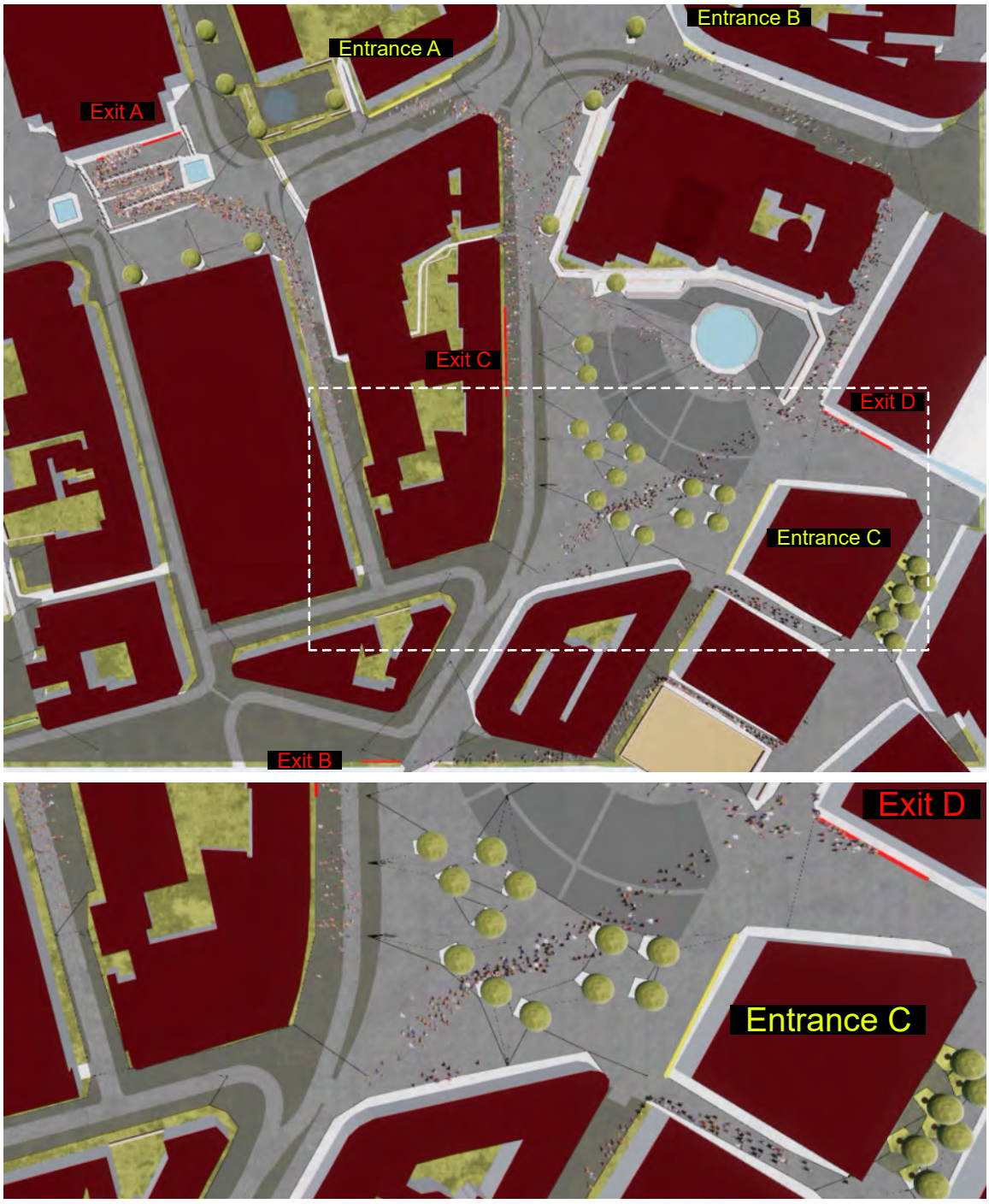
(d) (cont.)

Figure 4.8 (cont.): Simulation frames of a crowd being marshalled from Entrance C to Exit A in a city centre. (d) The new route followed by A-to-B pedestrians is blocked with a barrier before the crowd arrives. The highlighted area in the environment is enlarged to better show off the new barrier to protect the marshalled crowd.



(e) (cont.)

Figure 4.8 (cont.): Simulation frames of a crowd being marshalled from Entrance C to Exit A in a city centre. (e) The first pedestrians of the crowd arrive at Exit A. Highlighted areas in the environment are enlarged to better show off the crowd movement.



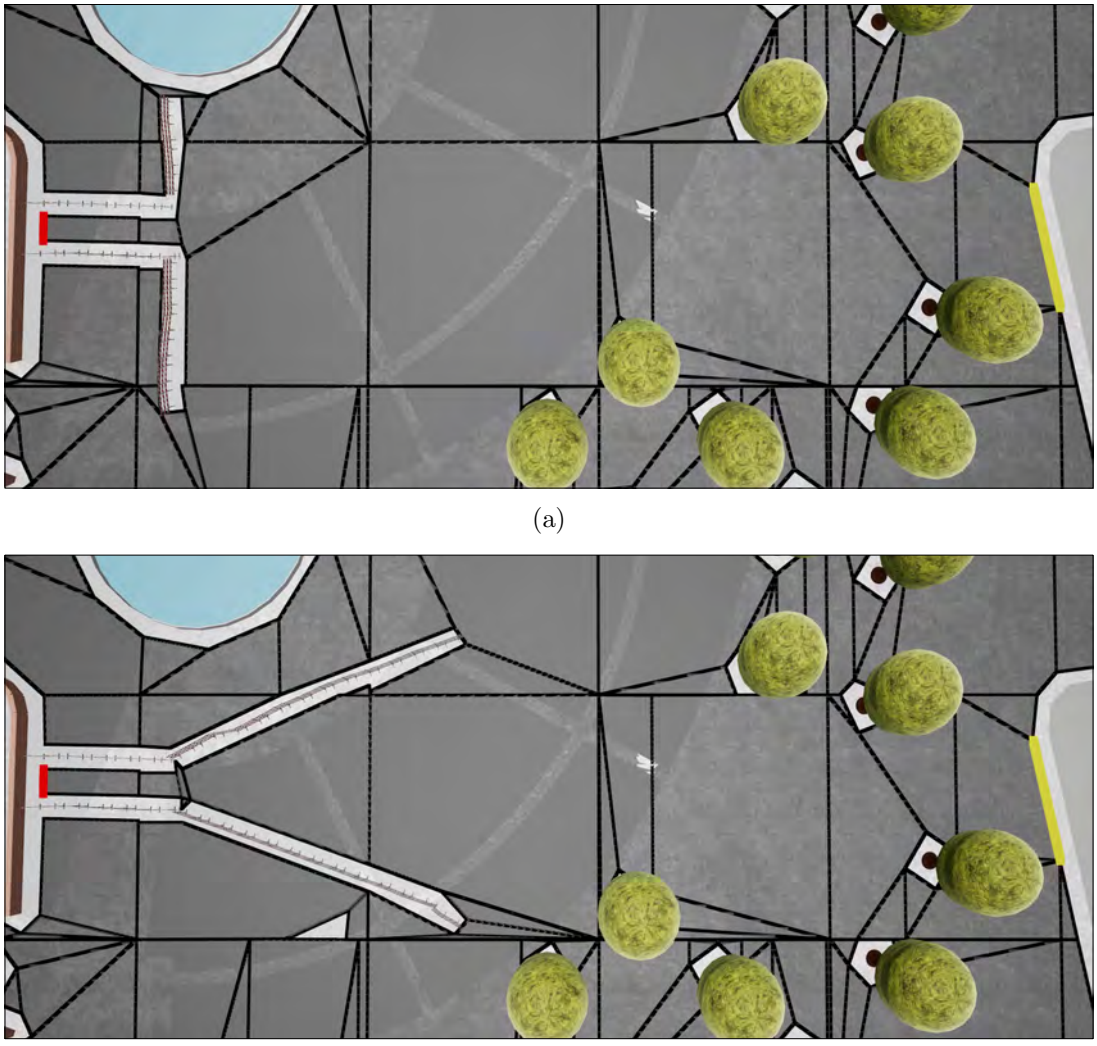
(f) (cont.)

Figure 4.8 (cont.): Simulation frames of a crowd being marshalled from Entrance C to Exit A in a city centre. (f) Barriers begin to be deleted as the crowd walks past certain regions. Pedestrians who are not part of the crowd can then continue on their preferred path. The highlighted area in the environment is enlarged to better show off the crowd reforming their path.

4.4 Additional system features

The system includes additional options that are not sketch-based but facilitate setting up simulations and obtain information of the crowd. First, users can save and load configuration files with all the elements created, including storyboards, timeline events and groups. This option means the same simulation can be run multiple times. Second, a script option is implemented to automatically run saved configuration files with specific duration and speed. This function means the script can be left running overnight or for days. Third, the resulting simulation can be saved. The system saves statistics such as the number of agents, exit rate, agent lifetime, position and velocity each frame. Fourth, more than one saved simulation can be replayed at the same time to compare the results. Last, the additional saved data such as population can be plotted to easily visualise the difference between simulations.

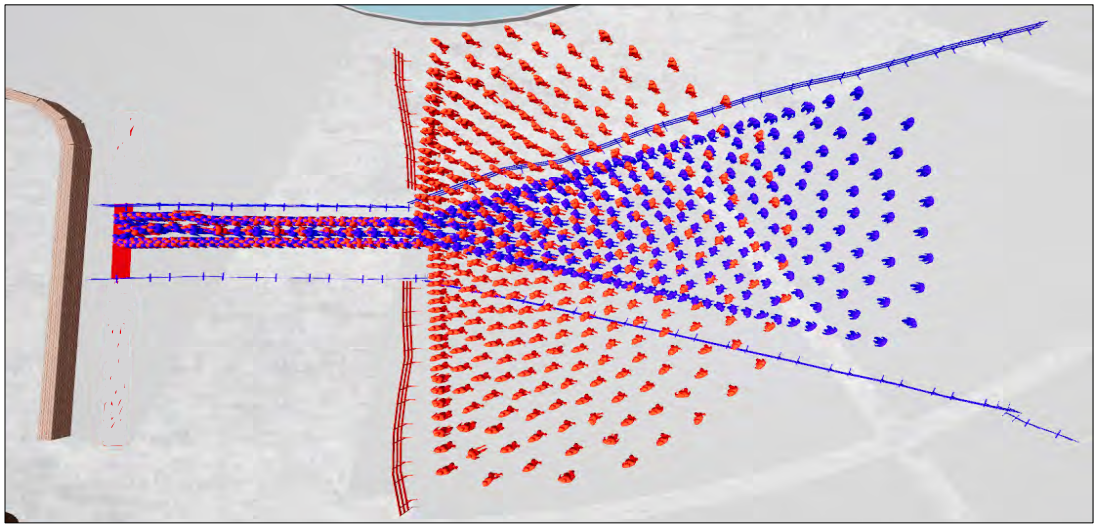
An example of these features is shown in Figures 4.9 and 4.10. Two evacuation scenarios were saved and run using the script option. These scenarios are rather simple and are used only to illustrate some of the system features. More reliable evacuation simulations require more complex agent and environment models. Figure 4.9a shows the first scenario where agents try to exit the simulation through the red exit. A narrow corridor, formed by barriers, creates congestion slowing down the evacuation. Figure 4.9b shows a variation of the same scenario. A funnel is created to enter the corridor. The idea is to save both simulations and replay them at the same time to observe the different behaviours. Figure 4.10a shows a frame of both simulations running. Pedestrians and barriers in red correspond to the first scenario, whereas the blue colour represents the second scene. The crowd without the funnel corridor creates a semicircle around the entrance of the corridor, while the blue crowd enter the corridor in a more organised manner. Besides observing the simulation, it is useful to plot the saved data to interpret it. The example scenario evacuates a crowd of 900 people. Figure 4.10b plots the number of agents over time for both scenarios. Once again, the red line corresponds to the narrow corridor with no funnel entrance. The graph shows that the second scenario evacuated faster (430s) than the first one (478s).



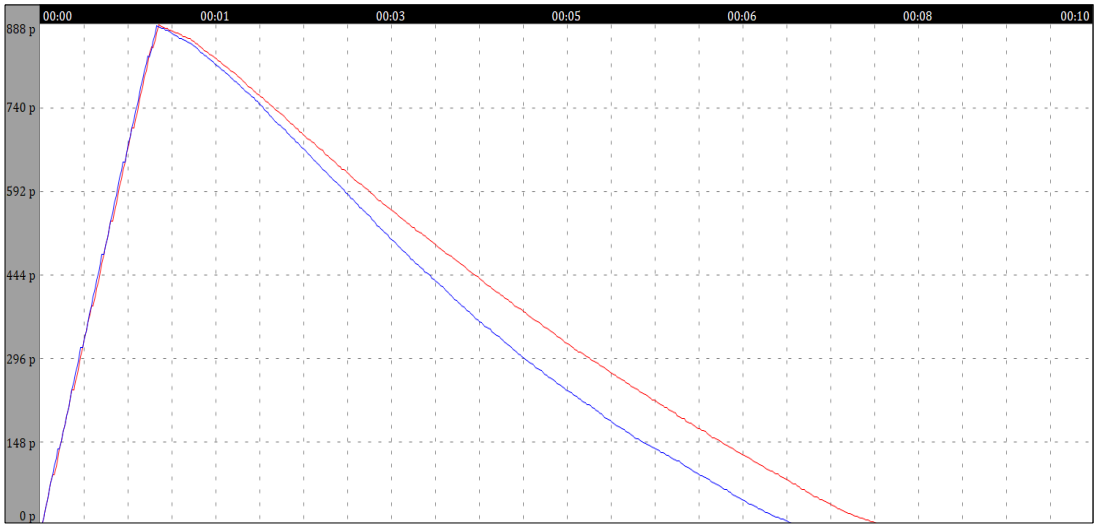
(a)

(b)

Figure 4.9: Evacuation scenario. Crowd moves from the yellow entrance to the red exit. (a) A narrow corridor in the exit slows down the pedestrians. (b) A funnel entrance to the corridor is created to improve pedestrian flow.



(a)



(b)

Figure 4.10: (a) Scenarios shown in Figure 4.9 are replayed at the same time. Red pedestrians correspond to the first scenario with just the corridor. Blue pedestrians represent the funnel corridor scenario. Different behaviour can be observed when overlapping both simulations. The red crowd forms a semicircle trying to enter the corridor. while the blue agents access the corridor in a more organised way. (b) Plot showing the population over time of both scenarios. The funnel corridor (blue) evacuates the crowd faster.

4.5 Controlling groups with sketching

So far, this research has only focused on sketching to modify the environment. The purpose of this feature is to give user fine-grained control over the crowd and to directly influence its behaviour by sketching. The idea is to select a group of agents by drawing a shape surrounding them and create a sketch-based storyboard. This storyboard is a sequence of multiple linked actions. These actions include following the exact path sketched by the user and following behaviours such as wait and evacuate. Control groups are included in the timeline to select the agents that are within the group area and follow the sketched storyboard at the time specified in the event. Selecting a subgroup [4] and sketching a path to follow [129, 46, 66, 4] have been done in previous work [129]. However, the idea of sketching a storyboard and linking events to define the pedestrians' journey has not been implemented. Hughes et al. [66] suggested a similar idea as future work, but no follow-up work has been found.

Group agents closely following a path would create unrealistic behaviour since they would form a line. The formation of the group must be maintained to create a plausible simulation. A solution to this issue is to obtain the centroid of the user-sketched shape. This centroid follows the sampled points of the path and the agents keep their relative position to the centre of the group. More complicated solutions that better maintain the formation could be used. Figures 4.13b and 4.13d show a group following a path while maintaining their distance. Groups need to store their centroid, the sequence of actions and the points of the sampled paths among other variables. The structure used to represent groups is shown in the code snippet below.

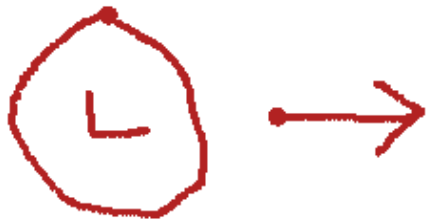
```
struct Group
{
  int noGroups;
  glm::vec3 *centre;
  int *noAgents;
  int **agents;
  int *noActions;
  int **story;
  int **noPoints;
  glm::vec3 ***points;
};
```

Behaviours are defined by sketching predefined symbols. A multi-stroke recogniser [6] is used to get a score and to determine if the user entered a valid behaviour. The recogniser compares template strokes against new sketch inputs using simple geometry and trigonometry. This tool is position and scale insensitive since it only focuses on the shape of the strokes. The score is based on the distance between the template and candidate points. The algorithm to compute the score is briefly described as follows:

- Generate single stroke permutations for the template shape.
- Combine input strokes into one
- Resample all the strokes into n points
- Rotate the strokes based on the starting point and the centroid

- Scale and translate the strokes to the origin
- Calculate start unit vector from starting point to the i_{th} point
- Obtain the minimum distance between star unit vector of the permutations and candidate strokes to get the score

Two example behaviours have been implemented in the system: wait and evacuate. Their symbols are shown in Figure 4.11. The order and the direction of the strokes are taken into account during the recognition process. A clock symbol represents the wait behaviour, and agents wait inside the circle of the symbol for a predefined amount of time before continuing with the storyboard. The symbol for evacuating is an arrow. When agents evacuate, they ignore their storyboard and their exit assigned and run to the nearest exit and leave the simulation.

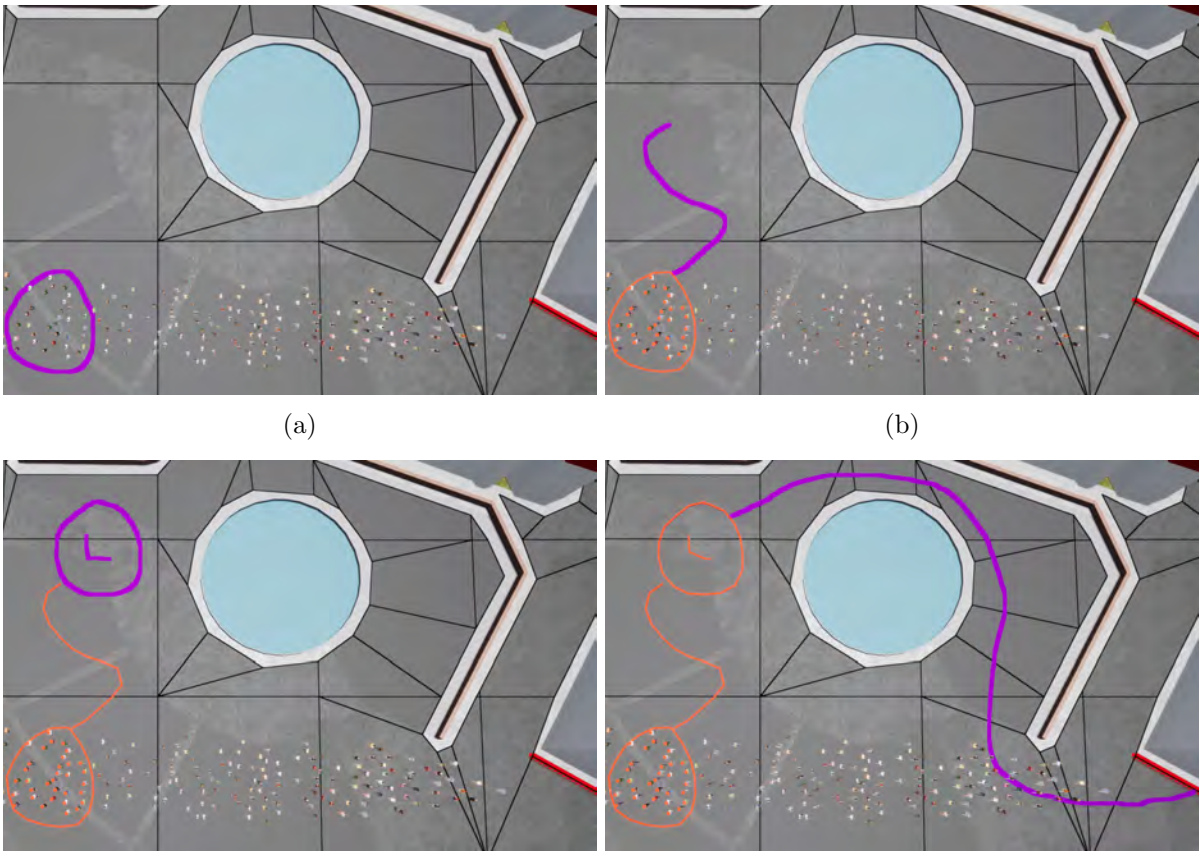


(a) (b)

Figure 4.11: (a) Wait symbol. (b) Evacuate symbol.

Paths and behaviours are linked to create the group storyboard. Agents immediately follow the storyboard as soon as it is created. Once the group storyboard is completed, pedestrians continue with their original journey to the exit. Figure 4.12 shows how to select a group of pedestrians and sketch a storyboard. In Figure 4.12a, a group is selected by sketching a circle around the agents. Pedestrians in the group are highlighted with an indicator above them. Then, the user defines a path (purple line) in Figure 4.12b. Current active lines are displayed in purple and parts of the storyboard that have been accepted are shown in orange. To specify an action or behaviour, the user has to sketch a predefined symbol. In Figure 4.12c, the wait behaviour is added to the storyboard by sketching a clock symbol. The group waits inside the area for a fixed amount of time. Lastly, the user defines the path towards the exit (Figure 4.12d). Figure 4.13 shows the group following the storyboard. In (a) the group of agents and the final storyboard are shown. Pedestrians follow the first part of the path (b), then wait inside the area defined by the wait behaviour (c) to finally take the path to the exit (d).

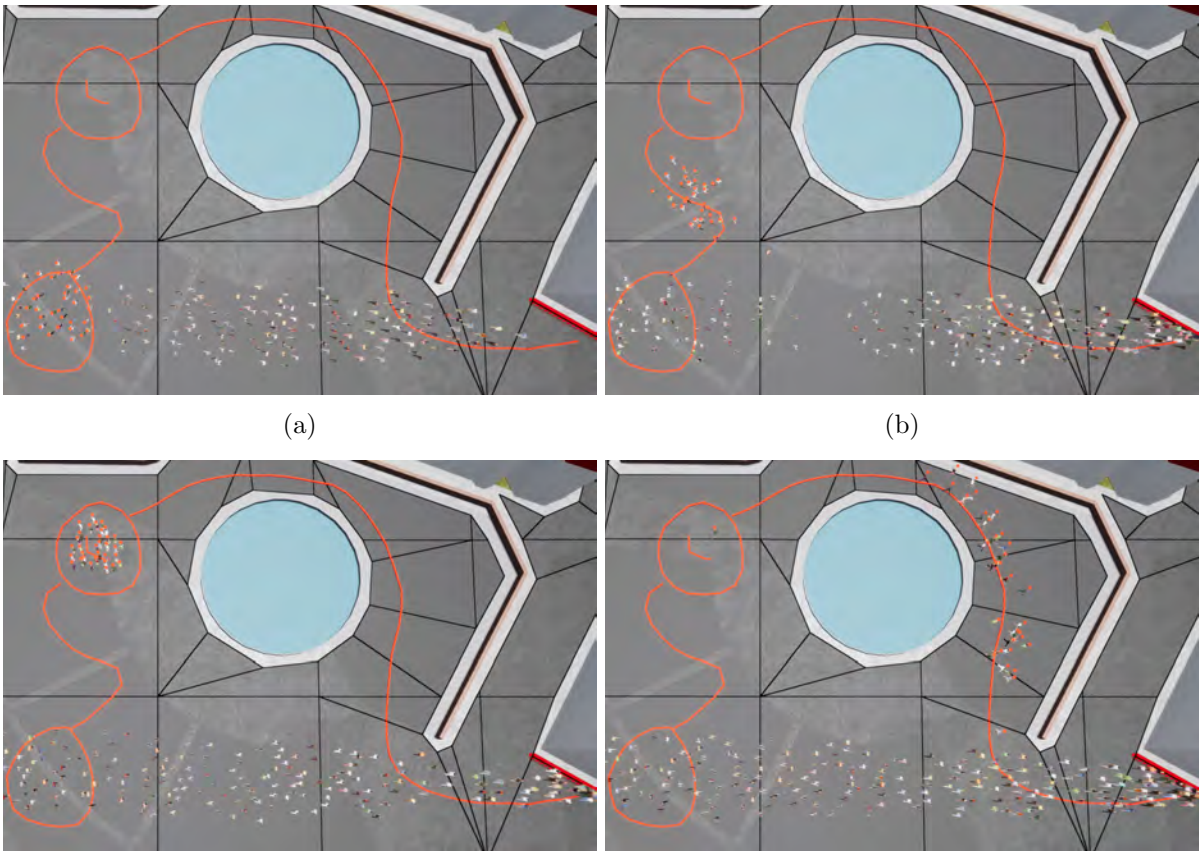
Sketching a new path with this approach overrides any previous path obtained from the navmesh. This new path may pose a problem when a barrier is added to block the group path. The solution is to check if any of the path segments are affected by the barriers. Since the path consists of connected lines, only the path between the starting and endpoint of the affected lines is recalculated. Figure 4.14 illustrates this process. Figure 4.14a shows the selection of a group with a sketched path. A barrier blocking this path is created in Figure 4.14b, and the affected segment of the path is replaced by the shortest path (obtained from the navmesh) between the start and endpoint of the original line segment.



(a) (b)

(c) (d)

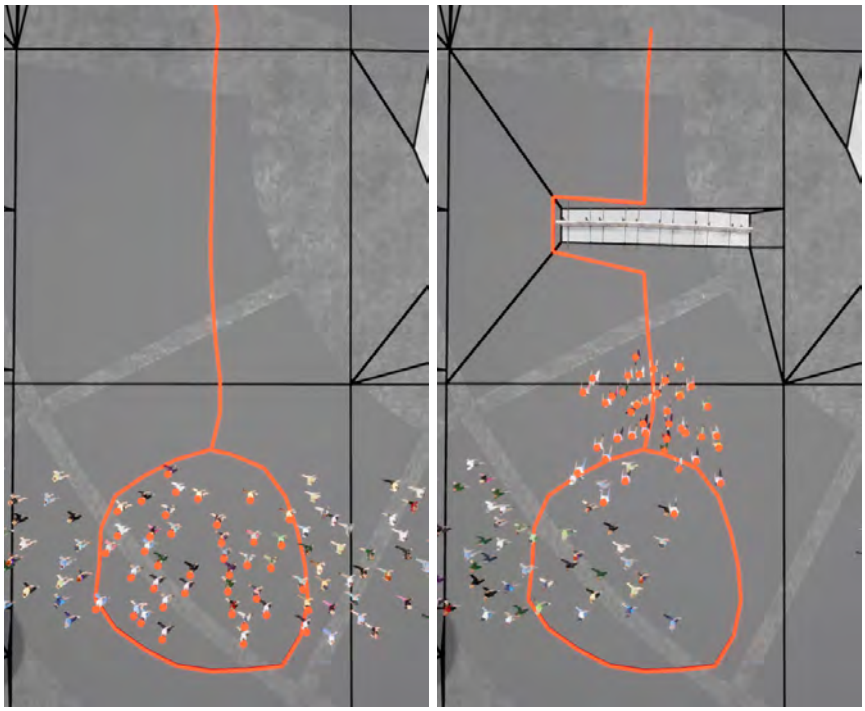
Figure 4.12: Group selection and creation of a storyboard. (a) A group of agents is selected by sketching a circle around them. (b) The user defines a path (purple line) as the initial part of the storyboard. (c) A waiting behaviour is specified by sketching a clock symbol. (d) The path to the exit is created.



(a) (b)

(c) (d)

Figure 4.13: Group of agents following a storyboard. This storyboard consists of two paths and a waiting behaviour. (a) The selected group is highlighted with small indicators on top of them. (b) The group following the initial path. (c) The selected agents are waiting inside the area defined by the symbol. (d) Finally, the group walks towards the exit.



(a) (b)

Figure 4.14: (a) Group selected and path defined by the user. (b) A barrier is added to block the crowd, and the sketched path is recalculated to go around the obstacle.

4.6 Dynamic Environment Knowledge

Modifying the environment in real-time presents new challenges. So far in this work, the environment changes made by the user are immediately mapped to a single navmesh (Section 3.3). This mesh is used to guide agents through the environment. Therefore, pedestrians react instantly to user sketches. This behaviour may be undesired since the crowd recalculates its route based on every obstacle, including distant barriers which are not visible to the crowd. Challenger et al. [22] noted that current crowd simulations do not consider that pedestrians are unlikely to possess complete knowledge of the environment. This issue is more relevant for dynamic environments and raises the question: when should agents react to dynamic changes? Previous work has proposed solutions to represent dynamic environments using constrained Delaunay triangulations [83], adaptive roadmaps [167] and navmeshes [181]. However, these structures are immediately updated after dynamic obstacles are introduced to the environment, and there is no discussion on when the crowd reacts to the new environment.

Some barriers partially block paths and could be represented without modifying the navmesh. These obstacles could be modelled as static agents with repelling forces to push pedestrians away. In this manner, agents would avoid the obstacle and retake their original path without making any changes to the environment. However, not every barrier can be avoided using local forces to steer the agents. In these cases, it is necessary to update the environment and calculate a new path to guide the crowd. This research work focuses on updating a navmesh in real-time using sketching, so every barrier updates the underlying navigation structure.

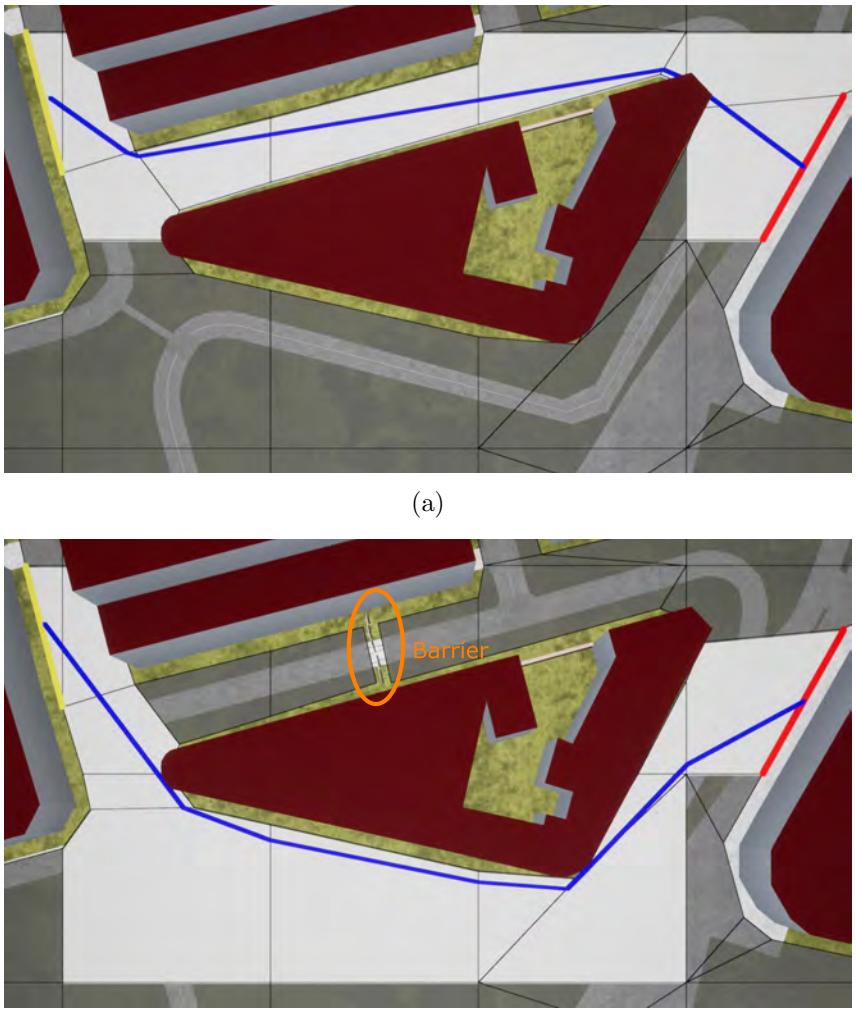
This section tries to answer the previously stated question by implementing possible solutions while considering their performance implications to find the most suitable alternative and produce a more realistic simulation. Two main types of environment knowledge are identified and discussed in the following sections: immediate and dynamic.

4.6.1 Immediate knowledge

The first case is where pedestrians gain knowledge of any environment modification immediately after the user input. The work described in Chapter 3 implements this approach. Updating the agent knowledge in this manner might result in unrealistic behaviour since agents that are located far away from the modified area change their path without seeing the obstacle. Furthermore, new pedestrians spawn with complete knowledge of the new environment, thus following the recalculated shortest path. This behaviour would be plausible in the unlikely scenario where all the pedestrians are notified in real-time (i.e. by a news item on their mobile phone or via the radio) of the dynamic updates.

To illustrate this behaviour, Figure 4.15 shows a part of the environment with an entrance on the left (yellow edge) and an exit on the right (red edge). The shortest path, shown in Figure 4.15a, goes directly from the starting point to the goal. A barrier is added in Figure 4.15b and the shortest path is recalculated now going around the building. The new obstacle is not visible from the entrance; nevertheless, pedestrians immediately adjust their route when the barrier is sketched.

Four frames of a simulation running in this part of the environment are shown in Figure



(a)

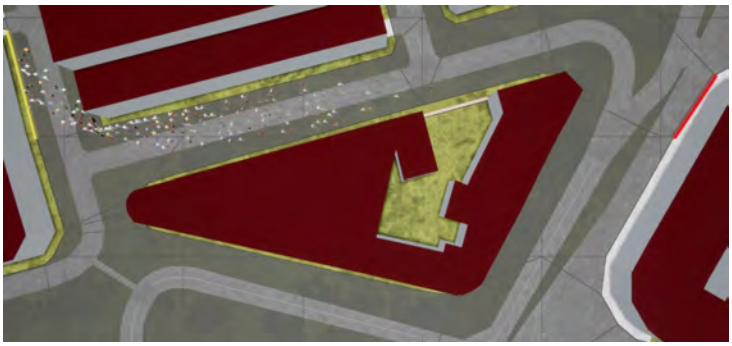
(b)

Figure 4.15: (a) Shortest path in blue from the entrance (yellow) to the exit (red). (b) A barrier has been added and the shortest path is recalculated.

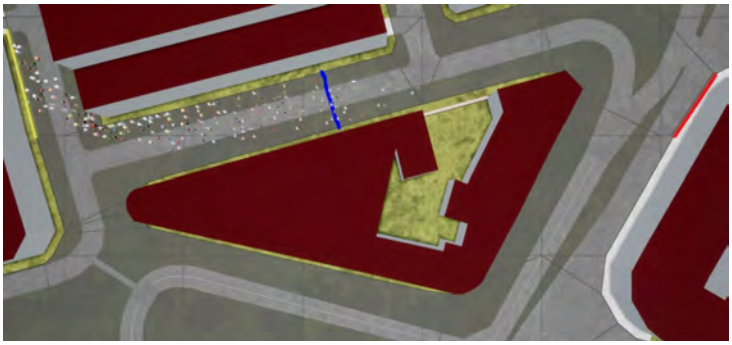
4.16. Frame (a) shows the crowd following the shortest path in Figure 4.15a. A barrier is being sketched (blue line) in frame (b). The crowd instantly reacts to the new obstacle and adjusts its path in frame (c). New agents follow the recalculated route since they spawn with this knowledge, as shown in the last frame (d).

4.6.2 Dynamic knowledge

In this solution, agents react to the new environment information when they see it. This produces a more realistic behaviour as people adjust their path only when they become aware of an object blocking the way. One challenge of this approach is to determine when pedestrians "see" the new obstacle. An optimal solution would be to equip agents with a visual sensor, as in [130], where rays are cast from the agent's eyes to detect objects inside the visual range. However, a more straightforward approach based on distance is implemented. Agents "see" barriers when



(a)



(b)

(c)

(d)

Figure 4.16: (a) Pedestrians following the shortest to the exit. (b) User sketching a barrier. (c) Agents adjusting their path after the barrier is created. (d) Spawning pedestrians instantly follow the new path.

they are within a radius from the centre of the obstacle. This method does not ensure that agents physically see the environment modification; a building could be obstructing their field of view. However, it is realistic-enough for the purposes of this simulation. Figure 4.17 shows people adjusting their path when they are inside the radius of the barrier.

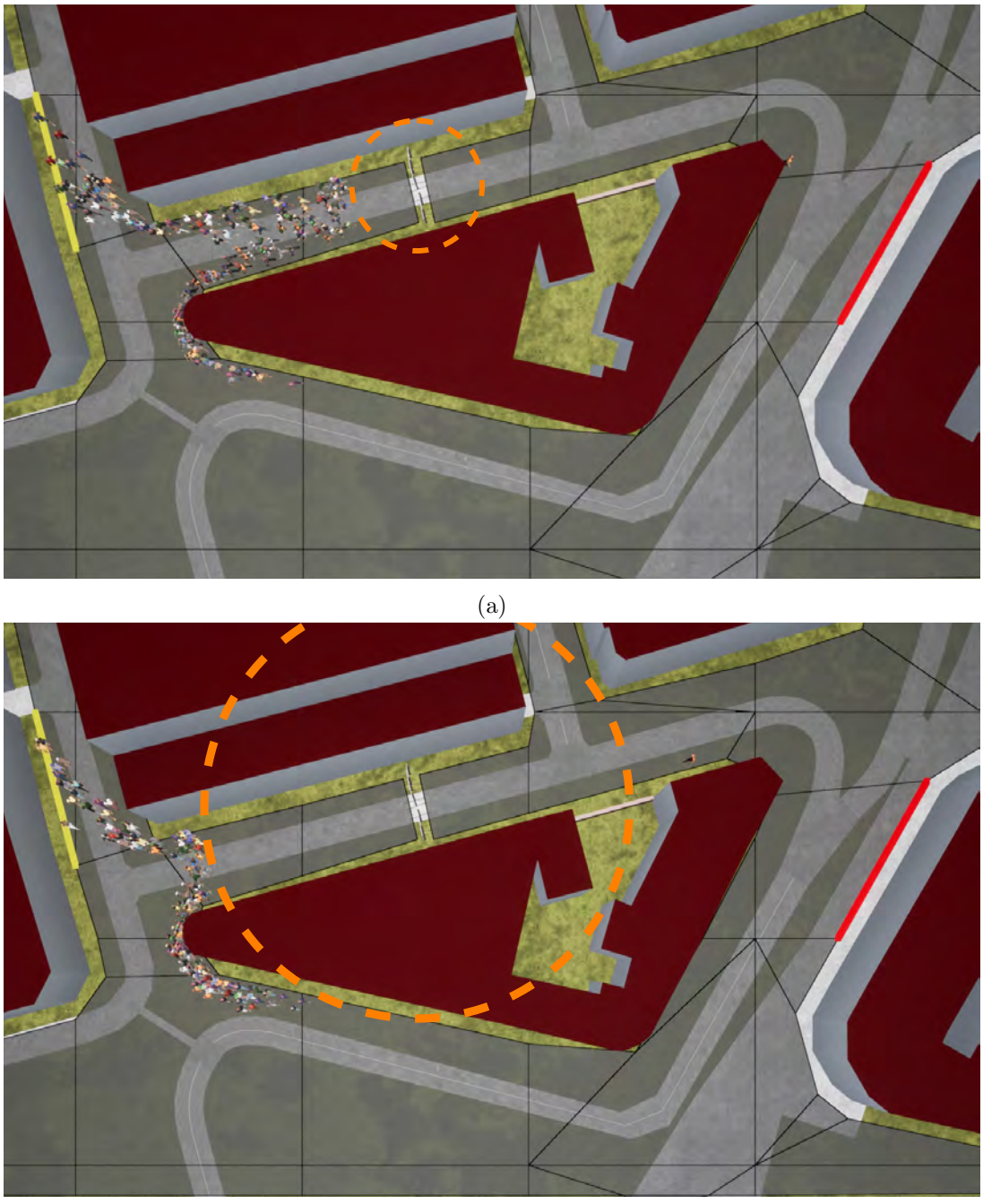
In this dynamic approach, each agent has a different knowledge of the environment depending on the obstacles they have discovered. Therefore more than one navmesh is required. A naive implementation would be to have every agent store its own navmesh and update it every time a barrier is found. However, this is unfeasible for large crowds since the amount of memory used increases linearly with the number of agents. Moreover, it is an inefficient solution since the same navmeshes are repeated thousands of times.

A more suitable alternative is to only create the minimum number of navmeshes needed to represent all the possible combinations of obstacles. For example, Figure 4.18 shows an environment with two barriers that requires four navmeshes: one to represent the empty environment, two navmeshes to represent each obstacle, and the last navmesh containing both barriers. The number of navmeshes is given by 2^n, where n is the number of obstacles. Having all these navmeshes allows pedestrians to only use the navmesh that includes the barriers they have found so far.

This solution is implemented limiting the number of barriers to five. This constraint is only applied to dynamic knowledge; there is no limit of obstacles in immediate knowledge. The maximum number of navmeshes, with the restriction of five barriers, is thirty-two since $2^5 = 32$. A disadvantage of the navmesh is the initial building time. Therefore it would take a considerable amount of time to create the initial navmesh thirty-two times. This problem is solved by creating only one navmesh which is iteratively updated to cover all the navmeshes. The tiles of the original navmesh are stored along with the tiles affected by each barrier. After a barrier is sketched, the original tiles are restored to start replacing them with the obstacle tiles in such a manner that every possible combination of barriers is covered. In every iteration, the shortest paths are calculated for that navmesh and stored in an array of structures.

The array of these structures and the position, radius and index of all the barriers are sent to FLAME GPU through the CPU shared memory segment. Pedestrians select a route to follow depending on the barriers they have discovered at any point of the simulation. The position of the agents is tested to determine if it lies within any barrier radius. Agents have a variable to indicate which navmesh they are using. The default navmesh is 0. Every time an agent finds a barrier, the bit of the *navmesh* variable corresponding to the barrier index is set to 1 with the following bitwise operation: $navmesh \mid= 1 << index$.

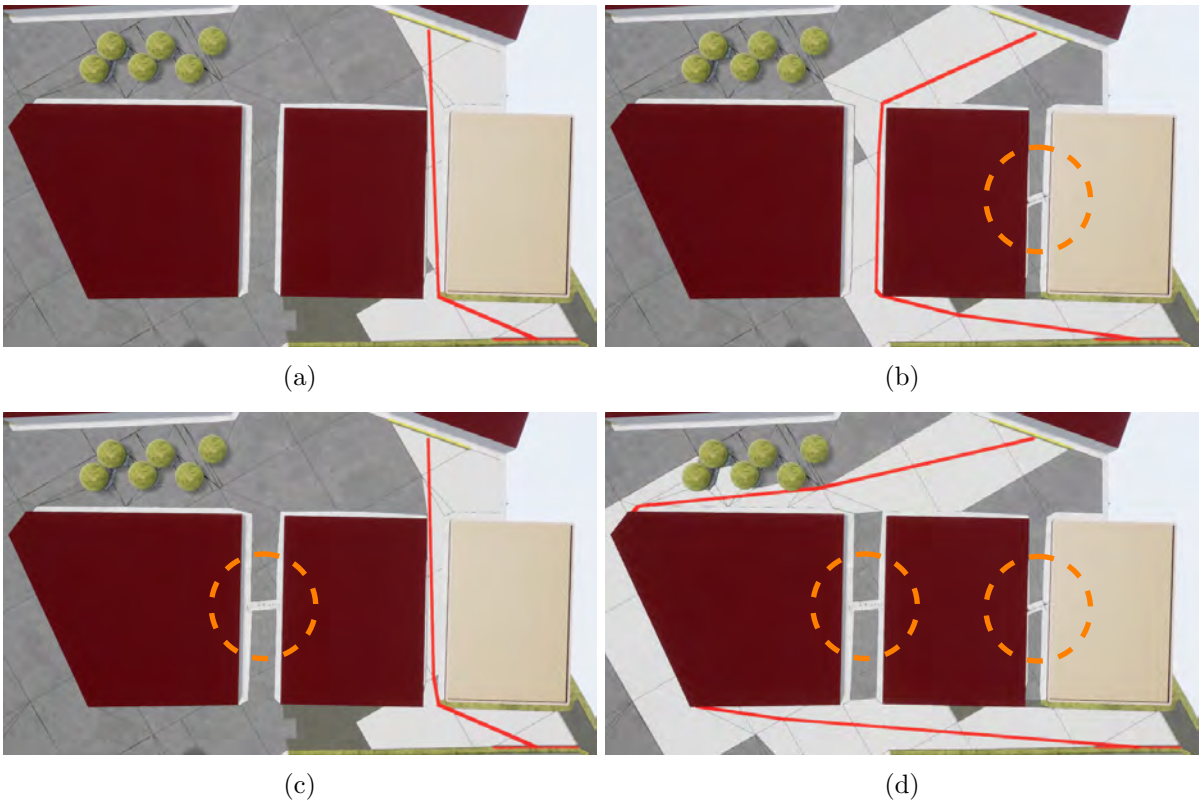
This variable is used to read the element at the navmesh position of the array of graphs read from Unreal. Using the example of Figure 4.18, the four graph structures are stored in an array. The first position of the array is the navmesh without obstacles, and the last one is the graph with both obstacles. The index of the barrier in Figure 4.18b is 0 and the obstacle in Figure 4.18c has an index of 1. Even though the environment has two barriers, agents follow the path shown in Figure 4.18a since the default value of *navmesh* is 0. After finding the first barrier, agents update their *navmesh* value to 1 and follow the graph in the second position in the array (Figure 4.18b). Similarly, when the second barrier is discovered, the new value of *navmesh* is



(a)

(b)

Figure 4.17: Pedestrians turning back after seeing the barrier (with different radius) blocking their path.



(a) (b)

(c) (d)

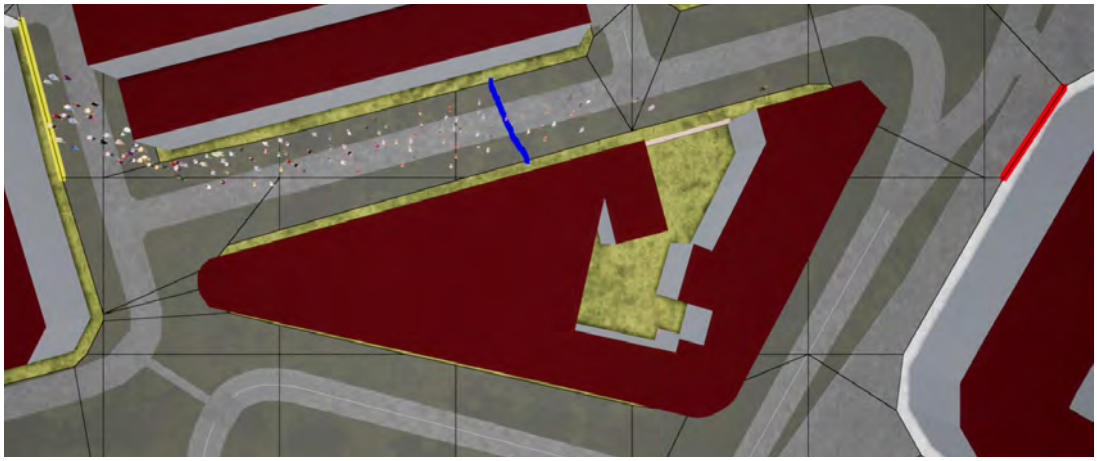
Figure 4.18: Environment showing the four possible combinations of two barriers and their shortest path. (a) Navmesh without barriers, (b) only the first obstacle is represented by the navmesh, (c) just the second barrier is sketched and (d) both obstacles included in the navmesh.

3 (path in Figure 4.18d) after setting the second bit to 1. This solution allows pedestrians to have individual knowledge of the environment.

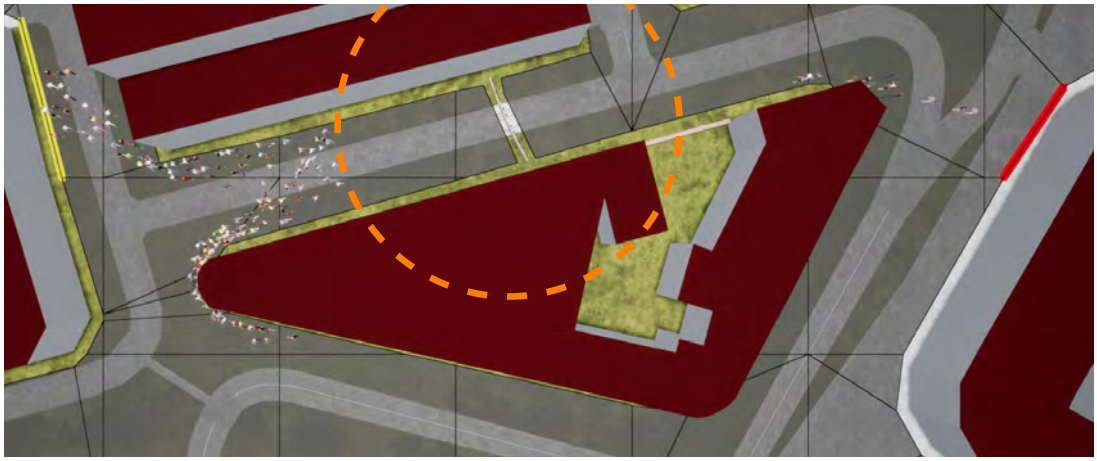
A further option is to mix this "vision" approach with time-based knowledge. In this type of knowledge, agents become aware of environment changes when they see them or after a short period when every pedestrian is notified. Currently, this time is fixed, and it is the same for all the obstacles. The Unreal simulation keeps track of the time and sends a message to FLAME GPU when the time is over. This notification sets the corresponding bit of the agents' navmesh variables to 0 if the barrier was deleted or to 1 if the obstacle was created. This behaviour is illustrated in the three frames of Figure 4.19. The crowd is moving from the yellow entrance to the red exit, and a barrier is sketched, blocking the followed path (4.19a). After the obstacle is created, pedestrians keep trying to follow the same way until they see the barrier in 4.19b. In 4.19c, the crowd has been notified of the new obstacle. Therefore they directly take the new path around the building.

4.6.3 Comparison

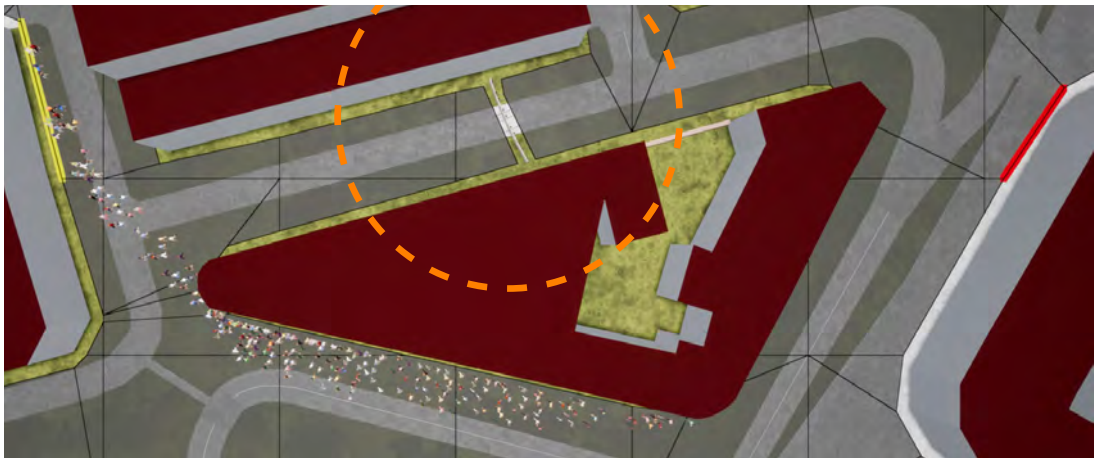
Figure 4.20 and Figure 4.21 show frames of the same simulation running three times each with a different knowledge approach. A video of the scenario is available at https://tinyurl.



(a)



(b)



(c)

Figure 4.19: (a) Barrier being sketched to block the crowd's path. (b) Agents reacting to the barrier when they 'see' the obstacle. (c) Pedestrians haven been notified of the existence of the new barrier and automatically follow the recalculated route.

com/yxmejzq5. The first column shows the immediate knowledge; the second shows the vision approach and lastly the vision plus time solution. In the simulations, agents move from the yellow entrance at the top to the red exit in the bottom building. Obstacles appear (Figure 4.20) and disappear (Figure 4.21) during the simulation and pedestrians adjust their path. Agents are represented by coloured circles depending on the navmesh they are using to clearly show the environment knowledge they possess at that time. During the simulation, three barriers are created; therefore, eight navmeshes are needed to cover all the possible barrier combinations. Red coloured agents are following the navmesh 0 (no barriers), blue pedestrians navmesh 1 (first barrier), white circles navmesh 3 (first and second barrier), and green agents navmesh 7 (all the barriers).

In the simulation with immediate knowledge (left column) agents are always in red since all the modification are made in one navmesh. In this column, it can be observed that pedestrians instantly change their route when a barrier is created. For example, in the second row of the Immediate column in Figure 4.20, after sketching the first barrier, all the agents are walking through the second corridor between the buildings on the left, whereas in the Vision and Vision+Time approaches only pedestrians that have seen the barrier (blue circles) are trying to walk around the building. In the third row of Figure 4.20 a second barrier has been sketched and the spawning pedestrians in the Vision + Time simulation have turned blue since they have been notified of the existence of the first barrier. Agents with only Vision knowledge are still following the original path until walking within the barrier radius. The last row shows the addition of a third barrier. Immediate knowledge agents keep adjusting their path as soon as the barrier spawns. The Vision column has agents of all different colours since they need to find the barriers one by one. In the Vision + Time approach, only two colours are observed. White agents have been informed about the first two barriers but not about the last one since it was recently created. Green agents know about all the barriers since they saw the new obstacle near the exit.

Figure 4.21 shows the same simulation but deleting the barriers in reverse order. In the first row, the third obstacle is removed. The immediate approach starts adjusting their path towards the exit to not go around the bottom building. For some agents that were already surrounding the building, the shortest path is the same that they were following when the barrier was still active. In the Vision column, pedestrians nearby the deleted barrier do not find it and keep walking straight to the exit. The Vision + Time approach shows all the agents in green (navmesh with all the barriers) since they have not been notified about the deletion of the last obstacle. The remaining rows show the same process of all the barriers. The key points to notice are: for immediate knowledge, agents instantly readjust their path, even by going back as shown in the second row between the two top buildings; for the Vision approach, agents do not go back since they cannot see that the barrier is gone; for the Vision + Time column, all the agents are changing colours (navmesh) at the same time since they cannot see any modification to the environment until they are notified.

A disadvantage of the dynamic implementation is that the path-finding time and memory used grow exponentially based on the number of barriers, as shown in Figure 4.22 and Figure 4.23.

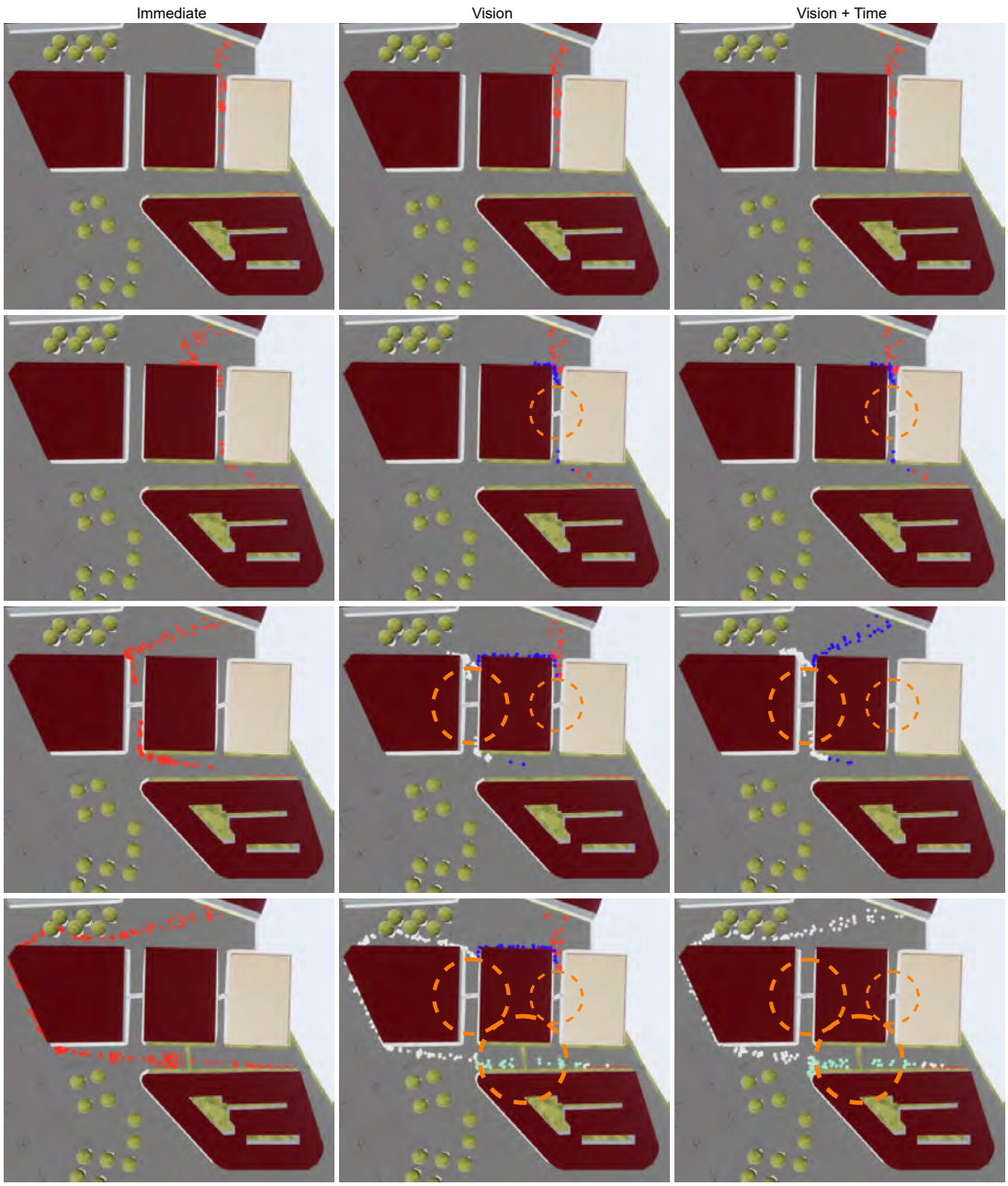


Figure 4.20: Frames of a simulation running three times with a different knowledge approach. Pedestrians move from the yellow entrance at the top to the red exit in the bottom building while a few barriers appear at different times to block their path. Agents are represented by coloured circles depending on the environment knowledge they possess at that time. The first column shows the immediate knowledge approach, where agents react instantly to any environment change. The second shows the vision approach - here, pedestrians become aware of obstacles when they "see" them by walking inside the radius (dashed orange circle) of the barriers. The last column illustrates the vision plus time solution - in this approach agents can know of the existence of an obstacle in two ways: by seeing it or by being notified a short time after the barrier was created. After being notified, pedestrians do not need to see the obstacle to changed their path.

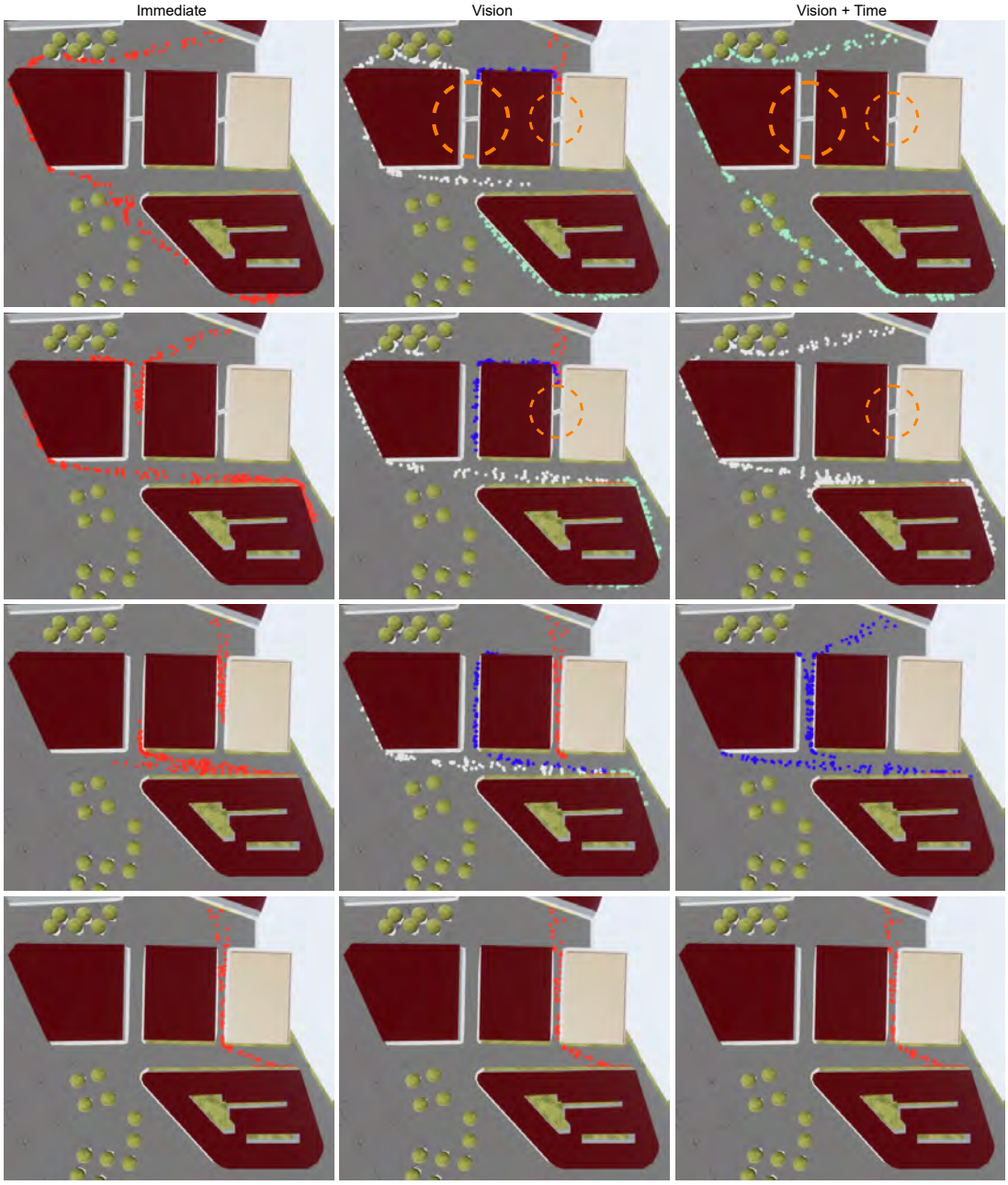


Figure 4.21: Frames of a simulation running three times with a different knowledge approach. Pedestrians move from the yellow entrance at the top to the red exit in the bottom building while the barriers disappear at different times to modify their path. Agents are represented by coloured circles depending on the environment knowledge they possess at that time. The first column shows the immediate knowledge where agents react instantly to any environment change. The second shows the vision approach, here pedestrians become aware of obstacles when they "see" them by walking inside the radius (dashed orange circle) of the barriers. Therefore, agents do not adjust their current path when barriers disappear. The last column illustrates the vision plus time solution, in this approach agents know about an environment change in two ways: by seeing it or by being notified a short time after the modification.

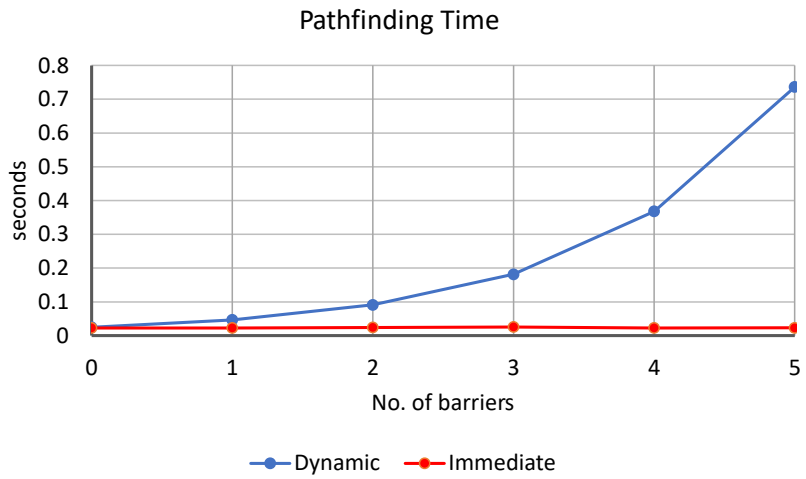


Figure 4.22: Comparison of the time taken in seconds to calculate the shortest path with multiple barriers by dynamic and immediate knowledge approaches.

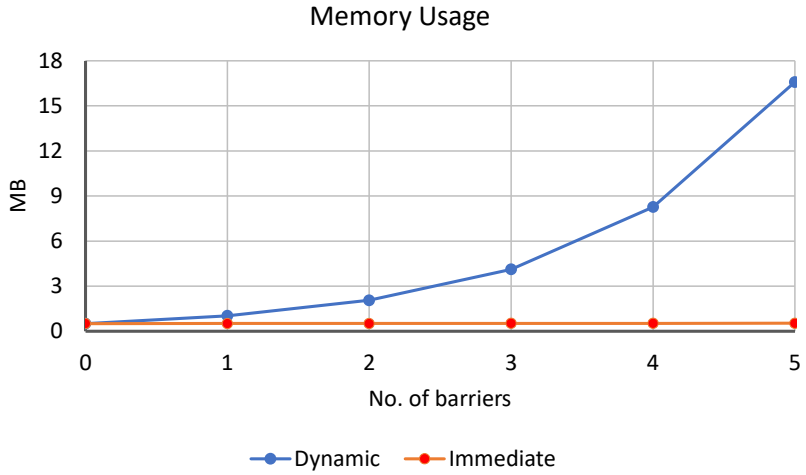


Figure 4.23: Comparison of the memory used by the navmeshes in the dynamic and immediate knowledge approaches with multiple barriers.

Immediate knowledge is an efficient solution since only one copy of the navmesh is required to handle all the obstacles sketched by users. Consequently, only one path calculation is performed per exit after each environment update. A drawback of this approach is that the resulting simulation is not sufficiently realistic for most scenarios. An alternative solution would be to give agents a dynamic knowledge of the environment where they react to new obstacles until they become aware of them.

4.7 Summary

This chapter has extended the sketch-based approach with other graphical interactions, such as clicking and drag and drop, to create more advanced features: storyboards, timeline interface and group control. Storyboards describe the journey of the crowd from an entrance to optional areas and finishing at an exit. The timeline interface creates events to facilitate the simulation of different times of the day. Last, the group control feature offers fine-grained control over the behaviour of pedestrians. This approach directly modifies agent parameters to affect only the group selected. The group control option allows selecting a group of agents and defining a sketch-based storyboard consisting of paths and behaviours. All these features can be used to create more complex scenarios, such as the ones described in Section 4.3. Overall, sketching provides an intuitive and quick way to control certain aspects of a simulation. However, complex features such as time-based events might be easier to implement by combining sketching with more advanced interfaces and graphical interactions.

An advantage of the sketching approach is that it can be used before running the simulation or in real-time. The tiled navmesh (Section 3.3) can be quickly updated since only the modified tiles are recalculated. This feature enables sketching elements in real-time to alter the environment and change agent behaviour. This interaction creates a new problem: when should the crowd become aware of the user inputs? Three kinds of environment knowledge were considered: immediate, vision and vision + time. The latter two produce more realistic behaviour, but increase the memory requirements for the environment.

Chapter 5

Comparing the sketching approach with MassMotion

Previous chapters have presented a sketch-based control system. The main idea of the system is to reduce the time for creating and tuning a crowd simulation. One way to determine if the sketching approach is easier to use and faster than traditional interfaces is to compare it against a commercial system. Another aspect is the simulation itself. An intuitive interface and quick control would not be useful if the simulation produced is not realistic enough. Therefore, not only the user interface has to be evaluated, but also the agent model. Nevertheless, it is important to emphasise that improving a simulation model and its realism is not the aim of the research.

This chapter compares the sketching approach with the validated commercial system MassMotion. A license of this piece of software was obtained from *Oasys* to explore the possibility of interacting with the simulation in real-time. The comparison of both systems is used to evaluate two aspects: the plausibility of the simulation and the user interface. Previous sketch-based systems have been evaluated against conventional interfaces [129, 4]. However, no formal evaluation method is used, only completion time and customised questionnaires. This chapter uses tools such as 'steersuite' [164], KLM [20] and SUS [18] to make the evaluation.

A crucial difference between both systems is that MassMotion does not allow real-time modifications of the environment. Therefore, a comparison of the agent behaviour when users interact with the simulation running is not possible. An attempt to simulate the addition of a barrier while the simulation is running was made using an API provided by MassMotion (more details in Appendix A). However, it is not possible to change the layout of the environment in real-time since MassMotion precalculates the path of the pedestrians based on a distance map generated during the compilation of the scene to be simulated. For this reason, only the setup of the simulation can be compared.

Section 5.1 describes how the plausibility of the simulation is evaluated using the benchmark suite 'steersuite'. Section 5.2 explains the methodology followed to assess the graphical user interface. A user study is required to compare the sketching approach with MassMotion. The results of the study are divided into three sections: KLM, efficiency and effectiveness, and SUS. Section 5.3 describes the scenarios used for the user study. Section 5.4 proposes and adapted

KLM version to include a new sketching operator and presents the results of the KLM analysis. Section 5.5 describes how efficiency and effectiveness are computed and compares the results of both systems. Section 5.6 explains the System Usability Scale (SUS) questionnaire and discusses its results. Lastly, Section 5.7 summarises the results of all the evaluation methods in this chapter.

5.1 Plausibility of the simulation

The focus of the research is not to improve the realism of the simulation. However, the resulting simulation should be realistic enough to evaluate the proposed graphical control interface. In this case, realism is considered as low-level individual motions of the agents. The simulation is compared with MassMotion. The same scenario (Figure 5.1) is created in both systems and run twice: with no barriers and with barriers (agents in MassMotion) spawning in the middle of the simulation. The benchmark suite "steersuite" [164] is used to obtain some agent metrics, such as time, speed, angular speed and energy, which can be used to compare both agent models. This tool is used since it provides a practical way to obtain data from a crowd simulation. Furthermore, the extracted metrics evaluate and compare steering algorithms objectively. *Steersuite* reads the position and orientation of the agents every frame to calculate the metrics. Table 5.1 shows the average value of the data obtained from all the agents after running five times the scenario without barriers.

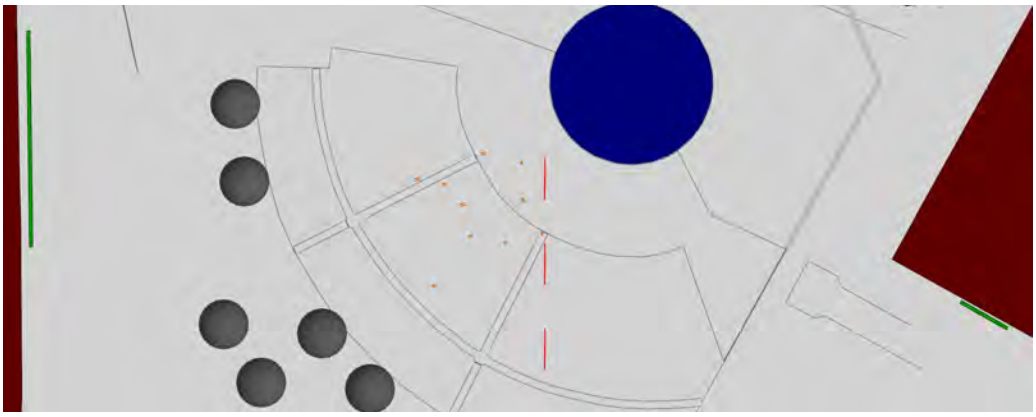


Figure 5.1: Scenario running in MassMotion. Pedestrian moves from the left entrance to the right exit. The three red barriers appear at a specific frame of the simulation.

The results are similar in most of the metrics. These suggest that there is no noticeable difference between both simulations. However, three measures showed a significant variance: total speed change, max angular speed and max degrees turned. The cause of the disparity is the implementation of the social agent model. MassMotion considers seven forces to calculate the movement of the agents, whereas the sketching approach only three. The high total speed value in the sketching approach may be caused by the acceleration of the agents when they enter the simulation and before exiting it. Nevertheless, these differences do not produce an evident simulation disparity and do not affect the real-time sketching process.

	MassMotion	Sketching	Difference
Time (s)	75.62	75.367	0.33%
Distance (m)	114.891	112.454	2.12%
Speed (m/s)	1.521	1.492	1.90%
Max speed (m/s)	1.618	1.572	2.84%
Total speed change (m/s)	2.523	15.108	498.81%
Degrees turned	163.453	123.389	24.51%
Max angular speed (deg/s)	27.641	45.905	66.07%
Max degrees turned	5.528	9.452	70.98%
Sum kinetic energy (J)	438.308	407.419	7.04%
Total integral kinetic energy (J)	175.323	168.245	4.03%

Table 5.1: Metrics average obtained with *Steersuite* after running the scenario without barriers five times.

	MassMotion	Sketching	Difference
Time (s)	76.3	77.197	1.17%
Distance (m)	115.347	114.388	0.83%
Speed (m/s)	1.513	1.481	2.11%
Max speed (m/s)	1.618	1.676	3.58%
Total speed change (m/s)	2.961	17.269	493.43%
Degrees turned	237.069	173.011	27.02%
Max angular speed (deg/s)	46.57	44.848	4.47%
Max degrees turned	9.314	9.346	0.34%
Sum kinetic energy (J)	438.005	412.251	5.87%
Total integral kinetic energy (J)	175.202	170.874	2.47%

Table 5.2: Metrics average obtained with *Steersuite* after running five times the scenario with barriers.

Table 5.2 shows the average measures obtained after running the scenario with the barriers five times. Comparing the numbers produced by MassMotion with and without obstacles, a significant difference in angular speed can be observed. This variation occurs when pedestrians enter the area of influence of the "barriers" and quickly adjust their path. In contrast, the sketching approach does not present a considerable change since agents react at the moment the environment modifications are done. This early reaction produces a more gradual movement adjustment. As mentioned in Section 4.6.1, immediate knowledge is used for the simulation since it requires less memory and it is faster than the other types of knowledge explored.

The results produced by 'steersuite' were mostly similar for both systems. A few metrics showed some variations due to the different agent model implementation used by each system. However, this difference did not have a visual impact on the simulation, thus it can be assumed that the simulation is realistic enough to continue with the evaluation of the user interface.

5.2 User Interface

The evaluation of the user interface is done by comparing MassMotion with the sketching interface. However, this can only be made for the simulation setup process (i.e. entrances/exits, obstacles) since MassMotion does not support real-time interaction. A user study is required

for a valid comparison. The study was ethically approved via the University of Sheffields Ethics Review Procedure. Eight participants with no prior knowledge of the crowd simulations systems were recruited. All the participants had experience using computers.

In this study, the eight participants were asked to create three simple scenarios in MassMotion and the desktop interface of the sketching approach. The scenarios were designed with three objectives in mind: simulate practical situations, include elements present in both systems, and gradually increase the complexity. Before the beginning of the experiment, participants were provided with an information sheet describing the purpose of the study, a consent form and an instructions sheet with the tasks needed to complete the scenarios. The experiments were divided into two sessions (one per piece of software) with a break of 15 minutes. At the start of each session, a general introduction to the system in question was given to users. Sessions were video recorded to count the number of operations (i.e. mouse clicks, keystrokes). The video focuses on hand movement and the screen as shown in Figure 5.2. At the end of each session, participants were asked to answer a small questionnaire. The idea of having a separate session per system is to avoid confusion and allow participants to focus only on one piece of software at a time. The drawback of this approach is that users are already familiar with the scenarios for the second session. Results could be affected if the first session is always the same system (Massmotion or sketching). For this reason, half of the participants started with MassMotion and the other half with the sketching system. All the documents given to the participants are included in Appendix B.

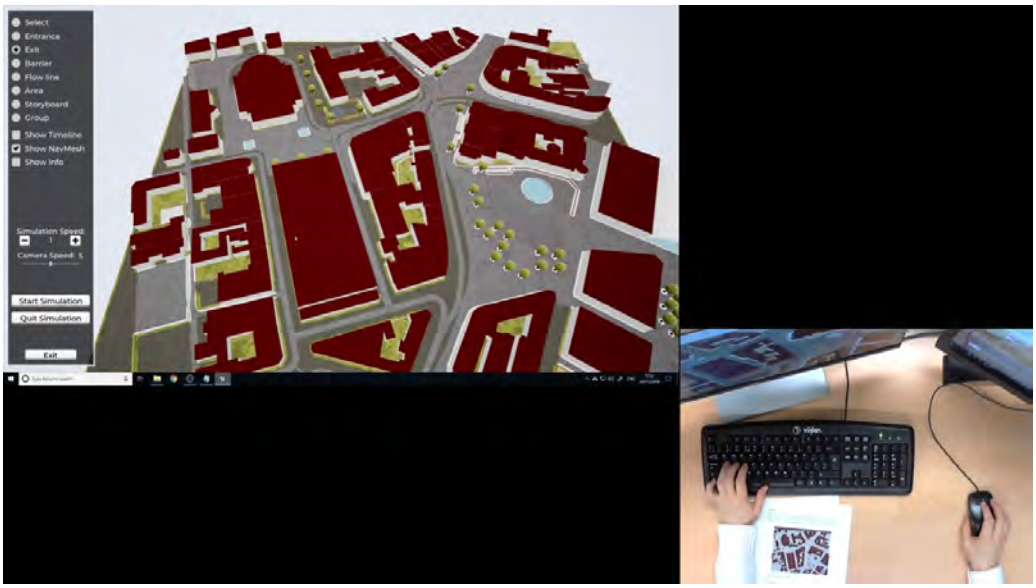


Figure 5.2: Frame of video recorded during the user study. The video focuses on the keyboard, mouse and the screen.

Two approaches were followed to evaluate the results of the user study. The first approach was based on KLM [20], which is a quantitative analysis tool used to evaluate interactive systems and predict user performance time. This model can be adapted to measure the number of operations required to complete a task and compare MassMotion and the sketching approach.

The second approach used is based on [183]. This work used multiple metrics to evaluate the use of tablets and laptops in common tasks. The criteria included effectiveness, efficiency and usability. The effectiveness is measured by counting the number of attempts needed to perform a task. Efficiency is determined by the time taken to complete an action. Usability is assessed with the help of the System Usability Scale questionnaire [18] (included in Appendix B), in which users have to answer ten questions using a 5-step Likert from"strongly disagree" to"strongly agree".

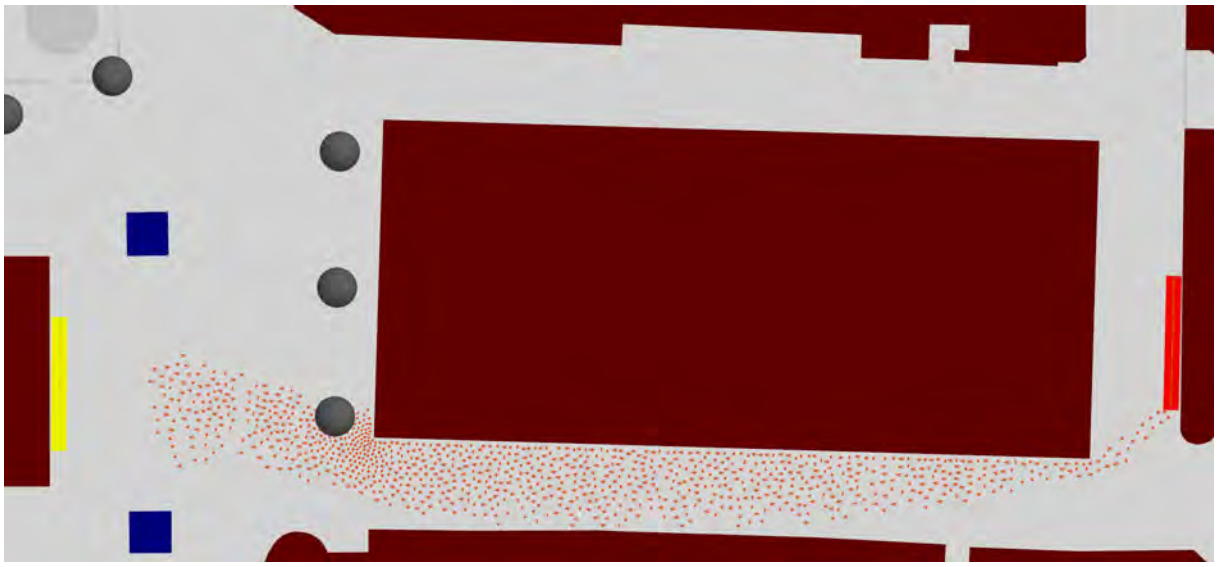
The results of the study are divided into three categories: KLM, effectiveness and efficiency, and user survey. The next section describes the scenarios included in the user study. Then each result category is presented in a separate section: KLM results in Section 5.4, effectiveness and efficiency in Section 5.5 and user survey in Section 5.6.

5.3 Scenarios

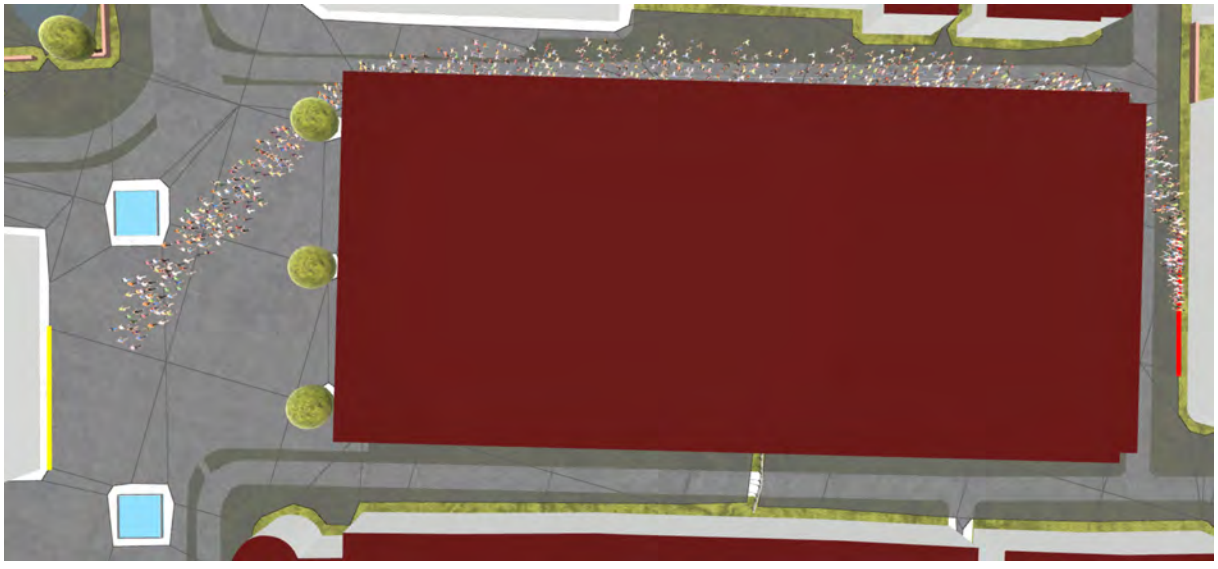
Three scenarios were included in the user study. Their complexity increased with every scenario. The scenarios were designed in this way to help participants familiarise themselves with the systems. In addition, complex scenarios could not be included due to time limitations. The first scenario is simple and serves as an introduction to the system. The second scenario divides the crowd into groups. Finally, the third scenario introduces the use of storyboards and areas.

Scenario 1, shown in Figure 5.3, initially consists of only one entrance and one exit. After running and observing the simulation, participants need to add a barrier to block the path followed by the crowd. The simulation is rerun with the new obstacle included. The purpose of this scenario is to start with a simple simulation and to show the steps needed to modify the environment. This scenario is divided into the following tasks:

- Create and rename the entrance
- Create and rename the exit
- Spawn 1,000 pedestrians at a 10 pedestrians/s rate
- Run simulation
- Stop the simulation when all agents exit the simulation
- Create the barrier
- Run simulation
- Stop the simulation when all agents exit the simulation



(a)

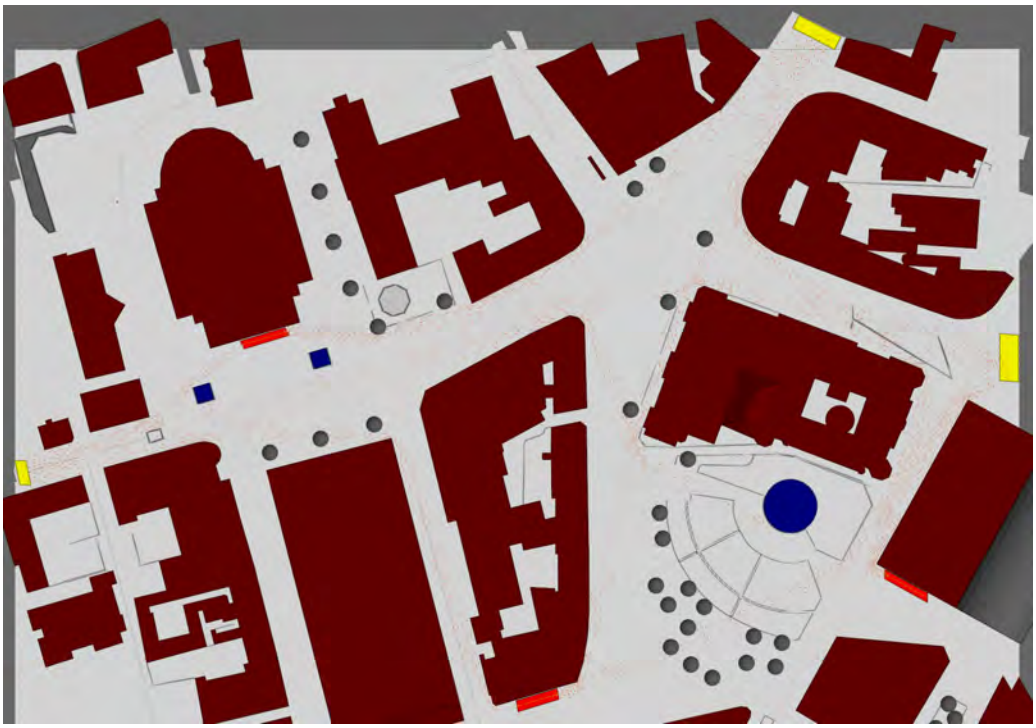


(b)

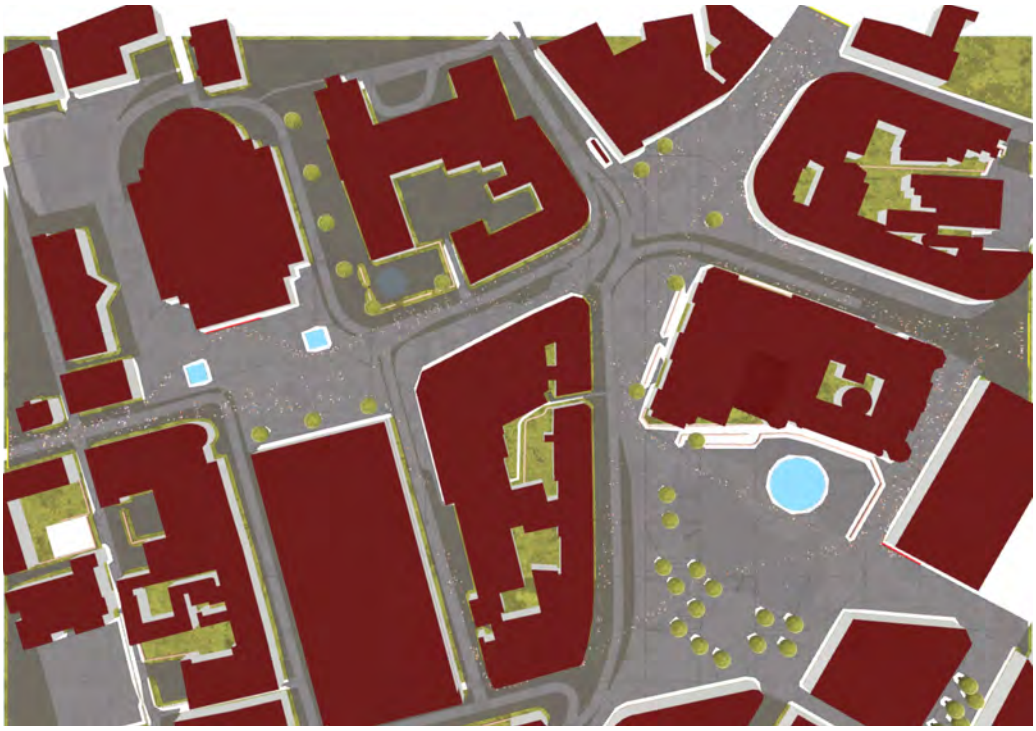
Figure 5.3: Scenario 1 running in (a) MassMotion and in (b) Unreal Engine. Pedestrians move from the yellow entrance to the red exit.

Scenario 2 has three entrances and three exits. Users are asked to equally divide the pedestrians spawning from each entrance into three groups. Every group should move towards a different exit. The scenario running is shown in Figure 5.4. The reason for the second scenario is to separate the crowd into groups following different exits. This scenario is divided into the following tasks:

- Create and rename the entrances
- Create and rename the exits
- Spawn 900 pedestrians per entrance at a 3 pedestrians/s rate
- For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit. For example, the groups from Entrance A would be 300 agents to Exit 1, 300 agents to Exit 2 and 300 agents to Exit 3.
- Run simulation
- Stop the simulation when all agents exit the simulation



(a)

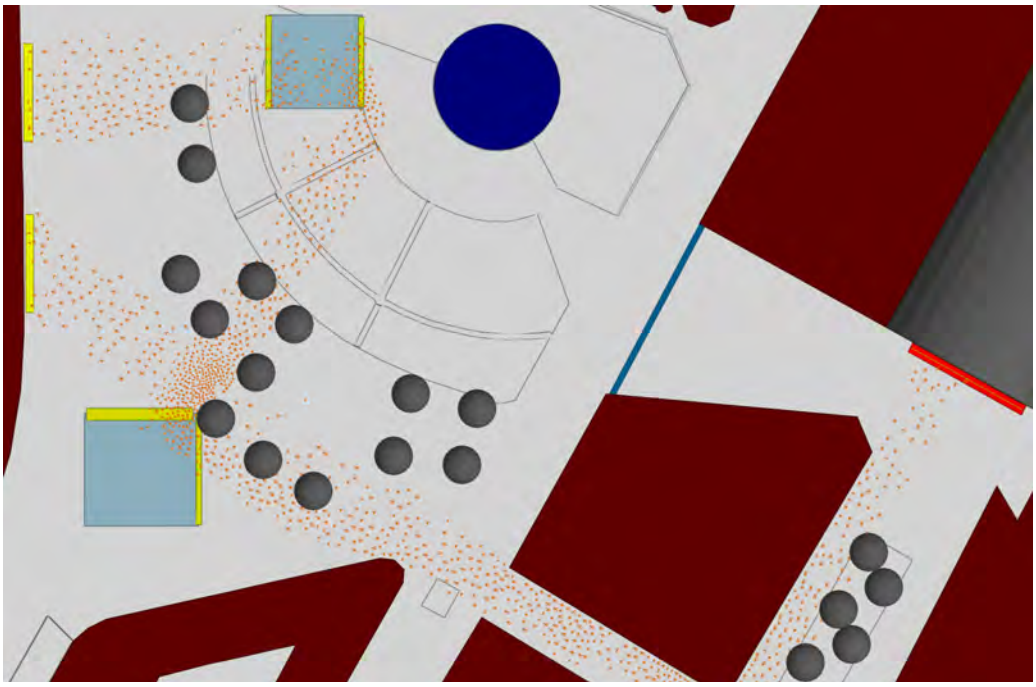


(b)

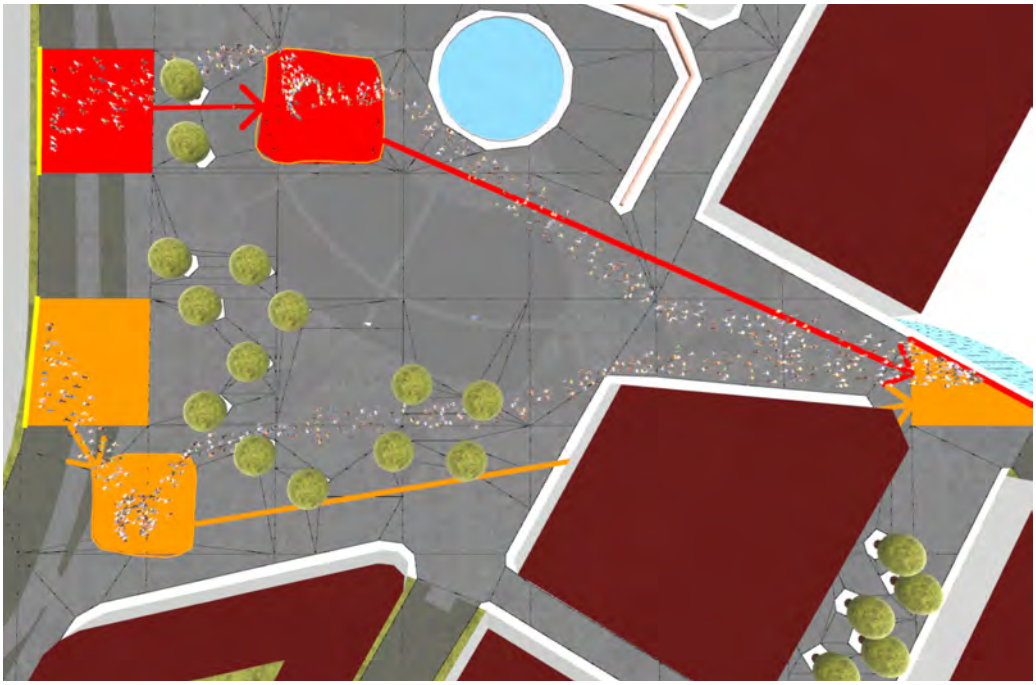
Figure 5.4: Scenario 2 running in (a) MassMotion and in (b) Unreal Engine. Pedestrians spawn from the three yellow entrances and move towards the three red exits.

Scenario 3 (Figure 5.5) includes two areas used as waypoints, where pedestrians briefly wait before continuing to their destination. The goal of the scenario is to specify the journey and actions of the crowd. This journey is modified after simulating for the first time. Similar to the first scenario, a barrier is added to block the direct path to the exit. This scenario is divided into the following tasks:

- Create and rename the entrances
- Create and rename the exit
- Create and rename the areas
- Spawn 900 pedestrians per entrance at a 3 pedestrians/s rate
- Make pedestrians from Entrance A visit Area 1 and wait 10 seconds inside then move to the exit
- Make pedestrians from Entrance B visit Area 2 and wait 10 seconds inside then move to the exit
- Run the simulation
- Stop the simulation when all agents exit the simulation
- Change the journey of Entrance A to visit and wait in both areas before going to the exit
- Change the journey of Entrance B to move directly to the exit
- Create the barrier
- Run the simulation
- Stop the simulation when all agents exit the simulation



(a)



(b)

Figure 5.5: Scenario 3 running in (a) MassMotion and in (b) Unreal Engine . Pedestrians visit the areas before continuing to their final destination.

5.4 KLM

The goal of this model is to predict the time needed to complete a task by counting the number of low-level operations necessary. This method is used in computer systems but has been adapted and extended to work with mobiles phones [63] and in-vehicle systems [155, 103]. KLM assumes that expert users perform the tasks being evaluated. Each low-level operation is represented by an operator which is given a time value to obtain the total amount of time required for the job. This study does not use KLM to predict task completion time but to get a metric to compare the sketching approach and MassMotion. The next section describes the operators used.

5.4.1 Operators

A modified KLM version is required to include tasks such as sketching or the use of the interface. Therefore, the operators can be divided into two groups: original and new/adapted. The original group of operators include:

- *Keystroke* (***K***). This is the most used operator and represents any keystroke or mouse click.
- *Pointing* (***P***). This operator represents the action of pointing to a target in the interface with the mouse.
- *Homing* (***H***). This operator is used when users move their hands between physical devices. A mouse and a keyboard are the only devices needed for the user study.
- *Response Time* (***R***). This operator represents the system response time. It is used when users have to wait for the system.

Two additional operators were added for the use of menus and sketching. These are:

- *Visual Searching* (***VS***). This operator was used by Lee et al. [103] to represent the task of searching behaviours in a list. In this study, the VS operator is used when users need to find an option within the interface.
- *Dragging and Sketching* (***D***). This operator covers two actions since they are similar – both involve clicking and holding while moving the mouse to a target. The original KLM includes a *Drawing* operator that represents the drawing of a straight line between two points. This operator was used under particular constraints such as limiting the cursor to a specific grid. In this study, the operator is used for sketching without any constraints. Furthermore, dragging is also represented by this operator. Dragging and pointing could be considered as the same operation. However, Gillan et al. [41] and MacKenzie et al. [111] carried out studies to compare the time taken to perform the same task using pointing and dragging. The results showed that dragging is slower than pointing; therefore, it is included in the sketching operator instead of pointing (***P***).

Table 5.3 summarises the operators used in this study.

Operators	Description
K	Keystroke or mouse click
VS	Visual searching for item
P	Pointing to a target
H	Hand movement mouse-keyboard
D	Dragging or sketching
R	Response time

Table 5.3: Summary of operators

5.4.2 KLM Results

The results of this section are divided into three parts. First, I[1] created the scenarios in both systems, counting the number of operations and measuring the time needed to complete each task. These values are considered as the expert user results and are compared with the performance of the study participants. Second, the videos recorded in the study are analysed to obtain the KLM results of each participant. Last, a free tool is used to monitor the mouse and keyboard activity during the creation of the scenarios. This application produces statistics such as distance, number of keystrokes, mouse clicks and mouse wheel scrolls. These metrics give a broad idea of the user activity during each scenario.

Expert results

Table 5.4 and Figure 5.6 show the number of low-level operations, performed by an expert user, for each scenario using MassMotion and the sketching approach. It is important to clarify that the tasks were done separately. It is possible to complete the scenarios with fewer operations if tasks are combined. For example, renaming an entrance and setting the number of spawning pedestrians can be done by opening the entrance configuration menu only one time. The operator *Response Time* (***R***) is not included because it is only used during the compilation in MassMotion.

		Operators				
Scenario	System	K	VS	P	H	D
1	MassMotion	89	37	63	5	25
1	Sketching	64	16	26	5	4
2	MassMotion	193	76	118	12	54
2	Sketching	173	38	62	21	6
3	MassMotion	336	156	246	16	71
3	Sketching	175	43	78	13	12

Table 5.4: Operators needed to complete the three scenarios in both systems.

The results indicate that MassMotion needs significantly more actions to complete the scenarios. This finding is more evident in Scenario 3 since the creation of areas in MassMotion requires multiple steps (create floor, links and connect them) whereas, in the sketching approach,

[1]The word 'I' is used here to make it clear that I am the expert operator

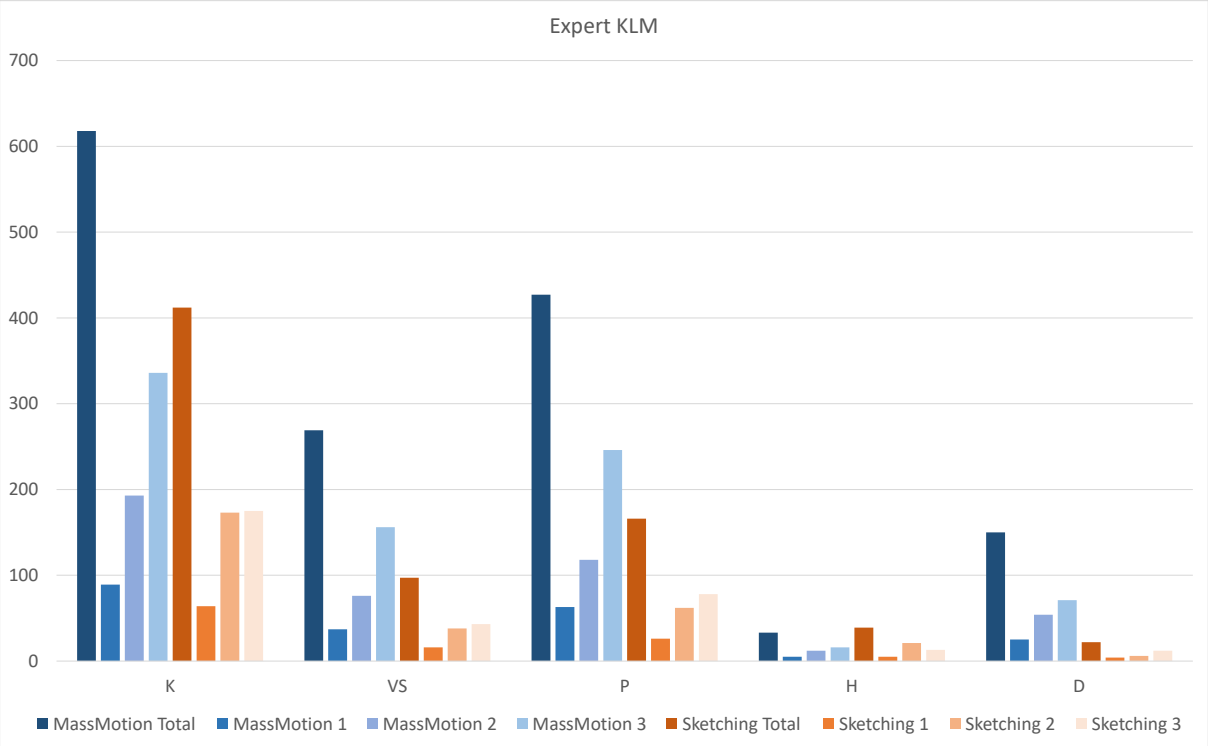


Figure 5.6: Operators needed by expert user to complete the three scenarios in both systems.

this can be done with one stroke. A factor to consider is that the MassMotion interface is more complicated since it provides more options to control the simulation. Consequently, more actions are needed to locate the desired option. More detailed information about the scenarios, tasks and low-level operations is included in Appendix B.

Participants results

Figure 5.7 shows the average number of tasks performed by the participants to complete the scenarios in both systems. The chart includes the total of each operator and the individual value for every scenario. In general, the number of operators increases as the scenarios get more complex. A significant finding is that for all the operators, the total value in sketching is less than the tasks in Scenario 3 for MassMotion. In other words, with the actions done to complete only Scenario 3 in MassMotion, users would have finished all the scenarios using the sketching approach. The error bars indicate the standard error of the population. This is calculated as follows: $SE = \sigma/\sqrt{n}$ where σ is the standard deviation and n the population size. This error is considerably smaller for sketching operators, indicating that the population range of values is smaller. This suggests that the sketching approach has a gentle learning curve compared to MassMotion.

Chart 5.8 compares the performance of the participants (dark shades) and expert user results (light shades). The results show a bigger difference between participants and the expert while using MassMotion. Operators like ***VS*** and ***P*** have similar values for participants and expert in

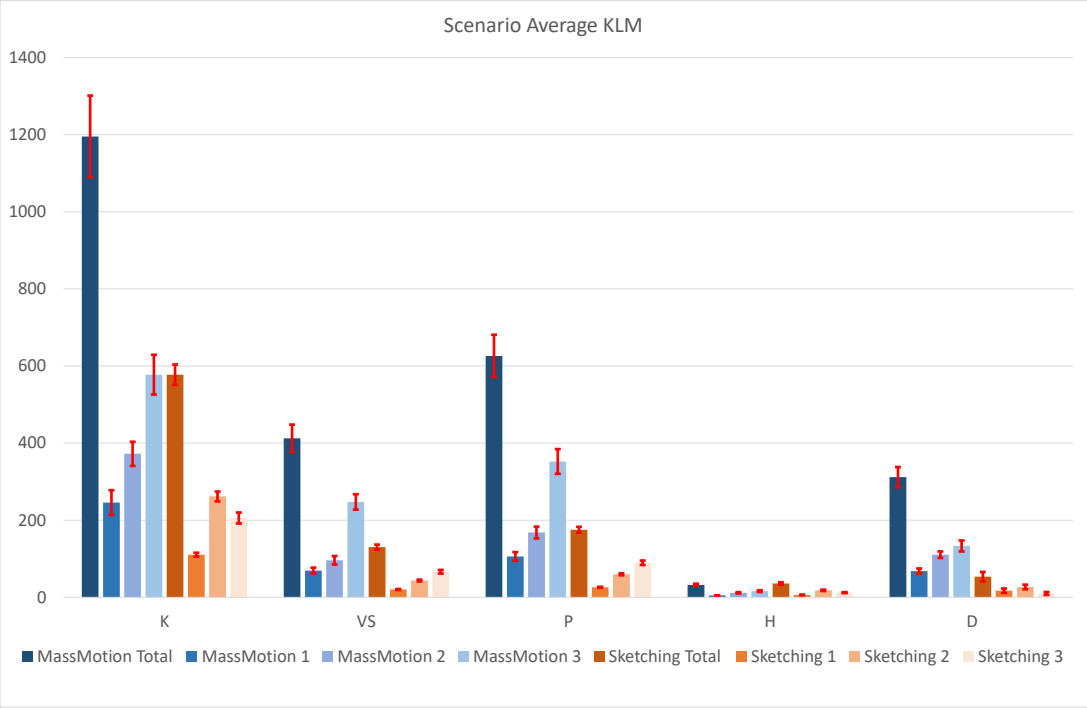


Figure 5.7: Average number of tasks done by study participants in all 3 scenarios and in total for MassMotion (blue) and sketching (orange). The red error bars show the standard error.

the sketching approach. This result again supports the fact that the sketching interface is more intuitive for inexperienced users.

Metrics

A free tool was used to collect data about the mouse and keyboard activity during the study. This tool provides more detailed information about some of the KLM, for example, left, right, middle and double clicks which are all included in the ***K*** operator. All the information obtained and the KLM analysis are included in Appendix B. Figure 5.9 shows the average participant metrics in total and for each scenario. The metrics coincide with the KLM results as they show more user activity during the MassMotion scenarios. Each metric is explained in the following sections.

Distance

This value indicates the distance in metres travelled by the mouse cursor. This metric is not a reliable indicator since it depends on the camera zoom rather than the user interface. Figure 5.9 shows a noticeable difference in the distance travelled since MassMotion controls the camera position by movement of the mouse, whereas the sketching system uses the keyboard.

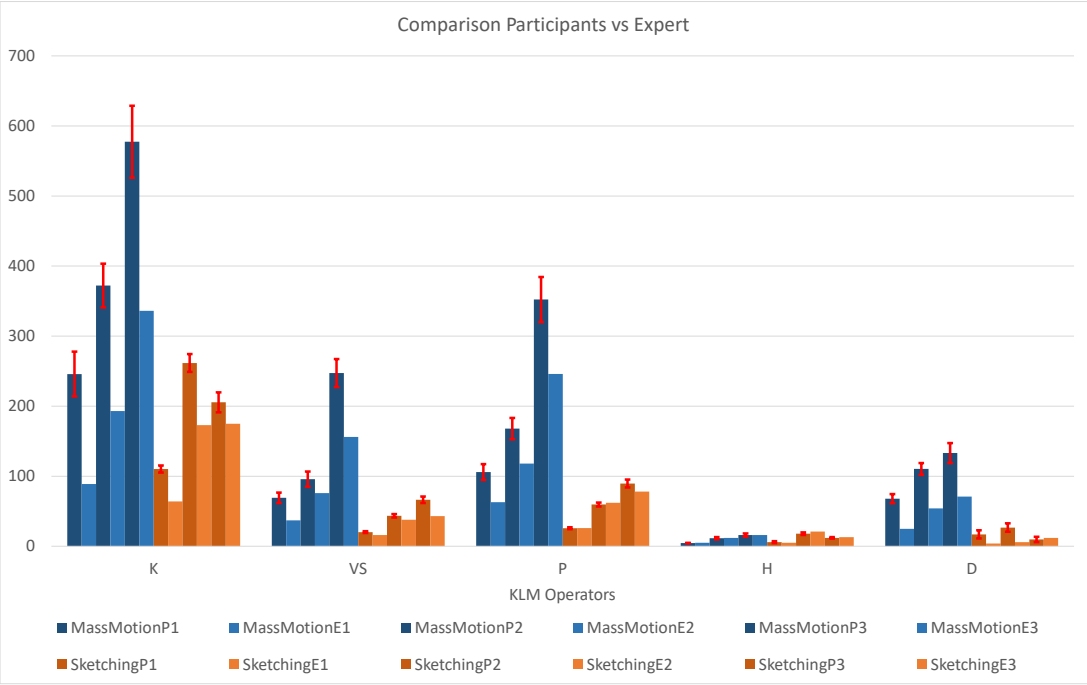


Figure 5.8: Comparison between participants (dark shades) and expert user (light shades). MassMotionP1 is the participant average for Scenario 1 in Massmotion. Similarly, MassMotionE1 is the expert user result for each operator in Scenario 1 in MassMotion. The red error bars show the standard error of the participant averages.

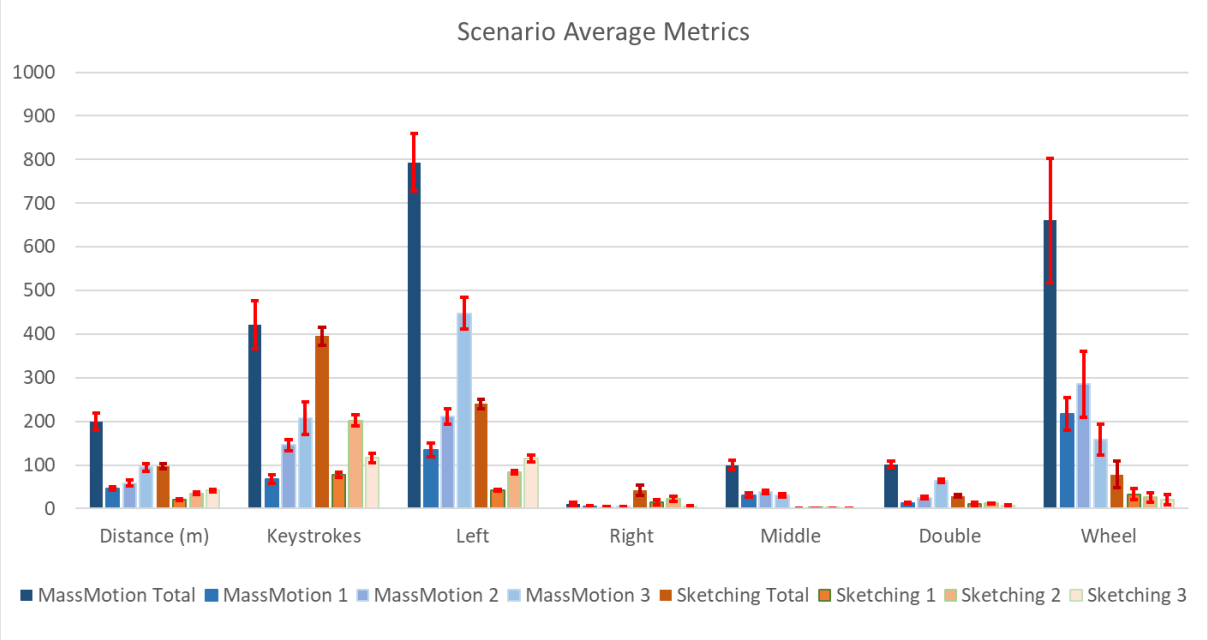


Figure 5.9: Metrics participant average in every scenario and total. MassMotion values are shown in blue. Sketching values are shown in orange. The standard error is represented by the red error bars.

Keystrokes

This metric shows the number of keystrokes required to complete each scenario. The total number of keystrokes is similar for MassMotion and Sketching, however, there are differences in some scenarios. Participants needed more keystrokes to complete Scenario 1 and 2 in the sketching system (compared to MassMotion) because the camera is controlled using the WASD keys. The camera control represents most of the keystrokes; this can be observed by comparing Scenario 3 in both systems. This scenario does not require the camera to be constantly moved since all the elements in the environment are in the same area. Thus, fewer keystrokes were needed to complete this scenario.

Left Click

This operator is the best indicator since most of the tasks involve left-clicking. In all scenarios, the number of left clicks was more than double for MassMotion. This is similar to the ***K*** operator results, however, the metrics show a more significant difference because MassMotion uses the mouse to control the camera.

Right, Middle and Double Click

These indicators are not relevant since they were not frequently used during the study. Right and middle-click only highlight the different camera control used by both systems. The right-click is used to rotate the camera in the sketching approach, and the middle button is used in MassMotion to pan the camera.

Mouse Wheel

The mouse wheel is only used to zoom the camera. The mouse wheel was used a lot more in MassMotion due to two reasons. First, some delicate tasks in MassMotion, such as connecting the area links, require the camera to be moved close to the area to ensure the correct placement of the links. Second, the sketching also uses the 'w' and 's' key to zoom in and out, reducing the use of the mouse wheel.

5.5 Efficiency and Effectiveness

The efficiency of a system is determined by the time taken to complete the scenarios. The time was measured for all the participants and the expert user in both interfaces. Table 5.5 and Figure 5.10 show the time needed by the expert user to complete all the scenarios. The time is divided into setup, compilation and run because MassMotion compiles the simulation before visualising it. Compiling the simulation ensures that setup is correct and the simulation runs correctly. The compilation time is not present in the sketching approach since it simulates in real-time. The results of this table are consistent with the number of operations obtained with the KLM model. The most significant time to highlight the intuitiveness of the user interface is the 'Setup' time. Completing the scenarios in MassMotion took at least twice the time needed in

the sketching interface. The 'Simulation' time differs because the simulation was run at different speeds during some intervals of time.

		Time (s)			
Scenario	System	Setup	Compilation	Simulation	Total
1	MassMotion	195	91	126	412
1	Sketching	63	0	108	171
2	MassMotion	330	88	63	481
2	Sketching	147	0	73	220
3	MassMotion	566	165	86	817
3	Sketching	141	0	98	239

Table 5.5: Time needed to complete the three scenarios in both systems by the expert user.

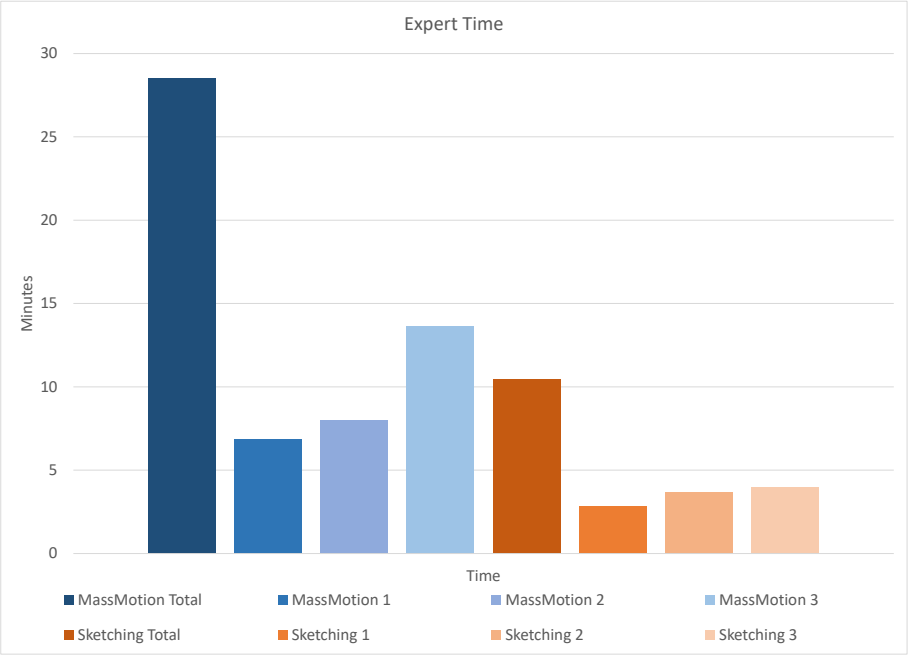


Figure 5.10: Time taken in minutes by the expert user to complete every scenario using both systems.

The times of all the study participants are shown in Figure 5.11. Similar to the expert user, the sketching system completed the scenarios in half of the time compared to MassMotion. The time increases with the complexity of the scenario. The complexity depends on the type and number of elements included in the scenario. A similar finding to KLM can be observed in the chart. Completing Scenario 3 in MassMotion took longer than finishing all the scenarios with the sketching interface for all the participants. Figure 5.12 shows detailed average time information about the scenarios. Setting up the simulation takes most of the time in each scenario. The 'Simulation' time is lower in MassMotion since most of the participants ran the simulation at 20x speed, while the sketching system was limited to 5x speed.

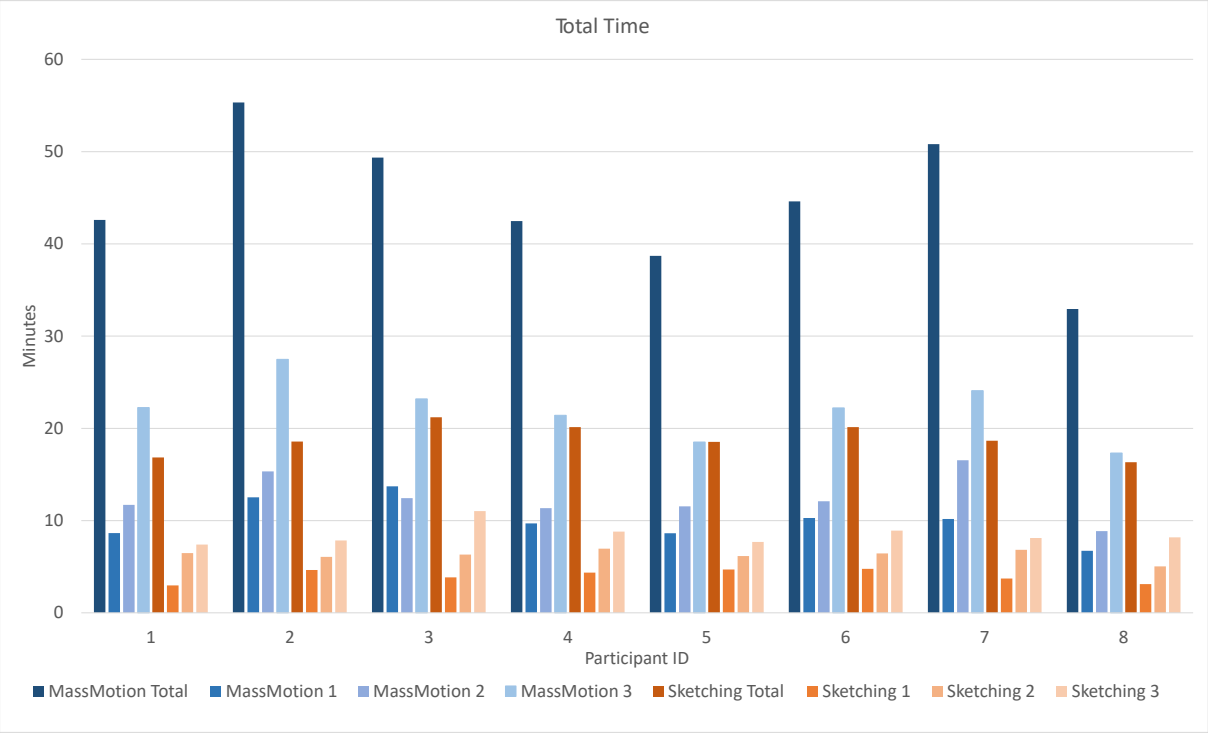


Figure 5.11: Time taken in minutes by each participant to complete every scenario using both systems.

The effectiveness of a system is calculated by dividing the number of tasks by the number of attempts required to fulfil them. The scenarios are divided into high-level tasks to calculate the effectiveness of the interfaces. Tables 5.6, 5.7 and 5.8 show all the tasks, number of attempts and the effectiveness(%) for every scenario in both systems. Overall, MassMotion had higher effectiveness. Three tasks were difficult to complete using the sketching interface. The first task was the creation of entrances/exits. An issue was found when double-clicking to create an entrance. This problem produced two entrances in the same location affecting the resulting simulation. The second task was the creation of a barrier. A different criterion of what counted as an attempt was used for this task. In MassMotion, participants could translate, scale and rotate the barrier as many times as needed before compiling the simulation. A missed barrier creation attempt was counted only when participants ran the simulation, and the barrier did not completely block the path of the crowd. In the sketching approach, a failed attempt was counted every time participants sketched a barrier that did not block the entire path of the crowd. The reason for this is that users cannot modify existing barriers, so the misplaced barrier had to be replaced by a new obstacle. A bad camera angle not showing the entire path was the most common cause of this mistake. The third task was the creation of journeys or storyboards. The sketching interface does not show enough feedback to guide users while creating a storyboard. All these issues could be easily addressed to improve the usability and effectiveness of the sketching interface.

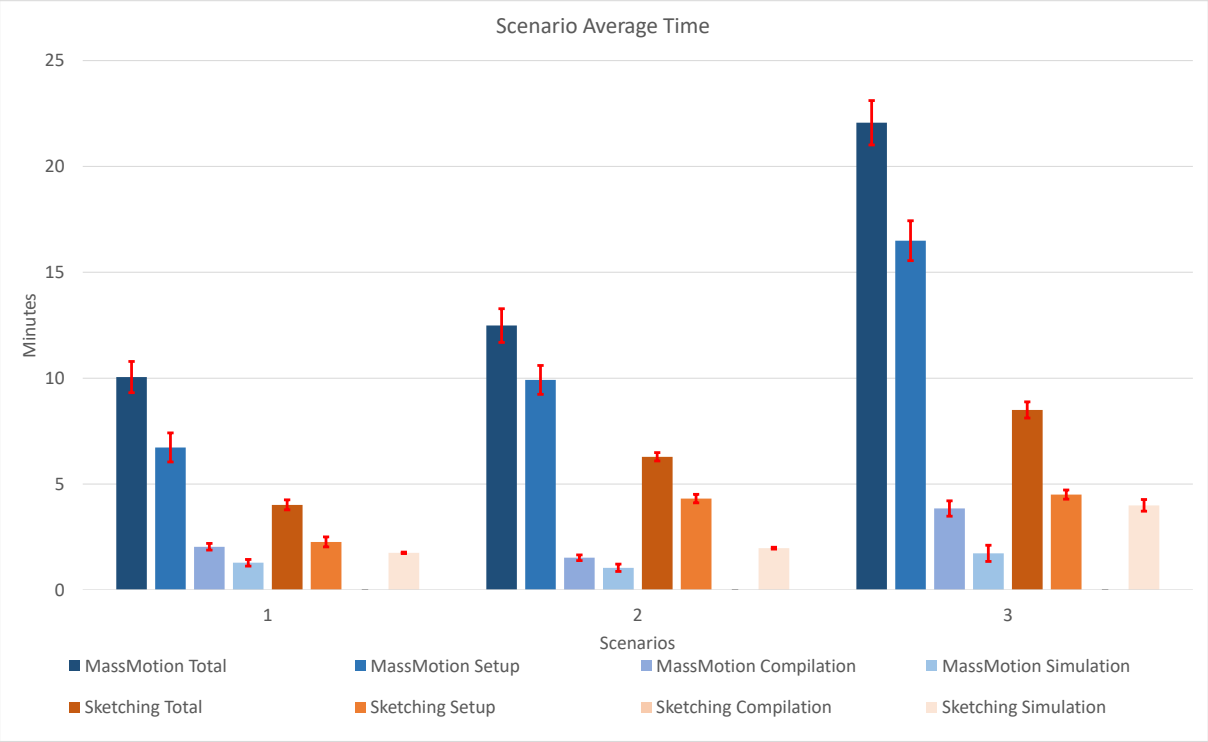


Figure 5.12: Participant average time for each scenario. Time is divided into total, setup, compilation and simulation. The red error bars show the standard error of the population.

		MassMotion		Sketching	
Scenario 1 Tasks	Total	No. Attempts	%	No. Attempts	%
Create and rename the entrance	8	11	72.7	8	100
Create and rename the exit	8	9	88.9	12	66.7
Spawn 1,000 pedestrians at a 10 ped/s rate	8	10	80	11	72.7
Run the simulation	8	8	100	8	100
Stop the simulation	8	8	100	8	100
Create the barrier	8	9	88.9	14	57.1
Run simulation	8	8	100	8	100
Stop the simulation	8	8	100	8	100
	64	71	90.1	77	83.1

Table 5.6: Tasks, number of attempts and effectiveness of both systems for the first scenario.

		MassMotion		Sketching	
Scenario 2 Tasks	Total	No. Attempts	%	No. Attempts	%
Create and rename the entrances	24	27	88.9	24	100
Create and rename the exits	24	27	88.9	32	75
Spawn 900 peds. per entrance at a 3 p/s rate	24	24	100	24	100
Divide each entrance crowd into 3 groups	24	25	96	24	100
Run simulation	8	8	100	8	100
Stop the simulation	8	8	100	8	100
	112	119	94.1	120	93.3

Table 5.7: Tasks, number of attempts and effectiveness of both systems for the second scenario.

		MassMotion		Sketching	
Scenario 3 Tasks	Total	No. Attempts	%	No. Attempts	%
Create and rename the entrances	16	16	100	17	94.1
Create and rename the exits	8	8	100	9	100
Create and rename the areas	16	22	72.7	16	100
Spawn 900 peds. per entrance at a 5 p/s rate	16	16	100	16	100
Move crowd from Entrance A to Area 1	8	12	66.7	13	61.5
Move crowd from Entrance B to Area 2	8	11	72.7	15	53.3
Run simulation	8	8	100	8	100
Stop the simulation	8	8	100	8	100
Move crowd from Entrance A to both areas	8	9	88.9	10	80
Move crowd from Entrance B to the exit	8	10	80	8	100
Create the barrier	8	9	88.9	9	88.9
Run simulation	8	8	100	8	100
Stop the simulation	8	8	100	8	100
	128	145	88.2	144	88.9

Table 5.8: Tasks, number of attempts and effectiveness of both systems for the third scenario.

5.6 System Usability Scale results

The System Usability Scale (SUS) questionnaire consists of ten questions (the questionnaire is included in Appendix B) that users answer using a 5-step Likert from "strongly disagree" to "strongly agree". The score ranges from 0 to 100, with higher scores indicating better usability. To obtain the score, each item is assigned with a value from 0 to 4. Since every question has five possible answers, the most negative answer receives a value of 0 and the most positive a value of 4. The sum of all items is multiplied by 2.5 to get the final score. The score for each participant is shown in Figure 5.13. There are some outliers such as participant 3, who has almost a perfect rating for both systems, and participant 6 who gave MassMotion a low score. However, these two participants do not have a substantial impact on the average score, which is 57.2 for MassMotion and 86.6 for the sketching system. These numbers do not indicate if a user finds a system usable. Bangor et al. [11] tried to match the SUS score to user opinion. Their study added an eleventh question to get an overall usability user perspective of the system. The question has seven options from "Worst imaginable" to "Best imaginable". The findings of the study are shown in Table 5.9. According to the results, the usability of MassMotion is 'OK-Good', whereas the sketching approach is 'Excellent'.

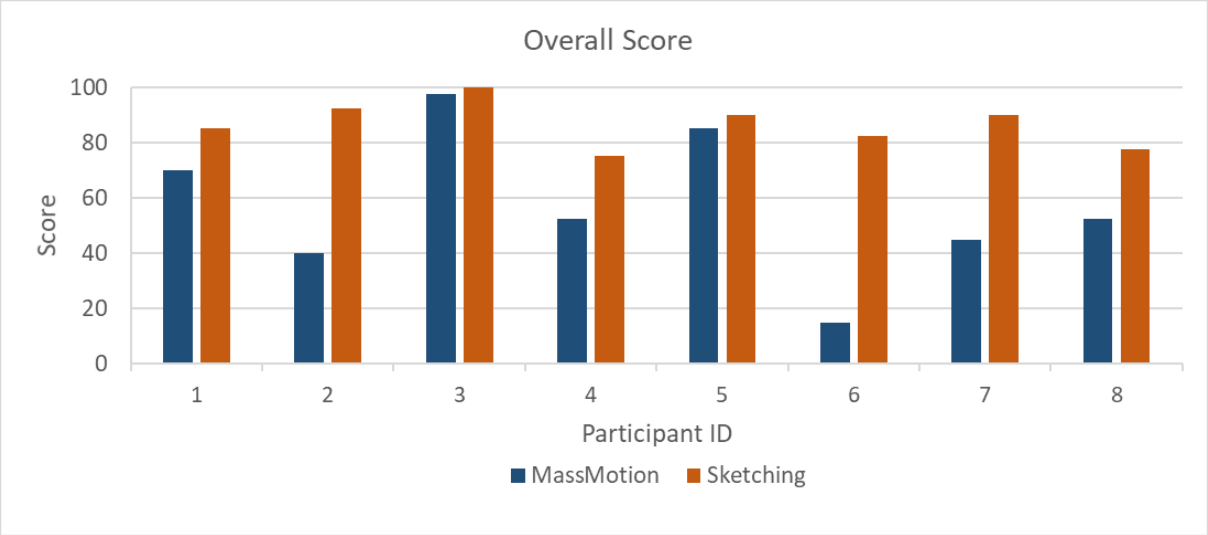


Figure 5.13: Overall SUS score for every participant.

Figure 5.14 shows the average score for all ten questions of the survey. Items 3, 4, 7 and 10 have the most considerable score difference between systems. These questions are related to the intuitiveness of the system and the ease of use for non-experienced users.

Adjective	Mean Score
Worst Imaginable	12.5
Awful	20.3
OK	50.9
Good	71.4
Excellent	85.5
Best Imaginable	90.9

Table 5.9: Adjective given to the user-friendliness of the system by users and the mean score of the SUS score.

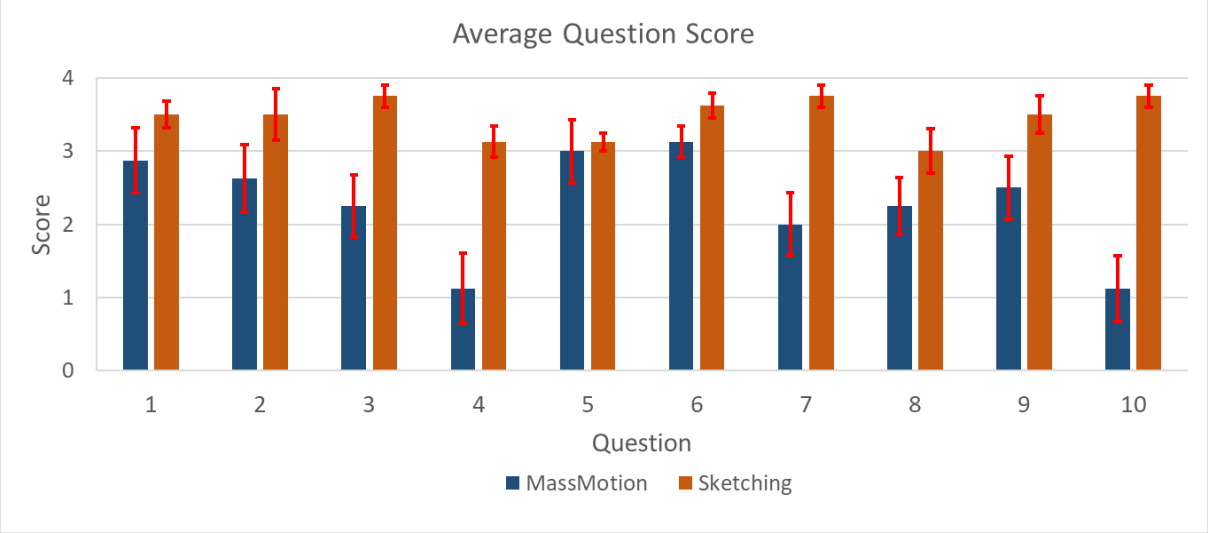


Figure 5.14: Average score for each question of the SUS survey. The red error bars show the standard error of the population.

5.7 Summary

This chapter has investigated the differences between sketching and an alternative approach to create crowd simulations. This method is the commercial system MassMotion. Commercial systems do not use sketching to change aspects of the simulation; instead, they have complex interfaces to tweak parameters. These sophisticated interfaces require a certain level of knowledge to find the desired parameter. Therefore, non-experts users might find it challenging to set up a simulation. An advantage of the commercial systems over the sketching approach is the level of control they provide. Many options are included to create the environment and define the behaviour of the crowd. The high level of control and flexibility are the main reasons for having complex interfaces.

Commercial systems typically interact with the simulation before running. The environment is created, entrances and exits are defined, the crowd journeys are specified, and special behaviours are described in an offline stage. Dynamic changes are not possible since the setup information needs to be compiled before visualising the simulation. Taking MassMotion as an example, some dynamic possibilities exist. Using its SDK, it is possible to do some actions in the middle of the simulation, such as creating new agents. However, environment modification is not possible since the maps used for crowd navigation are obtained during the compilation and cannot be recomputed in real-time. An attempt to add an obstacle while the simulation is running was made by adding static agents to simulate a barrier (Appendix A). The crowd did avoid the static agents producing a similar result to an obstacle. However, this experiment is still not real-time since it is defined before compiling the simulation. Furthermore, this is not an intuitive approach since users need to be familiar with the API and specify the exact simulation frame and agent positions.

This chapter has evaluated the sketching approach by comparing it against MassMotion. Two main aspects were evaluated. First, the realism of the simulation was assessed using the benchmark suite 'steersuite'. This tool calculates agent metrics such as total time, distance, speed and acceleration based on the agent position on every frame. The results showed slight differences in the agent models, which did not affect the resulting simulation.

Second, a user study was carried out to evaluate the graphical user interface. The experiment consisted in creating three scenarios using both systems. The user study results were compared according to the following criteria: KLM, efficiency, effectiveness and usability. An adapted version of the KLM model was proposed to calculate the number of actions needed to complete the experiment. KLM results were obtained from an expert user and the participants of the study. For both cases, MassMotion required more operations to complete every scenario. The difference between the sketching approach and MassMotion is more evident for the study participants. They performed more actions in Scenario 3 using MassMotion than the total number of operations for all the scenarios in the sketching interface. A more significant gap can be observed in MassMotion when comparing the results of the expert user and the participants. This disparity suggests that the sketching approach is more intuitive for non-expert users.

The efficiency of the systems was measured by the time taken to finish the experiment. MassMotion required twice as much time as the sketching interface. The effectiveness was computed by dividing the number of tasks by the number of attempts done. The sketching

approach was slightly less effective for two reasons. First, users incorrectly created duplicate entrances and exits in the sketching approach, since it was not disallowed by the interface. The second reason was a lack of feedback during the creation of storyboards. Some participants struggled to complete a storyboard since they were not sure in what step of the creation process they were. A possible solution for this would be for the system to provide feedback indicating what elements (entrance, area or exit) can be selected in each step. Last, the System Usability Scale (SUS) was used to assess the usability of both interfaces. The average score of the questionnaire was matched to an adjective describing the user perspective on the system. The user experience using MassMotion was 'OK-Good' whereas the sketching approach was 'Excellent'.

The KLM and efficiency results of the evaluation would not be as different if MassMotion expert users were recruited for the experiment. However, the sketching system would still require fewer tasks and time to complete the scenarios. The reason is the design of the interface. For instance, to create an obstacle in MassMotion, users must create, place, rotate and scale the barrier. Whereas in the sketching approach, only one stroke is needed to accomplish the same.

Chapter 6

Virtual Reality

Virtual Reality (VR) immersion can be used to design buildings or vehicles, learn/train and comprehend complex phenomena [7]. In this chapter, VR allows users to experience at first hand the real-time modifications of a pedestrian environment which provides a better insight about how pedestrians react to different circumstances (comprehension), which could lead to a better design of venues, train stations among others (design). The chapter will cover the advantages and disadvantages of sketching in an immersive environment to set up and control a pedestrian simulation while being part of the crowd. The aim of this chapter is to investigate the viability of sketching immersed in the environment compared to the traditional desktop interface with mouse and keyboard.

Section 6.1 briefly describes the implementation of the VR interface. Two control modes are implemented to offer flexibility and a better user experience: edit and pedestrian mode. Edit mode is similar to the desktop interface. A top-view camera is used to move around the environment quickly and to configure the simulation. Pedestrian mode places the camera at ground level to simulate being part of the crowd. Users can still control the simulation in this mode, but the field of view is limited compared to the previous camera. Virtual pedestrians react to the user position to avoid collisions. The hardware used for the implementation is the *Oculus Rift*[1] headset and controllers. This equipment was used since it does not require a large area to work properly. Figure 6.1 shows the user running a crowd simulation using the VR headset.

Section 6.2 proposes an adapted KLM version with new operators specific to the VR interface. The same three scenarios from Section 5.3 are used to compare the KLM results and timings of the expert user[2] using both interfaces. No user study was conducted due to the COVID-19 pandemic restrictions. The analysis performed in this chapter only considers the results of the expert user. Section 6.3 compares the desktop and VR interfaces. Section 6.4 offers a brief consideration of using other devices for a sketch-based approach. Section 6.5 then summarises the advantages and disadvantages of sketching in an immersive environment.

[1] https://www.oculus.com/

[2] I am the expert operator

Figure 6.1: User with VR headset running a crowd simulation.

6.1 Implementation

Unreal Engine has built-in support for Oculus to facilitate the implementation of the VR interface. The headset is enabled by toggling a run-time option, and the camera rotates following the orientation of the headset. The user input devices must be carefully selected since standard input methods such as mouse and keyboard cannot be used in an immersive VR experience [28]. The *Oculus* controllers offer a variety of actions and their position can be tracked in the VR environment, making them a suitable alternative to the mouse. Two controllers are required to move inside the virtual world and to control the simulation via sketching. The *Oculus Touch* controllers are enabled by creating a predefined object for each controller. A 3D model is added to the *Unreal* objects to visualise them in the simulation as shown in Figure 6.2. A crucial aspect to consider when creating the VR interface is motion sickness, which is the most common adverse health effect derived from the use of a VR headset. Scene motion and the illusion of self-motion (vection) are two important contributors to motion sickness [70]. Therefore, the character and camera movement speed were carefully defined to prevent the symptoms of motion sickness.

6.1.1 Controllers

The left controller is used to move through the environment, change modes and control the position of the menu (Section 6.1.2). The movement of the camera depends on the selected mode – a detailed explanation is given in Sections 6.1.3 and 6.1.4. The right controller interacts with the environment (sketching) and with the user interface. Figure 6.2 shows the function of the buttons used in the system.

To interact with the environment and menu, users need to know where the controller is pointing inside the 3D world. This position is calculated by tracing a line from the right

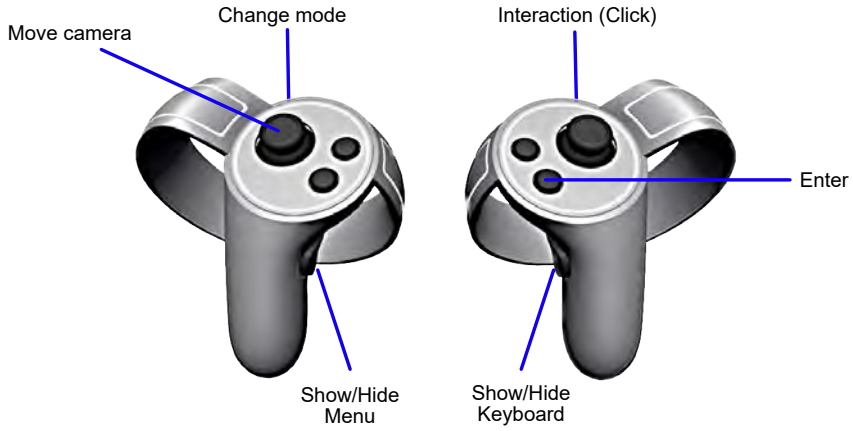


Figure 6.2: Controller configuration.

controller location to a distant point in the direction faced by the controller. Figure 6.3 shows the line traced and an indicator where the line collides with the environment to let the user know the object that is being hovered. The size of the indicator is adjusted, based on the distance between the impact point and the controller, to maintain the same aspect ratio from the user's perspective. The trigger button of the right controller is used as a mouse click. Its functionality depends on the hovered object. If the user is pointing at the menu interface, then it only selects the corresponding option. Otherwise, the trigger button will create (sketch) or select (click) an element in the navmesh.

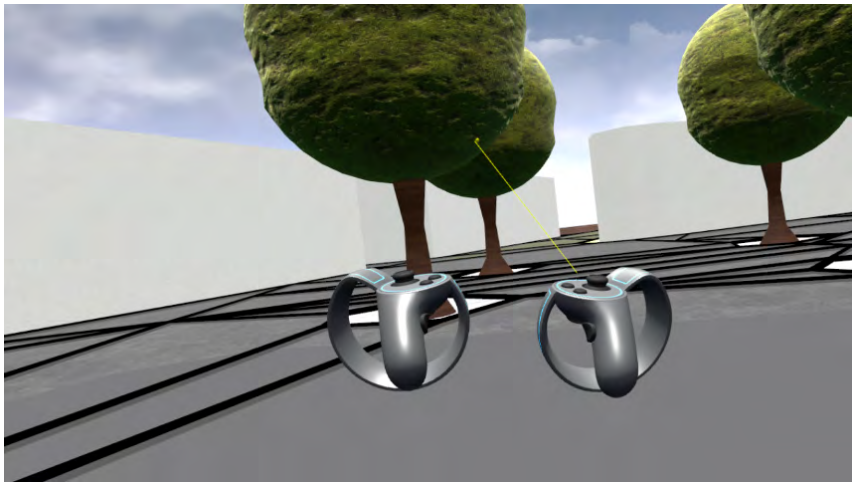


Figure 6.3: Controllers and yellow line traced to indicate the location that the user is pointing.

6.1.2 Menu

A user interface is created to set up and control the simulation (Figure 6.4). The desktop interface menu in Section 3.3.4 can not be used since the user is now immersed in the virtual environment. Therefore the menu must be a 3D object that is part of the world. Another

difference is that fewer options are included to keep the interface small and easy to use with the new input device. For instance, the timeline feature is not implemented on the VR interface. Some UI design guidelines described in [62] are followed to create the menu interface, for example, translate the familiar functionality of the 2D interface to the 3D world – this new menu must produce the same results to meet user expectations. Moreover, the position of the interface components should be carefully defined since reaching distant elements can cause discomfort to the users. The menu is designed to follow the left controller to avoid having it in a fixed location. In this manner, users can control the menu position by moving the left arm. Additionally, the menu can be hidden when not used so as not to interfere with the rest of the scene.

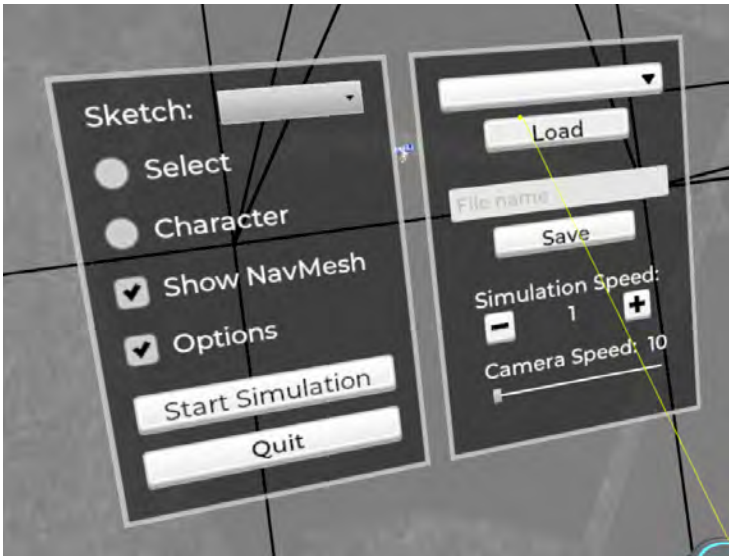


Figure 6.4: 3D VR Menu used to set up and control the simulation.

The actions of sketching elements (barrier, flowlines, areas), select, show navmesh and the options to save/load configurations are described in Section 3.3.4. A new option 'Character' is added in Edit mode (Section 6.1.3) to allow the user to place the character controlled in Pedestrian mode (Section 6.1.4). This function is described in the next section. The 'Camera Speed' option only works for Edit mode, and it is considerably slower than the desktop interface version to prevent motion sickness.

Specific actions, such as changing the name of an element or modifying a percentage, require a keyboard to enter alphanumerical values. The physical keyboard cannot be used since users are wearing the VR headset. This issue is solved by including a virtual keyboard (Figure 6.5) in the user interface. A disadvantage of this approach is that users must find, point to and select every key. This process could be a time-consuming task for long element names. According to Alger [3], comfort, speed, and accuracy are the principal objectives of text input in VR. The proposed solution does not meet the speed requirement. Some alternatives are discussed in the following sections when comparing the desktop and VR interfaces.

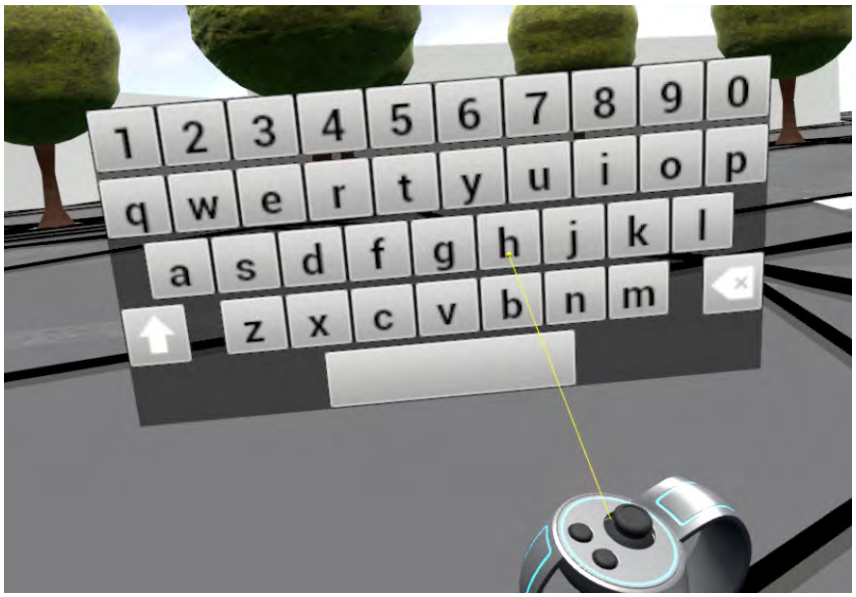


Figure 6.5: Virtual keyboard used for alphanumerical inputs.

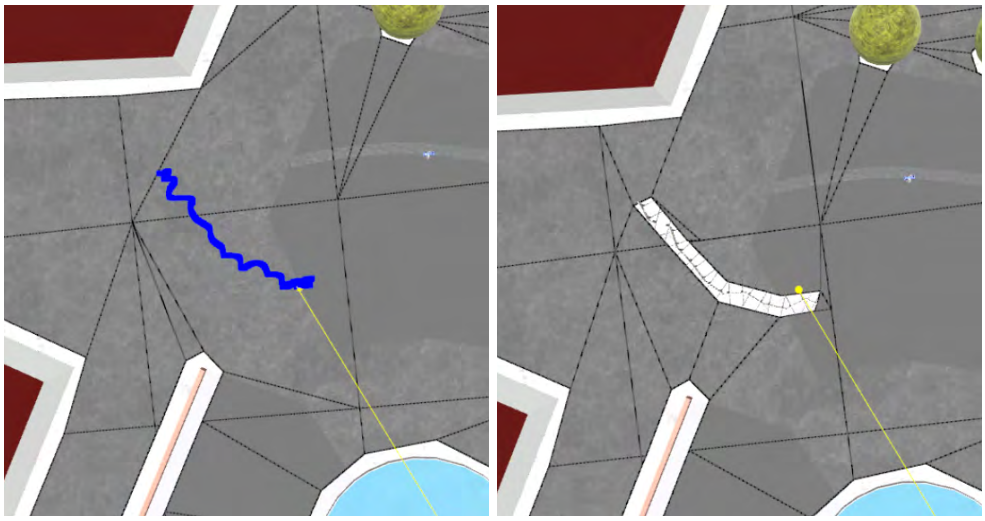
6.1.3 Edit mode

In this mode, the camera is located above the simulation plane to have a better view of the entire environment. This top view facilitates the setup and control of the simulation. The user can move the camera freely around the 3D world, similar to the desktop interface. Figure 6.6 shows the sketching of a barrier in Edit mode. The blue line is wavy since sketching with the VR controller is more difficult than using the mouse. Section 6.3.1 compares sketch speed and accuracy for both interfaces. The sketching and sampling process is the same as the one described in Section 3.1.4. The only difference is the input device. The creation of entrances/exits and storyboards is the same. Figure 6.7 illustrates this process. The yellow entrance is created in Figure 6.7a by selecting a polygon edge with no adjacent polygon on the other side. The red exit in Figure 6.7b is created in the same way. The journey of the storyboard is defined by selecting the starting entrance in Figure 6.7c and then the final the destination in Figure 6.7d. This example does not include intermediate waypoints.

In Edit mode, the user can determine the position of the character (camera) that is used in Pedestrian mode (Section 6.1.4). The option is included because moving around the environment in Pedestrian mode could be relatively slow. The teleportation of the character is done by selecting the menu option 'Character' and pointing to the desired position, as shown in Figure 6.8. A 3D humanoid model and a camera are used to indicate the position of the user-controlled character.

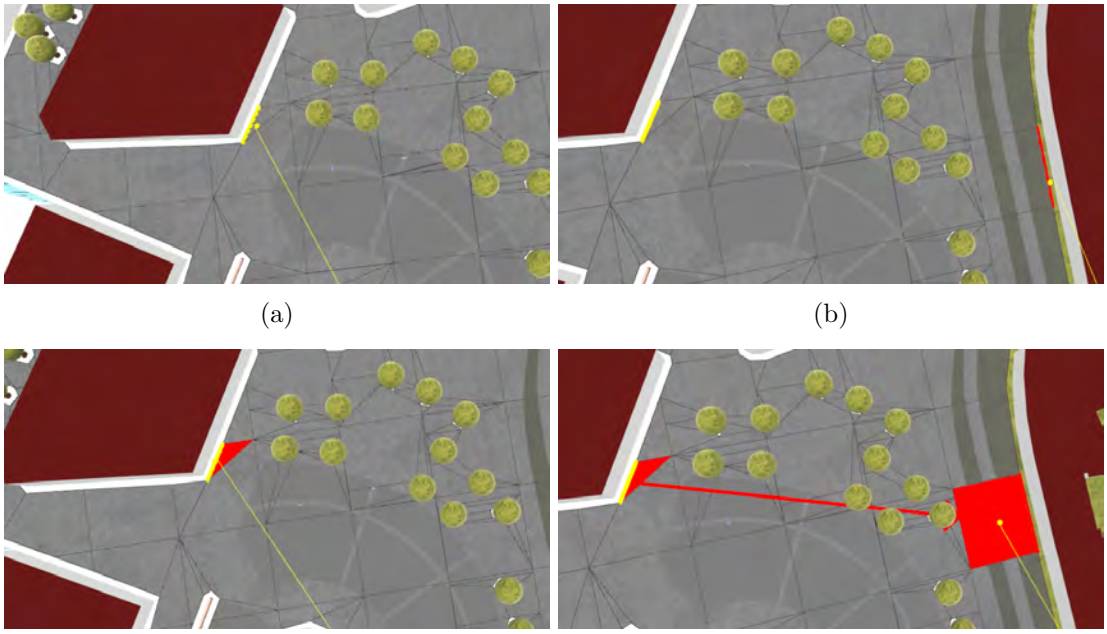
6.1.4 Pedestrian mode

The second way of interacting with the simulation is the Pedestrian mode. The camera is positioned at ground level to simulate being part of the crowd. The camera speed is fixed to the average human walking speed, which is the same as the virtual crowd. This camera



(a) (b)

Figure 6.6: Creating of a barrier in Edit mode. (a) Blue line sketched by the user. (b) Barrier created from the sketch.



(a) (b)

(c) (d)

Figure 6.7: Storyboard creation process. (a) Select a polygon edge to create the yellow entrance. (b) Select a polygon edge to create the red exit. (c) Select the starting point of the storyboard. (d) Select the final destination of the storyboard.

setup produces a more realistic immersive experience and prevents motion sickness. Visual accelerations and sickness represent a smaller problem when users control the camera. However, it is essential to have discrete speeds and avoid slow transitions when using analogue sticks as

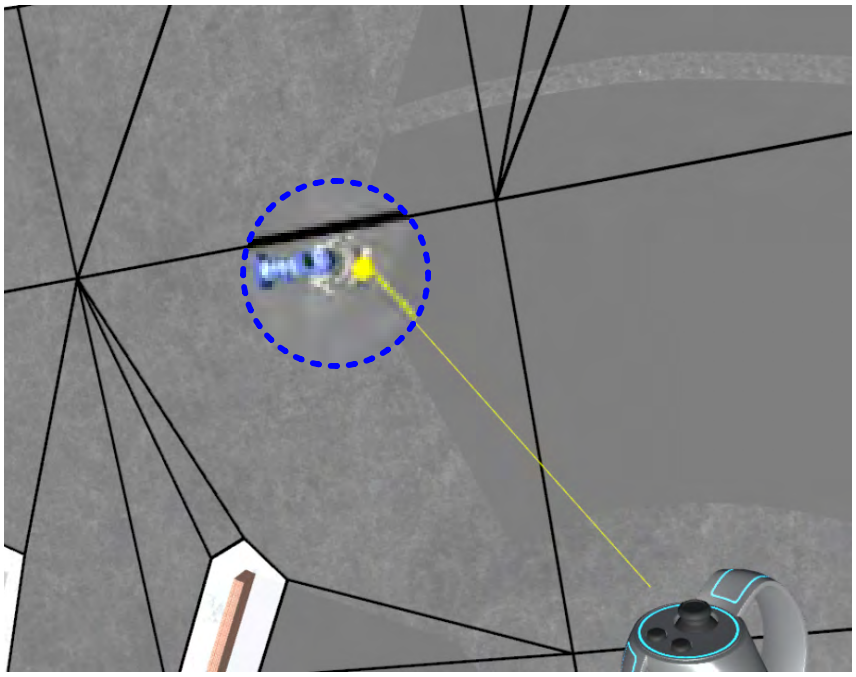


Figure 6.8: User selecting the position of the virtual character for Pedestrian mode. The blue dotted circle area is enlarged to better show the character.

the input device [70]. Therefore, the movement speed of the character instantly changes from 0 to the maximum speed. In this mode, users can only move forward and backwards in the direction they are facing. The position of the character is sent to the FLAME GPU simulation each frame to generate a repulsive force to avoid collisions with the pedestrians.

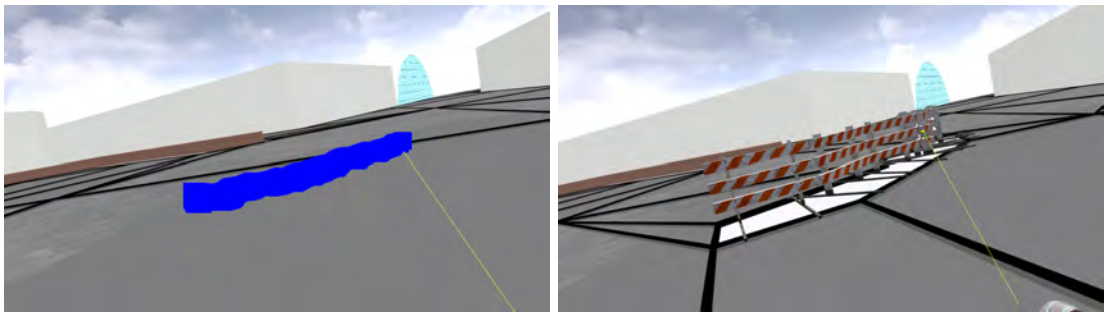
The environment can still be modified in this mode. Figure 6.9 shows a barrier created in Pedestrian mode. The drawback is that elements can only be created in the surrounding areas due to the limited field of view. Figure 6.10 shows four different frames of a simulation running in Pedestrian mode. In 6.10a, the user is observing a barrier and the crowd from a distance. Then it moves closer to the pedestrians (6.10b) and starts walking in the same direction (6.10c). The last frame (6.10d) shows the user walking in the opposite direction of the crowd, and the agents steer away to not collide with the virtual character.

6.2 KLM

This section presents the KLM results after creating the three scenarios used in Section 5.3 in Virtual Reality. It also discusses how the selection of input devices impacts on the performance of the interface, and the use of KLM to evaluate VR interfaces.

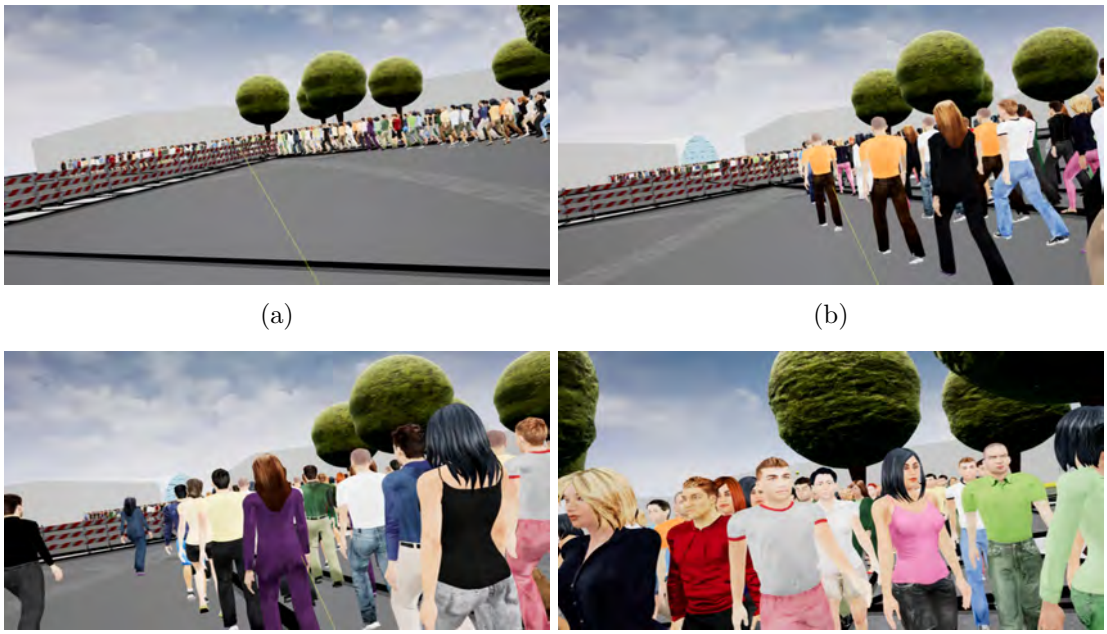
6.2.1 KLM modifications

The original KLM has to be modified to cover all the possible actions that can be done with the VR interface. However, there is a shortage of work exploring the use of KLM for more modern



(a) (b)

Figure 6.9: Barrier created by the user in Pedestrian mode. (a) User-sketched line. (b) Barrier spawned after finishing the sketch.



(a) (b)

(c) (d)

Figure 6.10: Four frames of a simulation in Pedestrian mode. (a) User observing the crowd from the distance. (b) User moves closer to the pedestrians. (c) User walks in the same direction of the group. (d) Agents avoid colliding with the user who is walking in the opposite direction.

interfaces, including VR and Augmented Reality [2]. Therefore, new operators are proposed to represent the low-level tasks needed to complete the scenarios using the VR interface. Table 6.1 summarises the operators used in the KLM analysis. Three new operators are added compared to the KLM approach used in Section 5.4.1. First, ***A*** represents the left arm movement of the user. This motion is required to control the position of the menu in the 3D world. Second, ***S*** corresponds to the use of the controller analogue stick to move the camera and the VR character. Last, ***Hd*** counts the number of times a user has to move his head to rotate the camera before moving it to the desired location. An additional difference is the ***H*** operator which is not

required for the VR interface. This operator represents the hand movement between devices. The only device used is the VR controller, thus the ***H*** never happens.

Operators	Description
K	Controller Buttons
VS	Visual searching for item
P	Pointing to a target
A	Left arm movement
D	Dragging or sketching
S	Controller analogue stick
Hd	Head movement

Table 6.1: Summary of operators for the VR interface

6.2.2 Results

A KLM analysis is performed to obtain a metric used to compare the desktop and VR interfaces rather than as a predictor of time. Table 6.2 shows the total operators and the time needed for every scenario using both interfaces. Operators not used by an interface are marked as '-'. The table includes two sets of results per scenario. The first covers all the actions in the scenario. The second result (No Key) shows the previous row minus the tasks involving a keyboard. The reason behind having two results is a fairer comparison between both interfaces regarding the setup of the simulation.

The findings show a considerable difference between Desktop and VR interfaces. The operators ***K***, ***VS***, ***P*** and the setup time are greater for VR. The main cause of this discrepancy is the use of the virtual keyboard. In the VR system, users must search, point to and press every letter using the controller. A similar process is followed when using a physical keyboard, but, in general, users are familiar with the device, and multiple fingers can be used at the same time instead of only one controller. After removing the actions involving the keyboard, the results are much closer, but the Desktop interface still requires fewer operations and time. The reason is that users need to show/hide and control the position of the menu and keyboard in the VR interface. Some of the operators perform a similar task despite representing a different action and being used in different interfaces. For example, the ***D*** (Dragging or Sketching) operator for the Desktop interface roughly corresponds to the sum of the ***D*** plus the ***Hd*** (Head movement) in VR. The explanation is that dragging is used to rotate the camera in the desktop system while moving the head to accomplish the same action in VR. Another example of different operators performing the same task is the Controller Analogue Stick operator ***S*** and the keystroke ***K***. The former moves the VR camera in any direction performing only one action, whereas the latter is used multiple times in the traditional interface depending on the number of times users change the direction of the camera.

The timings shown in Table 6.2 were obtained after an expert user completed the scenarios. Only the first scenario was used to demonstrate the difference in setup time when the keyboard is not used. For the desktop interface, the difference was only 4 seconds, from 63s to 59s. In

contrast, VR setup time decreased from 93s to 72s, showing that more than 20% of the time is spent in naming the different simulation elements. This information suggests that the virtual keyboard may not be the best text input approach.

The selection of the text input device must consider three main aspects: accuracy, speed and comfort [3]. While the virtual keyboard provides accuracy – the key buttons are big, clear and separated from each other to prevent possible mistakes – and comfort – the user does not have to take off the headset to use a physical keyboard – it does not fulfil the speed requirement. Therefore, alternative text input methods could be considered, for example, a device tracking the position of all the fingers to simulate typing on a real keyboard or voice recognition. Alternatively, the same device could be used but the input approach could be changed. Examples could be sketching letters in the air, changing the layout of the keyboard or using a swipe keyboard. All these alternatives have their drawbacks. Using an additional finger tracking device would not be comfortable since two devices would be needed. Voice recognition and the swipe keyboard are not as accurate as a standard keyboard. Sketching letters and changing the keyboard layout would still be slow. A possible solution might be to reduce the need for typing to a minimum. Generic names could be assigned by default to the elements created. For example, Exit1, Exit2 ... ExitN. In this way, text input would only be used for parameters with numerical values, and a numeric keypad could replace the keyboard. The disadvantage of this solution is the lack of customisation and the possible confusion of names and elements in extensive simulations.

		Operators								Setup time
Scenario	Interface	K	VS	P	H	D	S	Hd	A	(s)
1	Desktop	64	16	26	5	4	-	-	-	63
1	Desktop No Key	40	16	26	0	4	-	-	-	59
1	VR	76	46	52	-	1	3	3	28	93
1	VR No Key	46	23	29	-	1	3	3	28	72
2	Desktop	173	38	62	21	6	-	-	-	147
2	Desktop No Key	92	38	62	0	6	-	-	-	-
2	VR	250	155	173	-	0	6	6	82	204
2	VR No Key	139	68	86	-	0	6	6	82	-
3	Desktop	175	43	78	13	12	-	-	-	141
3	Desktop No Key	120	43	78	0	12	-	-	-	-
3	VR	209	126	149	-	3	8	8	72	234
3	VR No Key	135	72	92	-	3	8	8	72	-

Table 6.2: Operators needed to complete the three scenarios using both interfaces.

6.2.3 Using KLM for VR

The lack of research done on Virtual Reality systems and KLM could be due to two reasons. First, VR systems are relatively new compared to traditional systems. Second, KLM might not be a suitable approach to evaluate the performance of these systems. Several problems were

found during this research when trying to adapt the original KLM approach to meet the needs of the VR system.

The proposed operators are specific to the interface. For instance, ***A*** (Arm movement) might not be necessary in other VR systems. This operator was included since the user can move the position of the menu, but this could be in a fixed position. Another example is the ***S*** (Controller Analogue Stick) operator, which would not be present if other input devices, such as the *Leap Motion* controller or a treadmill were used to move the camera instead of the *Oculus Touch* controllers.

The total number of low-level operations might be difficult to count. Operators such as ***Hd*** (Head movement) could be performed by the user when it is not required to complete the task. For example, the user might want to explore other areas, or the crowd attracts the user's attention causing an unnecessary head motion. The problem is to decide if these actions should be counted towards the total or be ignored, producing inaccurate results. The ***A*** operator (Arm movement) presents the same issue. The user might unintentionally move the arm affecting the final count.

Establishing a default time for the operators is complicated. Generally, KLM is used to predict the time needed to complete a task. A default execution time is assigned to every operator to calculate the total time. This might be easier for the original KLM operators such as ***K***, ***P*** and ***H***. Nevertheless, introducing new operators for complex actions complicates defining an average time for each operator since it depends on many factors. Using ***S*** as an example, this action is used to move the camera in the VR system. The duration of this operation depends on distance, analogue stick sensitivity, camera speed and how far the user pushes the stick. All these variables impede the calculation of accurate average execution time for this operator.

Finally, KLM operators might not be suitable to compare two VR systems. This point is related to the first one, having specific operators to the system being evaluated. In this research, a desktop and a VR interface are compared using the KLM results. However, only four operators could be directly compared since they were the only operators present in both systems. A similar issue would arise when comparing two VR systems since the adapted operators depend on the input devices used and the design of the user interface. It is likely that the more similar the systems are, the more meaningful the comparison will be.

6.3 Comparing desktop and VR interfaces

This section analyses the advantages and disadvantages of the use of a traditional desktop interface and VR equipment to control a crowd simulation. A comparison is made in terms of sketching, camera control, system usability and interface complexity.

This section discusses the results of using sketching in an immersive environment in comparison to the traditional interface, and the trade-off between accuracy and speed when sketching in both the desktop and the VR interface. Overall, the desktop system allows the use of more sophisticated user interfaces with more features to control the simulation. Having a 2D fixed menu, the mouse and the keyboard make the interface more intuitive for the user and gives the ability to configure parameters in a more intuitive and faster manner. The main issue of

the VR system is to find an alternative to the physical keyboard and trying to include the same functionality in small menu interfaces without increasing their complexity. An example is the Timeline feature which could not be implemented for the VR system. However, the immersive environment offers a more in-depth insight into how the crowd reacts to the user inputs.

6.3.1 Sketching

Sketching in an immersive environment requires a new input device since it cannot be done with a mouse. The introduction of this device produces some differences in the final sketch. Arora et al. [8] explored the factors affecting sketching in a VR environment with a couple of experiments. However, no direct comparison between a traditional computer interface and the VR environment was made. One of the experiments compared sketching mi-air with sketching supported by a physical interface. The results showed more accurate strokes when the surface was used. Similar findings can be expected when comparing the speed and accuracy of sketching with a mouse and VR equipment. A simple test is made to evaluate sketching in both interfaces. The aim of the experiment is to compare sketching in general, not specifically to control a crowd simulation.

The experiment consists of sketching three basic shapes: circle, triangle and square. These shapes were selected since they are easy to draw and could be similar to the areas sketched to control the crowd simulation. First, the figures are drawn at a reasonable speed without focusing on strictly following a template line. Second, the sketches are done as accurately as possible following the perimeter of the shapes. Both experiments are repeated multiple times using a mouse and in the VR environment with the *Oculus Touch* controller. A score and time are assigned to each sketch to show the trade-off between sketch speed and accuracy. The score is obtained using the same multi-stroke recogniser [6] as in Section 4.5.

The first test consists of sketching the shapes six times without paying extra attention to accuracy. These drawings are done with a mouse and the VR controller. Figure 6.11 shows the expert user sketches overlapped, the average scores and the timings. The results show wobbly strokes in the VR environment which are reflected in a lower score than the mouse sketch. Moreover, drawing the figures was nearly twice as fast in the desktop interface. These findings might be explained by the fact that generally, users are familiar with the use of a mouse. Closer results may be obtained after getting comfortable with the VR controller.

The second experiment consists of drawing the same shapes three times as accurately as possible with both input devices. Time is not a concern for this experiment. Figure 6.12 shows the result of each individual sketch. A similar pattern can be observed. Sketching with a mouse produced better scores in less time than with a VR controller. In general, the drawings obtained a higher score than the average of the speed test. The only exception is the first circle of the VR interface. However, this accuracy improvement is at the expense of speed. Accurate mouse sketching is at least five times slower than regular mouse sketching. Similarly, accurate VR sketching is four times slower than normal drawing.

Mouse-based sketching was faster and more accurate than VR sketching. A comprehensive user study would be required to corroborate these results. The better performance of traditional mouse-based sketching might be explained by the user's familiarity using a mouse and the

Figure 6.11: Mouse and VR Controller sketch speed test. First row shows six sketches overlapped for each shape and their corresponding average score and time. The second row shows the results of the VR controller.

inexperience with VR controllers. Although the VR environment proved not to be the most suitable for sketching, speed and accuracy are not essential for sketching barriers and areas. Other applications may require precise strokes, therefore, an immersive environment would not be a viable option. Wobbly strokes can be improved by sampling the input line into a fixed number of points. Sampling would help to reduce the noise of the sketch. An alternative would be to recognise the geometric shape intended by the user and fit the stroke to a parameterised curve representing the shape.

6.3.2 Camera control

The attention required for the implementation of camera control is entirely different for both interfaces. The camera can be moved freely in the desktop interface, and users can adjust its speeds. This freedom of movement is not possible in a VR environment due to adverse health effects such as motion sickness. This ailment can be caused by scene motion and self-motion illusion [70]. Multiple considerations were followed to implement the VR camera. First, the speed of the top-view camera can be adjusted, but it is considerably slower compared to the desktop interface. Second, Pedestrian mode camera can only be moved on the simulation plane. Third, the speed of this camera is fixed to the average speed of the crowd. Last, the camera accelerates instantly from 0 to the fixed speed to avoid slow transitions. This careful implementation allows the VR camera to provide an immersive environment to the user and the illusion of being part of the crowd. This experience offers a more in-depth insight into the crowd behaviour and its reaction to real-time modifications of the environment.

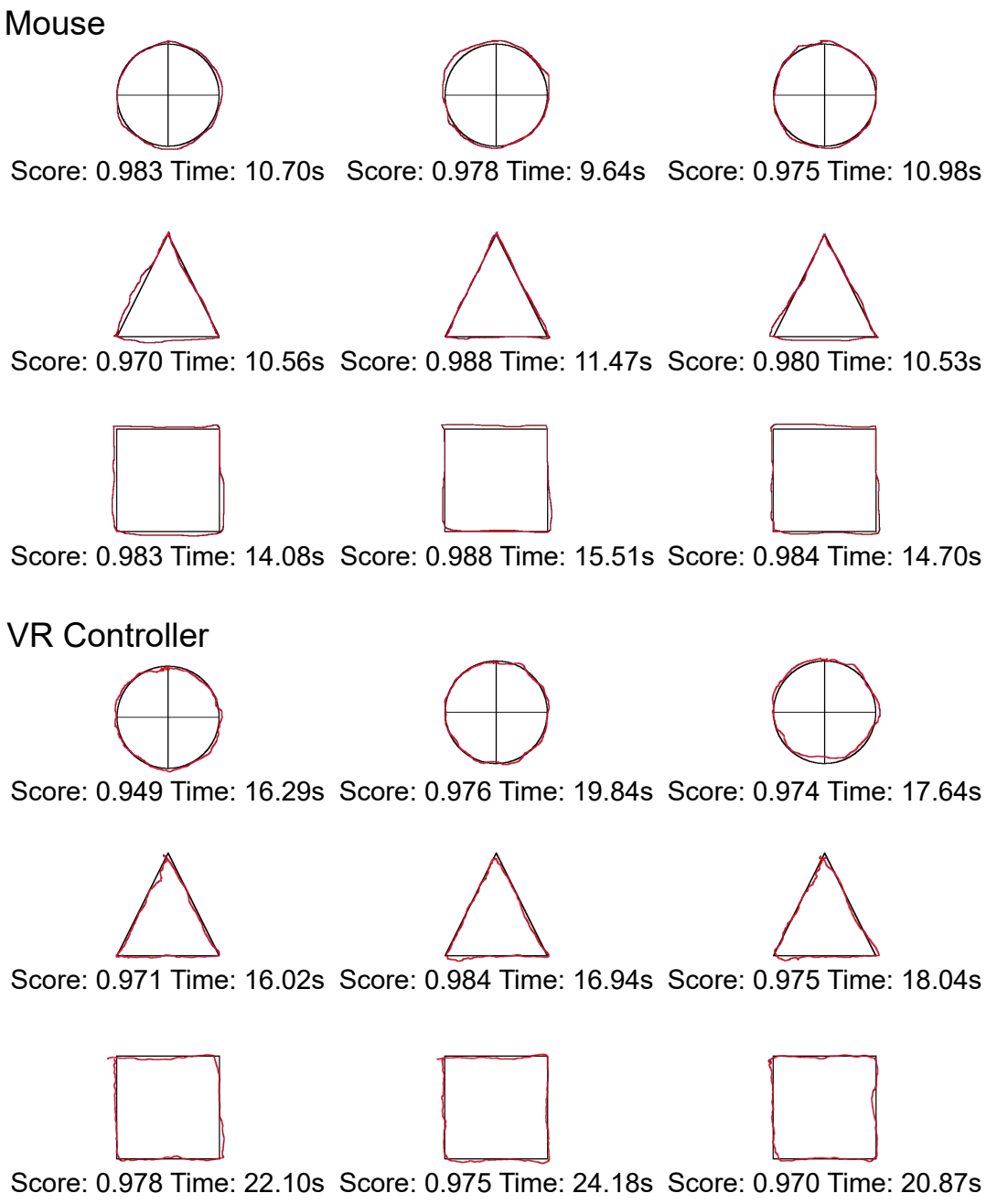


Figure 6.12: Mouse and VR Controller sketch accuracy test. Three shapes are sketched three times per input device. The first three rows show the scores and timings of the mouse. Last three rows show the VR controller results.

6.3.3 Interface complexity

The last aspect to consider is the complexity of the interface. The same user interface cannot be used for both systems since the desktop uses a 2D onscreen menu, whereas the VR requires a 3D

interface part of the world. This new 3D menu limits the number of features that can be included. Despite this limitation, the 2D menu was tried to be translated to the 3D version to maintain familiar functionality. An important feature not added to the VR interface is the timeline. The main disadvantage of the VR equipment was the impossibility of using the standard keyboard while wearing the headset. Therefore, a text input alternative was implemented. A virtual keyboard was included in the VR interface where users have to find, point to and select every key. This slow process does not meet one of the VR text input objectives: speed, comfort and accuracy [3]. Entering the element names and parameters was one of the most time-consuming tasks of the scenarios created in the user study (Section 5.5). Possible solutions to this problem are: using default generic names to reduce the amount of text to enter, changing the layout of the VR keyboard, implementing a swipe keyboard or using alternative input methods such as voice recognition and hand-tracking devices to simulate typing on a real keyboard. In general, the desktop interface supports more complex features and permits the use of a physical keyboard.

6.4 Other devices

Other input devices were briefly considered to assess their feasibility to sketch and use the graphical interface. A detailed comparison with experiments against the mouse input was not considered. The first device is a ten-point touch-screen monitor. This device allows the drawing of more precise and smoother curves compared to a mouse. The monitor can also be used as a tablet and the result displayed on another screen. A drawback of using this input method is the difficulty to use the menu. Controlling the camera would require using gestures like two fingers. Figure 6.13 shows the use of the touch-screen monitor. The best solution would be to use the touch feature only for sketching but still use mouse and keyboard for the interface.

Figure 6.13: Touch-screen monitor used as sketching input.

The other device considered was a Wacom Cintiq tablet (Figure 6.14). One issue with this tablet is the lack of a screen. Sketching in the desired position is challenging since the screen is mapped to the tablet, but there is no way to know the exact position of the pen. Using the menu presents the same issue: it is difficult to press the right button without practice. The tablet is not suitable because the goal of the graphical interface is to be intuitive for non-expert users. Moving the camera would require a button to change modes between sketching and camera control. A graphic tablet with a screen would be a better alternative for sketching but would still present the camera control problem.

Figure 6.14: Wacom cintiq

6.5 Summary

In summary, a VR environment is suitable to control a crowd simulation via sketching. It offers a more in-depth insight into how the crowd reacts in real-time to the modifications made by the user. However, it presents some drawbacks. First, sketching with a VR controller is slower and less accurate than using a mouse. Nevertheless, this does not pose an obstacle to the proposed system since time and accuracy are not a concern during the creation of elements. Second, a 3D menu interface part of the world cannot include complex features since its size is limited. Third, inappropriate input devices might slow down the setup process. Last, completing a task in VR requires more actions and time compared to a desktop interface.

Chapter 7

Conclusions

This research has investigated the use of sketching to graphically control crowd simulations. The idea is to sketch lines to control the crowd, in real-time, indirectly by modifying the environment, or directly by selecting a group and defining paths and behaviours. A data structure is required to represent the environment and to navigate the crowd through it. Two navigation methods, grid and navmesh, were implemented and adapted to support sketching on top of them. A comparison was made to determine the most scalable approach (Chapter 3). Multiple crowd control options were implemented using a navmesh-based simulation: barriers, flow lines, areas, storyboards, events and group control (Chapters 3 and 4). Some solutions to the issue of dynamic environment knowledge agents due to real-time environment modifications were also proposed (Chapter 4). The sketching system was evaluated by conducting a user study to compare the validated commercial system MassMotion with the sketch-based simulation (Chapter 5). The sketching approach was also implemented in a VR environment to assess its feasibility in comparison to the desktop interface (Chapter 6).

This thesis has investigated and compared two data structures: grid and navmesh. These are two of the most common approaches to represent the environment. Grids are easily implemented and it is a useful structure to create vector fields to navigate the crowd [145, 26, 132, 90]. Navmeshes are widely used in video games and game engines such as *Unity* and *Unreal Engine*. The implementation of the grid-based approach was described in Chapter 3 – this work is related to Patil *et al*'s work [132], in that flow lines can be sketched to direct pedestrians. However, three additional features are offered. First, obstacles can be created, cut and deleted by sketching in real-time during the simulation in order to direct the crowd. Second, the user can specify the pedestrian spawn locations and goals by sketching in the scene before the simulation starts (similar to Oshita and Oqiwara's work [129]). Third, multiple navigational layers are used to guide agents. The second data structure used to represent the environment is the navmesh; its core implementation is explained in Chapter 3. The main contribution of this approach is the use of sketching to update a navmesh in real-time. The navmesh is modified by sketching elements such as barriers, flow lines and areas.

Both navigation approaches were compared in terms of environment representation, sketch accuracy, memory usage and path-finding time. The navmesh is a better alternative than the grid since it represents the environment more accurately and requires less memory than the

grid-based approach, although computation time is similar for both approaches. However, the navmesh approach has some limitations. The initial construction time (which can be done in an offline step) and the addition of new elements increases the number of polygons and consequently the path-finding computation time. This comparison answers the second research question as a navmesh is the best data structure to support a crowd simulation and the sketch-based approach.

The sketch-based approach was extended to add more complex control features. Furthermore, a new problem of dynamically modifying the environment was identified. This work produced three more contributions (Chapter 4). First, storyboards can be created to specify a crowd's journey. Second, a timeline interface was presented to simulate events throughout the day. This work was based on the a future work idea proposed in [76] of changing crowd attributes such as direction and density to transition between hours of the day. Third, a sketch-driven group storyboard was presented. Users can select a group of agents and link paths and behaviours to create the storyboard of the group. A similar future work idea was proposed by Hughes et al. [66], where events could be linked in a sequence to define the story of the crowd. These extensions to sketching offer more options to create complex scenarios. Two examples of these scenarios and possible applications were presented in Section 4.3. The new control features require more sophisticated interfaces, which could have the effect of reducing the ease of use of the overall system. However, we could conclude that this hybrid approach is a good mix of the intuitiveness of the sketching approach and the flexibility of traditional interfaces since it provides similar options to commercial systems while being faster and easier to use.

Whilst undertaking the real-time sketching work, an issue was identified that does not seem to have been covered in other crowd simulation work: 'when should a pedestrian become aware of a change to the environment?'. In existing simulation models, since environments are set up in advance of the simulation running, agents then possess complete knowledge of the environment and automatically follow the shortest path to their destination. This immediate knowledge produces unrealistic behaviours when the environment is modified in real-time. Previous research has proposed methods to represent dynamic environments, but no discussion about dynamic environment knowledge was found. This thesis has considered three types of dynamic knowledge. The first option was that the crowd reacts immediately to user inputs. This type of knowledge produces unrealistic behaviour since agents adjust their paths even when they have not 'seen' the change in the environment. The second option considered was that the crowd should react when they 'see' the new element added. This solution produces a more realistic behaviour, than the immediate knowledge, since a real-world crowd do not become aware of any sudden events until it is close enough to react and adjust its path. The third option considered was a vision approach combined with time-based knowledge. The crowd react to the dynamic updates when they 'see' them or after some time when they are notified of the modification. This last knowledge considers the possibility of information being propagated in different ways (e.g. information board, internet, radio). The best type of knowledge in terms of memory usage and path-finding time is immediate knowledge since it only requires one navmesh for all the crowd. The approach that produces the most plausible simulation is vision plus time. However, the number of navmeshes required increases exponentially depending

on the number of obstacles. This work partially answers the third research question since the dynamic environment knowledge is a new problem and more research is needed to understand its implications for simulations.

Sketching as a graphical control tool proved to be intuitive, easy to use and faster to set up simulations than traditional approaches used in commercial systems. Chapter 5 describes the user study carried out to compare the sketching control with the validated commercial system MassMotion. The videos recorded were analysed using an adapted KLM version (supporting sketching) to count the number of actions needed to complete each task. Participants completed every scenario of the sketching experiment by doing less than half of the number of tasks performed in MassMotion. Moreover, the completion time of every scenario was recorded to assess the efficiency of both systems. Similar to the number of actions, participants took more than twice the time to complete the MassMotion experiment compared to the sketching control system. Some of the tasks, such as creating journeys/storyboards and barriers, were more difficult to achieve with the sketching approach. The main reasons were bad camera angle and lack of system feedback while creating a storyboard.

The user experience was assessed with the System Usability Scale. In general, participants had a better experience using the sketching system. They qualified the usability of this system as 'Excellent' and the MassMotion interface as 'OK-Good'. The simplicity of sketching lines is an intuitive way to control a simulation. However, it offers limited control options unless it is combined with other graphical modalities such as drag and drop, clicking and parameter input. Commercial systems offer far more options and flexibility to set up a simulation. Nevertheless, more parameters have to be tuned, complex interfaces are required, and users need prior knowledge of the system to use it.

One of the main aspects of the sketch-based approach is that real-time changes can be made during the simulation. A key advantage of updating the simulation in real-time is the ability to immediately see the effects of any modification. For instance, a traffic accident suddenly blocks a road. This event can only be simulated if the system allows real-time interaction since the underlying data structure can be updated without restarting the simulation. Traditional systems need to quit the simulation, update the environment and then start the simulation again. This behaviour is not realistic since agents restart the position and enter the simulation with knowledge of the blocked road. The user study answers the first research question proving that the sketch-based approach is intuitive and reduces the time of setting up a simulation.

The last aspect of the research that was investigated was whether or not a VR interface offered benefits. The VR interface *can* support sketching to control a crowd simulation. However, it presents some limitations that might complicate control. First, the interface implemented as a 3D object cannot include all the features due to its smaller size. Second, an incorrect input device might slow down the setup process. Third, sketching in a VR environment is slower and less accurate than using the desktop interface. Last, the VR interface requires more actions than the traditional interface to complete the same scenario. An adapted KLM version for VR was proposed to count the total tasks performed to complete the same three scenarios included in the user study (Chapter 5). The immersive environment offers a better perspective on how the crowd reacts in real-time to the dynamic simulation interaction. This work answers the fourth

research question.

This research has investigated features that can be created by sketching on top of the environment to modify the existing navmesh. However, this approach can be extended to create the environment itself. Examples would be defining the contour of a building, creating links or bridges to connect two separate navmeshes or using 3D sketching to create stairs/ramps to link multiple navmesh levels. The implementation of the navmesh could be improved in several ways. First, the number of polygons may be reduced with a better triangulation algorithm. Second, a hierarchical path-finding algorithm similar to the one proposed by Pelechano and Fuentes [133] could be implemented. The search could be divided into two levels: tiles and polygons. The path of connected tiles that leads to the goal could be found. This information would include the path of polygons within each tile, and the tile boundary joins. Thus, when the navmesh of a tile is updated, the path of polygons within the tile would be updated while still matching the boundary joins with abutting tiles. Third, the path-finding algorithm can be accelerated by using the GPU. This improvement would allow changing from calculating the shortest path of each polygon to have each agent calculate this path from their position. Each agent would calculate their path when they enter the simulation or after every dynamic update.

The problem of dynamic environment knowledge was briefly covered with some simple solutions. This subject needs further research, for example on other types of knowledge, different approaches to discover obstacles and other data structures to represent the environment. Similarly, the VR implementation of sketching control has some areas of possible future work. More input devices could be explored to find the best alternative to support sketching and text input. Moreover, a different interface layout might facilitate the inclusion of all the features present in the desktop interface.

References

[1] Akaydın, A. and Güdükbay, U. Adaptive grids: an image-based approach to generate navigation meshes. *Optical Engineering*, 52(2):027002–027002, 2013.

[2] Al-Megren, S., Khabti, J., and Al-Khalifa, H. S. A systematic review of modifications and validation methods for the extension of the keystroke-level model. *Advances in Human-Computer Interaction*, 2018, 2018.

[3] Alger, M. Visual design methods for virtual reality. *Ravensbourne*, 2015. http://aperturesciencellc.com/vr/VisualDesignMethodsforVR_MikeAlger.pdf.

[4] Allen, T., Parvanov, A., Knight, S., and Maddock, S. Using Sketching to Control Heterogeneous Groups. In Borgo, R. and Turkay, C., editors, *Computer Graphics and Visual Computing (CGVC)*. The Eurographics Association, 2015.

[5] Anderson, M., McDaniel, E., and Chenney, S. Constrained animation of flocks. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 286–297, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[6] Anthony, L. and Wobbrock, J. O. A Lightweight Multistroke Recognizer for User Interface Prototypes. In *Proceedings of Graphics Interface 2010*, GI '10, pages 245–252, Toronto, Ont., Canada, Canada, 2010. Canadian Information Processing Society.

[7] Arnaldi, B., Guitton, P., and Moreau, G. *Virtual Reality and Augmented Reality: Myths and Realities*. John Wiley & Sons, Incorporated, Newark, United States, 2018.

[8] Arora, R., Kazi, R. H., Anderson, F., Grossman, T., Singh, K., and Fitzmaurice, G. Experimental Evaluation of Sketching on Surfaces in VR. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 5643–5654, New York, NY, USA, 2017. ACM. event-place: Denver, Colorado, USA.

[9] Banerjee, B., Abukmail, A., and Kraemer, L. Advancing the layered approach to agent-based crowd simulation. In *2008 22nd Workshop on Principles of Advanced and Distributed Simulation*, pages 185–192, June 2008.

[10] Bangor, A., Kortum, P. T., and Miller, J. T. An Empirical Evaluation of the System Usability Scale. *International Journal of Human-Computer Interaction*, 24(6):574–594, July 2008.

[11] Bangor, A., Kortum, P., and Miller, J. Determining what individual sus scores mean: Adding an adjective rating scale. *Journal of usability studies*, 4(3):114–123, 2009.

[12] Barnett, A., Shum, H. P. H., and Komura, T. Coordinated crowd simulation with topological scene analysis. *Computer Graphics Forum*, 35(6):120–132, September 2016.

[13] Bayazit, O. B., Lien, J.-M., and Amato, N. M. Roadmap-based flocking for complex environments. In *10th Pacific Conference on Computer Graphics and Applications, 2002. Proceedings.*, pages 104–113, Oct 2002.

[14] Bayazit, O. B., ming Lien, J., and Amato, N. M. Better group behaviors in complex environments using global roadmaps. In *In Artif. Life*, pages 362–370, 2002.

[15] Berseth, G., Kapadia, M., and Faloutsos, P. ACCLMesh: curvature-based navigation mesh generation: Curvature-based navigation mesh generation. *Computer Animation and Virtual Worlds*, 27(3-4):195–204, May 2016.

[16] Braun, A., Musse, S. R., de Oliveira, L. P. L., and Bodmann, B. E. J. Modeling individual behaviors in crowd simulation. In *Proceedings 11th IEEE International Workshop on Program Comprehension*, pages 143–148, May 2003.

[17] Brooke, J. SUS: a retrospective. *Journal of Usability Studies*, 8:29–40, January 2013.

[18] Brooke, J. et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[19] Campanella, M., Hoogendoorn, S., and Daamen, W. Quantitative and Qualitative Validation Procedure for General Use of Pedestrian Models. In Weidmann, U., Kirsch, U., and Schreckenberg, M., editors, *Pedestrian and Evacuation Dynamics 2012*, pages 891–905. Springer International Publishing, 2014.

[20] Card, S. K., Moran, T. P., and Newell, A. The Keystroke-level Model for User Performance Time with Interactive Systems. *Commun. ACM*, 23(7):396–410, July 1980.

[21] Card, S. K., Newell, A., and Moran, T. P. *The Psychology of Human-Computer Interaction*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 1983.

[22] Challenger, W., Clegg, W., and Robinson, A. Understanding crowd behaviours: Guidance and lessons identified. *UK Cabinet Office*, 2009.

[23] Charlton, J., Gonzalez, L. R. M., Maddock, S., and Richmond, P. Fast Simulation of Crowd Collision Avoidance. In Gavrilova, M., Chang, J., Thalmann, N. M., Hitzer, E., and Ishikawa, H., editors, *Advances in Computer Graphics*, Lecture Notes in Computer Science, pages 266–277. Springer International Publishing, 2019.

[24] Charlton, J., Gonzalez, L. R. M., Maddock, S., and Richmond, P. Simulating crowds and autonomous vehicles. In *Transactions on Computational Science XXXVII*, pages 129–143. Springer, 2020.

[25] Chattaraj, U., Seyfried, A., and Chakroborty, P. Comparison of pedestrian fundamental diagram across cultures. *Advances in complex systems*, 12(03):393–405, 2009.

[26] Chenney, S. Flow Tiles. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA'04, pages 233–242, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association.

[27] Chew, L. P. Constrained delaunay triangulations. In *Proceedings of the Third Annual Symposium on Computational Geometry*, SCG '87, page 215222, New York, NY, USA, 1987. Association for Computing Machinery.

[28] Coburn, J. Q., Freeman, I., and Salmon, J. L. A Review of the Capabilities of Current Low-Cost Virtual Reality Technology and Its Potential to Enhance the Design Process. *Journal of Computing and Information Science in Engineering*, 17(3), 07 2017.

[29] Cui, X. and Shi, H. An overview of pathfinding in navigation mesh. *International Journal of Computer Science and Network Security*, 12(12):48–51, December 2012.

[30] Curtis, S., Snape, J., and Manocha, D. Way portals: Efficient multi-agent navigation with line-segment goals. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '12, pages 15–22, New York, NY, USA, 2012. ACM.

[31] Demyen, D. and Buro, M. Efficient triangulation-based pathfinding. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, AAAI'06, pages 942–947. AAAI Press, 2006.

[32] Duives, D. C., Daamen, W., and Hoogendoorn, S. P. State-of-the-art crowd motion simulation models. *Transportation Research Part C: Emerging Technologies*, 37:193–209, December 2013.

[33] Durupinar, F., Allbeck, J., Pelechano, N., and Badler, N. Creating Crowd Variation with the OCEAN Personality Model. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 3*, AAMAS '08, pages 1217–1220, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems. event-place: Estoril, Portugal.

[34] Ecormier-Nocca, P., Pettré, J., Memari, P., and Cani, M.-P. Image-based authoring of herd animations. *Computer Animation and Virtual Worlds*, 30(3-4):e1903, 2019.

[35] Farenc, N., Boulic, R., and Thalmann, D. An Informed Environment Dedicated to the Simulation of Virtual Humans in Urban Context. *Computer Graphics Forum*, 18(3): 309–318, September 1999.

[36] Fiorini, P. and Shiller, Z. Motion Planning in Dynamic Environments Using Velocity Obstacles. *The International Journal of Robotics Research*, 17(7):760–772, July 1998.

[37] Geraerts, R. Planning short paths with clearance using explicit corridors. In *2010 IEEE International Conference on Robotics and Automation*, pages 1997–2004, May 2010.

[38] Geraerts, R. and Overmars, M. H. Creating High-quality Roadmaps for Motion Planning in Virtual Environments. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4355–4361, October 2006.

[39] Geraerts, R. J. and Overmars, M. H. Enhancing corridor maps for real-time path planning in virtual environments. In *Computer Animation and Social Agents*, pages 64–71, 2008.

[40] Geraerts, R. and Overmars, M. H. The corridor map method: A general framework for real-time high-quality path planning: Research articles. *Computer Animation and Virtual Worlds*, 18(2):107–119, May 2007.

[41] Gillan, D. J., Holden, K., Adam, S., Rudisill, M., and Magee, L. How Does Fitts' Law Fit Pointing and Dragging? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '90, pages 227–234, New York, NY, USA, 1990. ACM. event-place: Seattle, Washington, USA.

[42] Goldenstein, S., Karavelas, M., Metaxas, D., Guibas, L., Aaron, E., and Goswami, A. Scalable nonlinear dynamical systems for agent steering and crowd simulation. *Computers & Graphics*, 25(6):983 – 998, 2001. Artificial Life.

[43] Gonzalez, L. R. M. and Maddock, S. Sketching for Real-time Control of Crowd Simulations. In *Computer Graphics and Visual Computing (CGVC)*. The Eurographics Association, 2017. DOI: 10.2312/cgvc.20171282.

[44] Gonzalez, L. R. M. and Maddock, S. A sketch-based interface for real-time control of crowd simulations that use navigation meshes. In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 1: GRAPP*, pages 41–52. INSTICC, SciTePress, 2019.

[45] Gu, Q. and Deng, Z. Context-aware motion diversification for crowd simulation. *IEEE Computer Graphics and Applications*, 31(5):54–65, Sept 2011.

[46] Gu, Q. and Deng, Z. Generating freestyle group formations in agent-based crowd simulations. *IEEE Computer Graphics and Applications*, 33(1):20–31, Jan 2013.

[47] Gu, Q. and Deng, Z. Formation sketching: An approach to stylize groups in crowd simulation. In *Proceedings of Graphics Interface 2011*, GI '11, pages 1–8, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2011. Canadian Human-Computer Communications Society.

[48] Gunnarsson, O. and Maddock, S. Sketching Faces. In Alvarado, C. and Cani, M.-P., editors, *Eurographics Workshop on Sketch-Based Interfaces and Modeling*. The Eurographics Association, 2008.

[49] Guy, S. J., Chhugani, J., Curtis, S., Dubey, P., Lin, M., and Manocha, D. Pledestrians: A least-effort approach to crowd simulation. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pages 119–128, Goslar Germany, Germany, 2010. Eurographics Association.

[50] Guy, S. J., Kim, S., Lin, M. C., and Manocha, D. Simulating Heterogeneous Crowd Behaviors Using Personality Trait Theory. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '11, pages 43–52, New York, NY, USA, 2011. ACM.

[51] Guy, S. J., van den Berg, J., Liu, W., Lau, R., Lin, M. C., and Manocha, D. A Statistical Similarity Measure for Aggregate Crowd Dynamics. *ACM Trans. Graph.*, 31 (6):190:1–190:11, November 2012.

[52] Hale, D. H. and Youngblood, G. M. Full 3d spatial decomposition for the generation of navigation meshes. In *Proceedings of the Fifth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, AIIDE'09, pages 142–147. AAAI Press, 2009.

[53] Hale, D. H., Youngblood, G. M., and Dixit, P. N. Automatically-generated Convex Region Decomposition for Real-time Spatial Agent Navigation in Virtual Worlds. *AIIDE*, 8:173–8, 2008.

[54] Hauri, S., Alonso-Mora, J., Breitenmoser, A., Siegwart, R., and Beardsley, P. Multi-robot formation control via a real-time drawing interface. In *Field and Service Robotics*, pages 175–189. Springer, 2014.

[55] He, F., Xiang, Y., Zhao, X., and Wang, H. Informative scene decomposition for crowd analysis, comparison and simulation guidance. *ACM Trans. Graph.*, 39(4), July 2020.

[56] Helbing, D. A mathematical model for the behavior of pedestrians. *Behavioral Science*, 36(4):298–310, October 1991.

[57] Helbing, D. A Fluid-Dynamic Model for the Movement of Pedestrians. *Complex Systems*, 6(5):391–415, 1992.

[58] Helbing, D. and Molnar, P. Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282, 1995.

[59] Helbing, D., Farkas, I., and Vicsek, T. Simulating dynamical features of escape panic. *Nature*, 407(6803):487–490, September 2000.

[60] Henry, J., Shum, H. P. H., and Komura, T. Environment-aware Real-time Crowd Control. In *Proceedings of the 11th ACM SIGGRAPH / Eurographics Conference on Computer Animation*, EUROSCA'12, pages 193–200, Aire-la-Ville, Switzerland, Switzerland, 2012. Eurographics Association.

[61] Ho, C. S., Nguyen, Q. H., Ong, Y.-S., and Chen, X. Autonomous multi-agents in flexible flock formation. In *Proceedings of the Third International Conference on Motion in Games*, MIG'10, pages 375–385, Berlin, Heidelberg, 2010. Springer-Verlag.

[62] Hoang, D. UX/UI Design for VR, Part. 1. *AR/VR Journey: Augmented & Virtual Reality Magazine*, January 2019. https://arvrjourney.com/ux-ui-design-for-vr-part-1-fccc79f7f3fa.

[63] Holleis, P., Otto, F., Hussmann, H., and Schmidt, A. Keystroke-level model for advanced mobile phone interaction. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '07*, page 1505, San Jose, California, USA, 2007. ACM Press.

[64] Hughes, R. L. A continuum theory for the flow of pedestrians. *Transportation Research Part B: Methodological*, 36(6):507–535, 2002.

[65] Hughes, R. L. The Flow of Human Crowds. *Annual Review of Fluid Mechanics*, 35(1): 169–182, January 2003.

[66] Hughes, R., Ondřej, J., and Chaurasia, G. Sketch-Based Annotation and Authoring of Geometrically Sparse 3d Environments. Technical report, TCD-CS-2014-01, School of Computer Science and Statistics, Trinity College Dublin, 2014.

[67] Igarashi, T., Matsuoka, S., Kawachiya, S., and Tanaka, H. Interactive beautification: a technique for rapid geometric design. In *Proceedings of the 10th annual ACM symposium on User interface software and technology*, UIST '97, pages 105–114, New York, NY, USA, October 1997. Association for Computing Machinery.

[68] Igarashi, T., Matsuoka, S., and Tanaka, H. Teddy: A sketching interface for 3d freeform design. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, page 409416, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[69] ISO 25010:2011(EN). *Systems and software engineering Systems and software Quality Requirements and Evaluation (SQuaRE) System and software quality models*, 2011.

[70] Jerald, J. *The VR Book: Human-Centered Design for Virtual Reality.* Association for Computing Machinery and Morgan & Claypool, 2015.

[71] Jiang, H., Xu, W., Mao, T., Li, C., Xia, S., and Wang, Z. A semantic environment model for crowd simulation in multilayered complex environment. In *Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology*, VRST '09, pages 191–198, New York, NY, USA, 2009. ACM.

[72] Jiang, H., Xu, W., Mao, T., Li, C., Xia, S., and Wang, Z. Continuum crowd simulation in complex environments. *Computers & Graphics*, 34(5):537–544, October 2010.

[73] Jin, X., Xu, J., Wang, C. C., Huang, S., and Zhang, J. Interactive Control of Large-Crowd Navigation in Virtual Environments Using Vector Fields. *IEEE Computer Graphics and Applications*, 28(6):37–46, November 2008.

[74] John, B. E. and Kieras, D. E. The GOMS Family of User Interface Analysis Techniques: Comparison and Contrast. *ACM Trans. Comput.-Hum. Interact.*, 3(4):320–351, December 1996.

[75] Jordao, K., Pettré, J., Christie, M., and Cani, M.-P. Crowd sculpting: A space-time sculpting method for populating virtual environments. *Comput. Graph. Forum*, 33(2): 351–360, May 2014.

[76] Jordao, K., Charalambous, P., Christie, M., Pettré, J., and Cani, M.-P. Crowd art: Density and flow based crowd motion design. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*, MIG '15, pages 167–176, New York, NY, USA, 2015. ACM.

[77] Jorgensen, C.-J. and Lamarche, F. From geometry to spatial reasoning: Automatic structuring of 3d virtual environments. In *Proceedings of the 4th International Conference on Motion in Games*, MIG'11, pages 353–364, Berlin, Heidelberg, 2011. Springer-Verlag.

[78] Kallmann, M. Path planning in triangulations. In *Proceedings of the IJCAI workshop on reasoning, representation, and learning in computer games*, pages 49–54, 2005.

[79] Kallmann, M. Shortest paths with arbitrary clearance from navigation meshes. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pages 159–168, Goslar Germany, Germany, 2010. Eurographics Association.

[80] Kallmann, M. Dynamic and Robust Local Clearance Triangulations. *ACM Trans. Graph.*, 33(5):161:1–161:17, September 2014.

[81] Kallmann, M. Dynamic and robust local clearance triangulations. *ACM Trans. Graph.*, 33(5):161:1–161:17, September 2014.

[82] Kallmann, M. and Kapadia, M. Navigation meshes and real-time dynamic planning for virtual worlds. In *ACM SIGGRAPH 2014 Courses*, SIGGRAPH '14, pages 3:1–3:81, New York, NY, USA, 2014. ACM.

[83] Kallmann, M., Bieri, H., and Thalmann, D. Fully Dynamic Constrained Delaunay Triangulations. *In Geometric Modelling for Scientific Visualization, G. Brunnett, B. Hamann, H. Mueller (Eds.), Springer-Verlag, 2003*, 2003.

[84] Kapadia, M., Singh, S., Hewlett, W., and Faloutsos, P. Egocentric affordance fields in pedestrian steering. In *Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games*, I3D '09, pages 215–223, New York, NY, USA, 2009. ACM.

[85] Kapadia, M., Wang, M., Singh, S., Reinman, G., and Faloutsos, P. Scenario Space: Characterizing Coverage, Quality, and Failure of Steering Algorithms. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '11, pages 53–62, New York, NY, USA, 2011. ACM.

[86] Kapadia, M., Shoulson, A., Boatright, C. D., Huang, P., Durupinar, F., and Badler, N. I. Whats next? the new era of autonomous virtual humans. In *International Conference on Motion in Games*, pages 170–181. Springer, 2012.

[87] Kapadia, M., Pelechano, N., Allbeck, J., and Badler, N. Virtual crowds: Steps toward behavioral realism. *Synthesis Lectures on Visual Computing: Computer Graphics, Animation, Computational Photography, and Imaging*, 7(4):1–270, 2015.

[88] Kara, L. B., Shimada, K., and Marmalefsky, S. D. An evaluation of user experience with a sketch-based 3d modeling system. *Computers & Graphics*, 31(4):580–597, August 2007.

[89] Karamouzas, I. and Overmars, M. Simulating and Evaluating the Local Behavior of Small Pedestrian Groups. *IEEE Transactions on Visualization and Computer Graphics*, 18(3): 394–406, March 2012.

[90] Karmakharm, T., Richmond, P., and Romano, D. M. Agent-based Large Scale Simulation of Pedestrians With Adaptive Realistic Navigation Vector Fields. *TPCG*, 10:67–74, 2010.

[91] Kieras, D. A guide to goms model usability evaluation using ngomsl, the handbook of human-computer interaction, 1996.

[92] Kim, J., Seol, Y., Kwon, T., and Lee, J. Interactive Manipulation of Large-scale Crowd Animation. *ACM Trans. Graph.*, 33(4):83:1–83:10, July 2014.

[93] Kim, M., Hyun, K., Kim, J., and Lee, J. Synchronized multi-character motion editing. *ACM Trans. Graph.*, 28(3):79:1–79:9, July 2009.

[94] Kim, M., Hwang, Y., Hyun, K., and Lee, J. Tiling motion patches. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, pages 117–126, Goslar Germany, Germany, 2012. Eurographics Association.

[95] Kim, S., Bera, A., Best, A., Chabra, R., and Manocha, D. Interactive and adaptive data-driven crowd simulation. In *2016 IEEE Virtual Reality (VR)*, pages 29–38, 2016.

[96] Kwon, T., Lee, K. H., Lee, J., and Takahashi, S. Group Motion Editing. In *ACM SIGGRAPH 2008 Papers*, SIGGRAPH '08, pages 80:1–80:8, New York, NY, USA, 2008. ACM.

[97] Lakoba, T. I., Kaup, D. J., and Finkelstein, N. M. Modifications of the helbing-molnár-farkas-vicsek social force model for pedestrian evolution. *Simulation*, 81 (5):339–352, May 2005.

[98] Lamarche, F. TopoPlan: a topological path planner for real time human navigation under floor and ceiling constraints. *Computer Graphics Forum*, 28(2):649–658, April 2009.

[99] Lamarche, F. and Donikian, S. Crowd of Virtual Humans: a New Approach for Real Time Navigation in Complex and Structured Environments. *Computer Graphics Forum*, 23(3): 509–518, September 2004.

[100] Lee, J., Won, J., and Lee, J. Crowd simulation by deep reinforcement learning. In *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games*, MIG '18, New York, NY, USA, 2018. Association for Computing Machinery.

[101] Lee, K. H., Choi, M. G., and Lee, J. Motion patches: Building blocks for virtual environments annotated with motion data. *ACM Trans. Graph.*, 25(3):898–906, July 2006.

[102] Lee, K. H., Choi, M. G., Hong, Q., and Lee, J. Group behavior from video: A data-driven approach to crowd simulation. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '07, page 109118, Goslar, DEU, 2007. Eurographics Association.

[103] Lee, S. C., Yoon, S. H., and Ji, Y. G. Modeling task completion time of in-vehicle information systems while driving with keystroke level modeling. *International Journal of Industrial Ergonomics*, 72:252–260, July 2019.

[104] Legion. Legion Software | Science in Motion. http://www.legion.com/legion-software, n.d.

[105] Lerner, A., Chrysanthou, Y., and Lischinski, D. Crowds by Example. *Computer Graphics Forum*, 2007.

[106] Lerner, A., Chrysanthou, Y., Shamir, A., and Cohen-Or, D. Data Driven Evaluation of Crowds. In Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J. M., Mattern, F., Mitchell, J. C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M. Y., Weikum, G., Egges, A., Geraerts, R., and Overmars, M., editors, *Motion in Games*, volume 5884, pages 75–83. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[107] Li, T.-Y., Jeng, Y.-J., and Chang, S.-I. Simulating virtual human crowds with a leader-follower model. In *Proceedings Computer Animation 2001. Fourteenth Conference on Computer Animation (Cat. No.01TH8596)*, pages 93–102, Nov 2001.

[108] Li, W., Di, Z., and Allbeck, J. M. Crowd distribution and location preference: Crowd distribution and location preference. *Computer Animation and Virtual Worlds*, 23(3-4): 343–351, May 2012.

[109] Loscos, C., Marchal, D., and Meyer, A. Intuitive crowd behaviour in dense urban environments using local laws. In *Proceedings of the Theory and Practice of Computer Graphics 2003*, TPCG '03, pages 122–129, Washington, DC, USA, 2003. IEEE Computer Society.

[110] Lu, G., Chen, L., and Luo, W. Real-time crowd simulation integrating potential fields and agent method. *ACM Trans. Model. Comput. Simul.*, 26(4):28:1–28:16, March 2016.

[111] MacKenzie, I. S., Sellen, A., and Buxton, W. A. S. A comparison of input devices in element pointing and dragging tasks. In *Proceedings of the SIGCHI conference on Human factors in computing systems Reaching through technology - CHI '91*, pages 161–166, New Orleans, Louisiana, United States, 1991. ACM Press.

[112] Martinez-Gil, F., Lozano, M., Garca-Fernndez, I., and Fernndez, F. Modeling, Evaluation, and Scale on Artificial Pedestrians: A Literature Review. *ACM Comput. Surv.*, 50(5): 72:1–72:35, September 2017.

[113] Massive. Massive Software Simulating Life. http://www.massivesoftware.com/, n.d.

[114] Mathew, T., Benes, B., and Aliaga, D. Interactive inverse spatio-temporal crowd motion design. In *Symposium on Interactive 3D Graphics and Games*, pages 1–9, 2020.

[115] McIlveen, J., Maddock, S., Heywood, P., and Richmond, P. PED: Pedestrian Environment Designer. In *Proc. CGVC 2016*, 2016.

[116] Millan, E. and Rudomin, I. Agent paint: Intuitive specification and control of multiagent animations. In *Proc. CASA*, 2005.

[117] Musse, S. R. and Thalmann, D. Hierarchical model for real time simulation of virtual human crowds. *IEEE Transactions on Visualization and Computer Graphics*, 7(2): 152–164, April 2001.

[118] Musse, S. R., Babski, C., Capin, T., and Thalmann, D. Crowd modelling in collaborative virtual environments. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 115–123. ACM, 1998.

[119] Musse, S. R., Cassol, V. J., and Jung, C. R. Towards a quantitative approach for comparing crowds. *Computer Animation and Virtual Worlds*, 23(1):49–57, February 2012.

[120] Musse, S. R. and Thalmann, D. A model of human crowd behavior: Group inter-relationship and collision detection analysis. In *Computer Animation and Simulation97*, pages 39–51. Springer, 1997.

[121] Narain, R., Golas, A., Curtis, S., and Lin, M. C. Aggregate dynamics for dense crowd simulation. *ACM transactions on graphics (TOG)*, 28(5):122:1–122:8, December 2009.

[122] Oasys. Oasys Software - MassMotion: Crowd Simulation and Pedestrian Modelling Software. http://www.oasys-software.com/products/engineering/massmotion.html, n.d.

[123] Oliva, R. and Pelechano, N. Automatic generation of suboptimal navmeshes. In *Proc MIG'11*, pages 328–339, Berlin, Heidelberg, 2011.

[124] Oliva, R. and Pelechano, N. NEOGEN: Near optimal generator of navigation meshes for 3d multi-layered environments. *Computers & Graphics*, 37(5):403–412, August 2013.

[125] Olivier, A.-H., Marin, A., Crtual, A., and Pettr, J. Minimal predicted distance: A common metric for collision avoidance during pairwise interactions between walkers. *Gait & Posture*, 36(3):399–404, July 2012.

[126] Olsen, L., Samavati, F. F., Sousa, M. C., and Jorge, J. A. Sketch-based modeling: A survey. *Computers & Graphics*, 33(1):85–103, February 2009.

[127] Ondřej, J., Pettré, J., Olivier, A.-H., and Donikian, S. A Synthetic-vision Based Steering Approach for Crowd Simulation. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, pages 123:1–123:9, New York, NY, USA, 2010. ACM. event-place: Los Angeles, California.

[128] OpenSteer. OpenSteer. http://opensteer.sourceforge.net/, n.d.

[129] Oshita, M. and Ogiwara, Y. Sketch-based interface for crowd animation. In *International Symposium on Smart Graphics*, pages 253–262. Springer, 2009.

[130] Pan, X., Han, C. S., Dauber, K., and Law, K. H. A multi-agent based framework for the simulation of human and social behaviors during emergency evacuations. *AI & SOCIETY*, 22(2):113–132, October 2007.

[131] Paris, S., Pettr, J., and Donikian, S. Pedestrian Reactive Navigation for Crowd Simulation: a Predictive Approach. *Computer Graphics Forum*, 26(3):665–674, September 2007.

[132] Patil, S., Van Den Berg, J., Curtis, S., Lin, M. C., and Manocha, D. Directing crowd simulations using navigation fields. *IEEE Transactions on Visualization and Computer Graphics*, 17(2):244–254, 2011.

[133] Pelechano, N. and Fuentes, C. Hierarchical path-finding for navigation meshes (hna*). *Comput. Graph.*, 59(C):68–78, October 2016.

[134] Pelechano, N., O'Brien, K., Silverman, B., and Badler, N. Crowd simulation incorporating agent psychological models, roles and communication. Technical report, Center for human modeling and simulation, Pennsylvania University, USA, 2005.

[135] Pelechano, N., Allbeck, J. M., and Badler, N. I. Controlling Individual Agents in High-density Crowd Simulation. In *Proc. SCA '07*, pages 99–108, 2007.

[136] Pelechano, N., Allbeck, J. M., and Badler, N. I. Virtual Crowds: Methods, Simulation, and Control. *Synthesis Lectures on Computer Graphics and Animation*, 3(1):1–176, January 2008.

[137] Peters, C. and Ennis, C. Modeling Groups of Plausible Virtual Pedestrians. *IEEE Computer Graphics and Applications*, 29(4):54–63, July 2009.

[138] Pettre, J., Laumond, J.-P., and Thalmann, D. A navigation graph for real-time crowd animation on multilayered and uneven terrain. In *First International Workshop on Crowd Simulation*, volume 43, page 194. New York: Pergamon Press, 2005.

[139] Pettr, J., Ciechomski, P. d. H., Mam, J., Yersin, B., Laumond, J.-P., and Thalmann, D. Real-time navigating crowds: scalable simulation and rendering. *Computer Animation and Virtual Worlds*, 17(3-4):445–455, July 2006.

[140] Pettr, J., Grillon, H., and Thalmann, D. Crowds of moving objects: Navigation planning and simulation. In *ACM SIGGRAPH 2008 classes*, page 54. ACM, 2008.

[141] Pettr, J., Ondej, J., Olivier, A.-H., Cretual, A., and Donikian, S. Experiment-based Modeling, Simulation and Validation of Interactions Between Virtual Walkers. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, pages 189–198, New York, NY, USA, 2009. ACM. event-place: New Orleans, Louisiana.

[142] Qiu, F. and Hu, X. Modeling group structures in pedestrian crowd simulation. *Simulation Modelling Practice and Theory*, 18(2):190–205, February 2010.

[143] Rao, Y., Chen, L., Liu, Q., Lin, W., Li, Y., and Zhou, J. Real-time control of individual agents for crowd simulation. *Multimedia Tools and Applications*, 54(2):397–414, August 2011.

[144] Reynolds, C. W. Flocks, herds and schools: A distributed behavioral model. *ACM SIGGRAPH computer graphics*, 21(4):25–34, 1987.

[145] Reynolds, C. W. Steering behaviors for autonomous characters. In *Game developers conference*, volume 1999, pages 763–782, 1999.

[146] Reynolds, C. W. Interaction with groups of autonomous characters. In *Game developers conference*, volume 2000, page 83, 2000.

[147] Ribicic, H., Waser, J., Gurbat, R., Sadransky, B., and Gröller, M. E. Sketching uncertainty into simulations. *IEEE Transactions on Visualization and Computer Graphics*, 18(12): 2255–2264, 2012.

[148] Richmond, P. and Romano, D. M. A high performance framework for agent based pedestrian dynamics on gpu hardware. *Proceedings of EUROSIS ESM*, 20:27–29, 2008.

[149] Rivers, E., Jaynes, C., Kimball, A., and Morrow, E. Using case study data to validate 3d agent-based pedestrian simulation tool for building egress modeling. *Transportation Research Procedia*, 2:123–131, 2014.

[150] Rodrigues, R. A., Lima Bicho, A., Paravisi, M., Jung, C. R., Magalhães, L. P., and Musse, S. R. Tree paths: A new model for steering behaviors. In *Proceedings of the 9th International Conference on Intelligent Virtual Agents*, IVA '09, pages 358–371, Berlin, Heidelberg, 2009. Springer-Verlag.

[151] Rudomin, I., Hernndez, B., De Gyves, O., Toledo, L., Rivalcoba, I., and Ruiz, S. Gpu generation of large varied animated crowds. *Computacin y Sistemas*, 17(3), 2013.

[152] Russell, S. J. and Norvig, P. *Artificial intelligence: a modern approach*. Pearson, Boston, 2020. OCLC: 1021874142.

[153] Sakuma, T., Mukai, T., and Kuriyama, S. Psychological model for animating crowded pedestrians. *Computer Animation and Virtual Worlds*, 16(3-4):343–351, July 2005.

[154] Schadschneider, A., Klingsch, W., Klpfel, H., Kretz, T., Rogsch, C., and Seyfried, A. Evacuation Dynamics: Empirical Results, Modeling and Applications. In Ph.D, R. A. M., editor, *Encyclopedia of Complexity and Systems Science*, pages 3142–3176. Springer New York, 2009. DOI: 10.1007/978-0-387-30440-3_187.

[155] Schneega, S., Pfleging, B., Kern, D., and Schmidt, A. Support for Modeling Interaction with Automotive User Interfaces. In *Proceedings of the 3rd International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, AutomotiveUI '11, pages 71–78, New York, NY, USA, 2011. ACM. event-place: Salzburg, Austria.

[156] Schuerman, M., Singh, S., Kapadia, M., and Faloutsos, P. Situation agents: agent-based externalized steering logic. *Computer Animation and Virtual Worlds*, 2010.

[157] Seer, S., Rudloff, C., Matyus, T., and Brndle, N. Validating Social Force based Models with Comprehensive Real World Motion Data. *Transportation Research Procedia*, 2:724–732, January 2014.

[158] Seffah, A., Donyaee, M., Kline, R. B., and Padda, H. K. Usability measurement and metrics: A consolidated model. *Software Quality Journal*, 14(2):159–178, June 2006.

[159] Shao, W. and Terzopoulos, D. Autonomous pedestrians. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 19–28. ACM, 2005.

[160] Shao, W. and Terzopoulos, D. Environmental modeling for autonomous virtual pedestrians. Technical report, SAE Technical Paper, 2005.

[161] Shen, Y., Henry, J., Wang, H., Ho, E. S., Komura, T., and Shum, H. P. Data-Driven Crowd Motion Control with Multi-touch Gestures. *Computer Graphics Forum*, 37(6): 382–394, Sep 2018.

[162] Simulex. Simulex. http://www.iesve.com/software/ve-for-engineers/module/Simulex/480, n.d.

[163] Simwalk. SIMWALK: Smarter People Flow Services Start Now. http://www.simwalk.com, n.d.

[164] Singh, S., Kapadia, M., Faloutsos, P., and Reinman, G. SteerBench: a benchmark suite for evaluating steering behaviors. *Computer Animation and Virtual Worlds*, 20(56):533–548, February 2009.

[165] Snook, G. Simplified 3d movement and pathfinding using navigation meshes. *Game Programming Gems*, 1(1):288–304, 2000.

[166] Stvel, S. A., Magnenat-Thalmann, N., Thalmann, D., v. d. Stappen, A. F., and Egges, A. Torso crowds. *IEEE Transactions on Visualization and Computer Graphics*, 23(7): 1823–1837, July 2017.

[167] Sud, A., Gayle, R., Andersen, E., Guy, S., Lin, M., and Manocha, D. Real-time navigation of independent agents using adaptive roadmaps. In *Proceedings of the 2007 ACM symposium on Virtual reality software and technology*, pages 99–106. ACM, 2007.

[168] Sud, A., Andersen, E., Curtis, S., Lin, M. C., and Manocha, D. Real-time path planning in dynamic virtual environments using multiagent navigation graphs. *IEEE transactions on visualization and computer graphics*, 14(3):526–538, 2008.

[169] Sung, M., Gleicher, M., and Chenney, S. Scalable behaviors for crowd simulation. In *Computer Graphics Forum*, volume 23, pages 519–528, 2004.

[170] Takahashi, S., Yoshida, K., Kwon, T., Lee, K. H., Lee, J., and Shin, S. Y. Spectral-Based Group Formation Control. *Computer Graphics Forum*, 2009.

[171] Tecchia, F., Loscos, C., Dalton, R., and Chrysanthou, Y. Agent behaviour simulator (abs): A platform for urban behaviour development. In *GTEC*, 2001.

[172] Treuille, A., Cooper, S., and Popović, Z. Continuum Crowds. In *Proc. SIGGRAPH 2006*, pages 1160–1168, 2006.

[173] Tsiros, A. and Leplâtre, G. *Evaluation of a Sketching Interface to Control a Concatenative Synthesiser*. Ann Arbor, MI: Michigan Publishing, University of Michigan Library, 2016.

[174] Ulicny, B. and Thalmann, D. Crowd simulation for interactive virtual environments and vr training systems. In Magnenat-Thalmann, N. and Thalmann, D., editors, *Computer Animation and Simulation 2001*, pages 163–170. Springer, Vienna, 2001.

[175] Ulicny, B. and Thalmann, D. Towards Interactive Real-Time Crowd Behavior Simulation. *Computer Graphics Forum*, 21(4):767–775, December 2002.

[176] Ulicny, B., Ciechomski, P. d. H., and Thalmann, D. Crowdbrush: Interactive Authoring of Real-time Crowd Scenes. In *Proc. SCA'04*, 2004.

[177] University of Greenwich. Exodus Introduction. http://fseg.gre.ac.uk/exodus/index.html, 1997.

[178] Van den Berg, J., Lin, M., and Manocha, D. Reciprocal Velocity Obstacles for real-time multi-agent navigation. In *2008 IEEE International Conference on Robotics and Automation*, pages 1928–1935, May 2008. ISSN: 1050-4729.

[179] Van den Berg, J., Guy, S. J., Lin, M., and Manocha, D. Reciprocal n-Body Collision Avoidance. In Pradalier, C., Siegwart, R., and Hirzinger, G., editors, *Robotics Research*, Springer Tracts in Advanced Robotics, pages 3–19. Springer Berlin Heidelberg, 2011.

[180] Van Toll, W., Cook, A. F., and Geraerts, R. Navigation meshes for realistic multi-layered environments. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3526–3532, Sept 2011.

[181] van Toll, W. G., Cook, A. F., and Geraerts, R. A navigation mesh for dynamic environments: A navigation mesh for dynamic environments. *Computer Animation and Virtual Worlds*, 23(6):535–546, November 2012.

[182] Wang, H., Ondrej, J., and O'Sullivan, C. Trending Paths: A New Semantic-Level Metric for Comparing Simulated and Real Crowd Data. *IEEE Transactions on Visualization and Computer Graphics*, 23(5):1454–1464, May 2017.

[183] Wetzlinger, W., Auinger, A., and Dörflinger, M. Comparing effectiveness, efficiency, ease of use, usability and user experience when using tablets and laptops. In *International Conference of Design, User Experience, and Usability*, pages 402–412. Springer, 2014.

[184] Wiggins, J. S. *The five-factor model of personality: Theoretical perspectives*. Guilford Press, 1996.

[185] Wolinski, D., J. Guy, S., Olivier, A.-H., Lin, M., Manocha, D., and Pettr, J. Parameter estimation and comparative evaluation of crowd simulations. *Computer Graphics Forum*, 33(2):303–312, May 2014.

[186] Xu, J., Jin, X., Yu, Y., Shen, T., and Zhou, M. Shape-constrained flock animation. *Computer Animation and Virtual Worlds*, 19(3-4):319–330, 2008.

[187] Xu, M., Wu, Y., and Ye, Y. Smooth and efficient crowd transformation. In *Proceedings of the 20th ACM international conference on Multimedia*, pages 1189–1192. ACM, 2012.

[188] Xu, M., Wu, Y., Ye, Y., Farkas, I., Jiang, H., and Deng, Z. Collective Crowd Formation Transform with Mutual Information-Based Runtime Feedback: Collective Crowd Formation Transform with Mutual Information-Based Runtime Feedback. *Computer Graphics Forum*, 34(1):60–73, February 2015.

[189] Xu, X., Liu, W., Jin, X., and Sun, Z. Sketch-based user interface for creative tasks. In *Proceedings of the 5th Asia Pacific conference on computer human interaction, Beijing*, pages 560–570, 2002.

[190] Yang, S., Li, T., Gong, X., Peng, B., and Hu, J. A review on crowd simulation and modeling. *Graphical Models*, 111:101081, 2020.

[191] Yeh, H., Curtis, S., Patil, S., van den Berg, J., Manocha, D., and Lin, M. Composite agents. In *Proc. SCA'08*, pages 39–47, 2008.

[192] Yersin, B., Maïm, J., Ciechomski, P., Schertenleib, S., and Thalmann, D. Steering a virtual crowd based on a semantically augmented navigation graph. In *Proc. V-CROWDS'05*, pages 169–178, 2005.

[193] Yersin, B., Maïm, J., Pettré, J., and Thalmann, D. Crowd patches: Populating large-scale virtual environments for real-time applications. In *Proc. I3D'09*, pages 207–214, 2009.

[194] Yu, Q. and Terzopoulos, D. A decision network framework for the behavioral animation of virtual humans. In *Proc. SCA'07*, pages 119–128, 2007.

[195] Zhang, P., Liu, H., and Ding, Y.-h. Crowd simulation based on constrained and controlled group formation. *The Visual Computer*, 31(1):5–18, January 2015.

[196] Zheng, L., Zhao, J., Cheng, Y., Chen, H., Liu, X., and Wang, W. Geometry-constrained crowd formation animation. *Computers & Graphics*, 38:268–276, February 2014.

Appendices

Appendix A

MassMotion SDK

A MassMotion license was obtained from *Oasys* to explore the possibility of interacting with the simulation in real-time. The license included the MassMotion SDK to have more options than the system interface. This section summarises the report to *Oasys* as a result of the work done with the SDK.

The goal of this work is to modify the environment in real-time, which is not possible using the interface. However, MassMotion's SDK allows to programmatically run and interact with simulations. Altering the simulation as it runs and visualising the result afterwards simulates the interaction in real-time.

MassMotion generates an obstacle map and an approach map of the environment, which are used to calculate the path of the agents. These maps are created while compiling the project before starting the simulation. Therefore it is not possible to update these maps once the simulation has started. The only possible way to simulate the effect of an obstacle created in real-time is to add static agents with a large radius blocking the path followed by the other agents. The profile of the static agents was modified to set their radius to 10. Additionally, their avatar was modified to look like a barrier, as shown in Figure A.1.

Using the SDK, the static agents are added to the simulation at a specific frame in a specific position while the simulation is running. This approach is not intuitive since the exact frame and position must be known by the user. To prevent MassMotion from updating the static agents' position and avoid overlapping with the walking agents, it is necessary to disallow their adjustment via the SDK. Figure A.2 shows three frames of the simulation running. In the first frame, pedestrians are walking towards the portal on the right. The barriers appear in the second frame; the blue dotted lines indicate the radius of the obstacles. The last frame shows agents reacting and avoiding static agents. The original idea was to create one large barrier; however, increasing the radius of the agent beyond a particular value did not have any effect on the simulation. For this reason, three small obstacles were added. Overlapping the static agents did not help since walking agents were able to find a way through the obstacles.

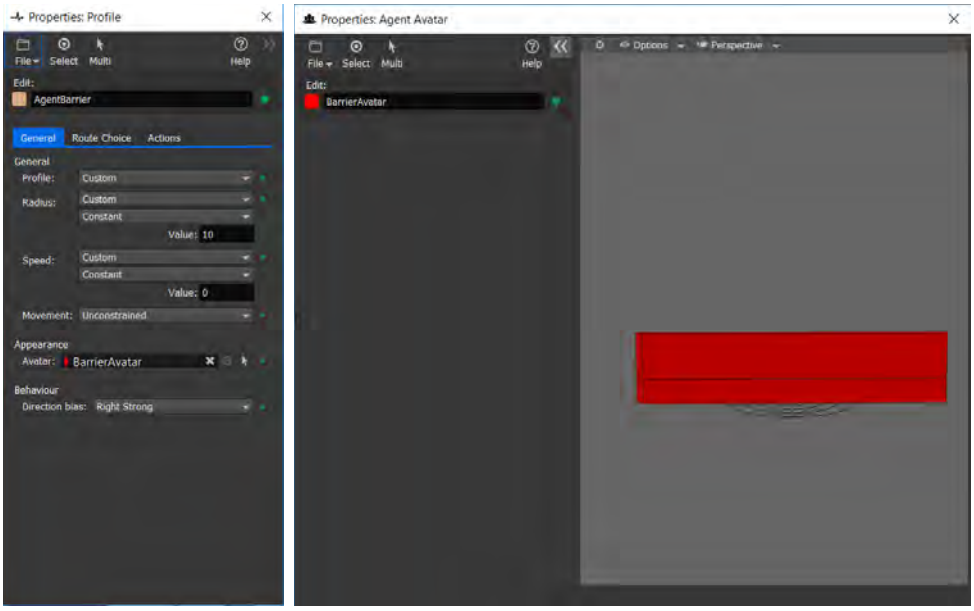
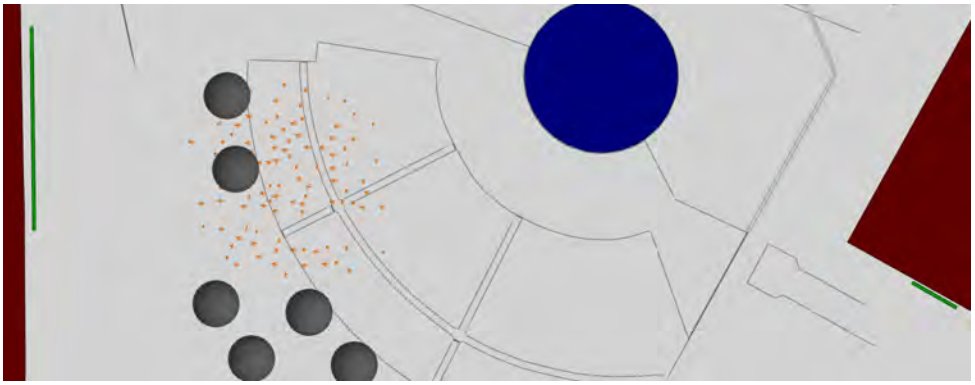
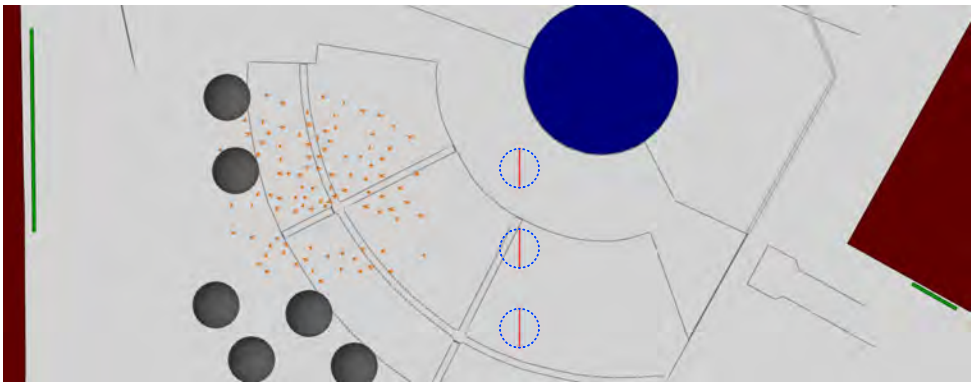


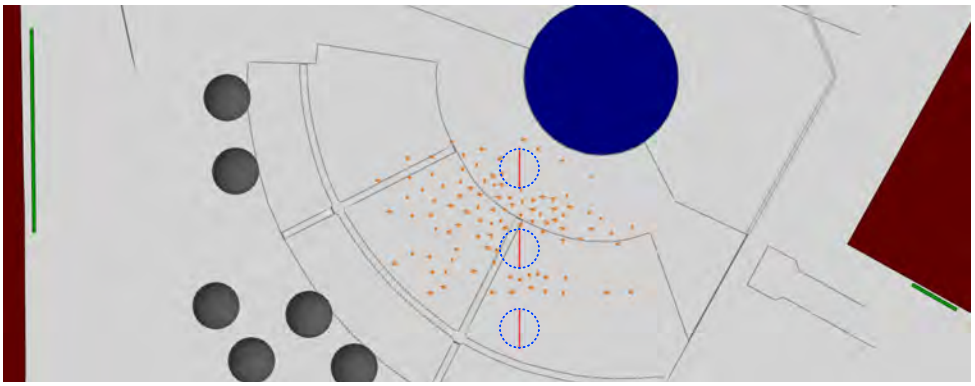
Figure A.1: Agent profile and avatar used for static agents.



(a)



(b)



(c)

Figure A.2: Simulation running in MassMotion. (a) Pedestrians walking from the left portal to the right portal. (b) Three obstacles appear while the simulation is running. (c) Pedestrians avoiding the barriers and walking past them. The blue dotted lines indicate the radius of the obstacles.

Appendix B

User Study

This appendix includes all the relevant documents and the results of the user study. Documents include:

- Information sheet
- Consent form
- Instruction sheet
- System Usability Survey
- Expert KLM results
- Participant KLM results
- Participant metrics
- System Usability Survey results

Participant Information Sheet

Sketching for Real-time Control of Crowd Simulations

You are being invited to take part in a research project. Before you decide whether or not to participate, it is important for you to understand why the research is being done and what it will involve. Please take time to read the following information carefully and discuss it with others if you wish. Ask us if there is anything that is not clear or if you would like more information. Take time to decide whether or not you wish to take part. Thank you for reading.

1. What is the project's purpose?

The aim of the research is to investigate the use of sketching to graphically control crowd simulations. This method involves directly drawing (with a mouse) in the 3D environment to alter pedestrian behaviour and to modify the environment itself. This research proposes several ideas to extend the use of sketching. These ideas include sketching barriers to block paths, sketching way-point areas and creating storyboards to specify the journeys of crowd subgroups. The purpose of the user study is to compare the effectiveness, efficiency and usability of the proposed sketching interface against the commercial software called MassMotion.

2. Why have I been chosen?

You have been chosen to participate as the research project requires people with no prior knowledge of the software involved.

3. Do I have to take part?

Participating in this research is entirely voluntary and it is up to you to decide whether or not to take part. If you do decide to take part, you will be given this information sheet to keep and be asked to sign a consent form. You can still withdraw at any time without any negative consequences. You do not have to give a reason. If you wish to withdraw from the research, please contact Luis Montana or Dr Steve Maddock on their contact details below (section 15). Your data will then be deleted immediately.

4. What will happen to me if I take part? What do I have to do?

The user study consists of comparing the process of setting up a crowd simulation using the proposed sketching interface and the commercial software MassMotion. You will be asked to create three scenarios in both systems.

The first scenario initially consists of only one entrance and one exit. After running and observing the simulation, a barrier is added to block the path followed by the crowd.

The second scenario has three entrances and three exits. The user is asked to equally divide the pedestrians spawning from each entrance into three groups. Every group should move towards a different exit.

The last scenario includes two entrances, one exit and two areas used as waypoints, where pedestrians briefly wait before continuing to their destination. The goal of the scenario is to specify the journey and actions of the crowd. This scenario consists of two stages. First, the crowd spawning from each entrance visits one area and then walks towards the exit. Second, after running the simulation, the journeys are modified to force the crowd to visit different areas.

Before the beginning of the experiment, you will be provided with the information sheet, the consent form and an instructions sheet. Based on earlier experiments, the expected completion time is up to 3 hours. However, it could be finished in less time. The experiment will be divided into two sessions (one per software) with a break of 15 minutes. There will also be 5-minute breaks between each scenario. At the start of each session, a general overview of the software in question will be given to you. Then, you will be asked to complete three scenarios following a list of instructions. The session will be video recorded to count the number of operations (i.e. mouse clicks, keystrokes) that were needed to complete the scenarios. The video

will focus on hand movement and the screen. At the end of each session, you will be asked to answer a small questionnaire to assess the usability of the software.

You are free to discuss anything that takes place in the session and may withdraw at any time.

5. Will I be recorded, and how will the recorded media be used?

You will be recorded while completing the scenarios in both systems. The video recordings of your activities made during this research will be used for analysis purposes only. The video will be used to count the number of operations needed to complete the experiment. The video camera will be focussed on the keyboard, mouse and screen rather than the user.

6. What are the possible disadvantages and risks of taking part?

There are no known risks of taking part in this experiment. However, if at any point during the recording, you decide that you do not want to carry on, I will stop the recording and you are free to withdraw from the study without giving a reason.

7. What are the possible benefits of taking part?

Your participation will contribute to the project which aims to improve and make more intuitive the control of crowd simulations. In exchange for completing the experiment, you will receive £25.

8. Will my taking part in this project be kept confidential?

All the information collected about you in this study will be kept strictly confidential in a password-protected Google drive folder. It will only be accessible to the student researcher and the supervisor for the duration of the project. If you agree to have your data stored, the recorded video and questionnaires will be saved in The University of Sheffield Research Data Catalogue and Repository (ORDA) and will only be accessible to the student researcher, the supervisor and other authorised researchers. You will not be able to be identified in any reports or publications. If you do not agree to the data being saved, it will be destroyed at the end of the project.

9. What is the legal basis for processing my personal data?

According to data protection legislation, we are required to inform you that the legal basis we are applying to process your personal data is that 'processing is necessary for the performance of a task carried out in the public interest' (Article 6(1)(e)). Further information can be found in the University's Privacy Notice https://www.sheffield.ac.uk/govern/data-protection/privacy/general.

10. What will happen to the data collected, and the results of the research project?

The results will be analysed and discussed as part of the researcher's PhD thesis and this analysis will likely form the basis of a research publication. Due to the nature of this research, other researchers may find the data collected to be useful in answering future research questions. We will ask for your explicit consent for your data to be stored and shared in this way. Otherwise, the data will be deleted at the end of the project.

11. Who is organising and funding the research?

The researcher's PhD is funded by CONACYT (Consejo Nacional de Ciencia y Tecnologia).

12. Who is the Data Controller?

The University of Sheffield will act as the Data Controller for this study. This means that the University is responsible for looking after your information and using it properly.

To collect and use your personal information as part of this research project, we must have a basis in law to do so. The basis that we are using is that the research is 'a task in the public interest'.
Further information, including details about how and why the University processes your personal information, how we keep your information secure, and your legal rights (including how to complain if you feel that your personal information has not been handled correctly), can be found in the University's Privacy Notice https://www.sheffield.ac.uk/govern/data-protection/privacy/general.

13. Who has ethically reviewed the project?

This project has been ethically approved via the University of Sheffield's Ethics Review Procedure, as administered by the Computer Science department. The University's Research Ethics Committee monitors the application and delivery of the University's Ethics Review Procedure across the University.

14. What if something goes wrong and I wish to complain about the research?

If you have any complaints, either from the researcher or something occurring during or following your participation in the project (e.g. a reportable serious adverse event), please contact Luis Montana or Dr Steve Maddock using the contact details below (section 15).

Should you feel your complaint has not been handled to your satisfaction, you can contact the Head of Department, Professor Guy Brown (g.j.brown@sheffield.ac.uk) who will then escalate the complaint through the appropriate channels. If the complaint relates to how your personal data has been handled, information about how to raise a complaint can be found in the University's Privacy Notice: https://www.sheffield.ac.uk/govern/data-protection/privacy/general.

15. Contact for further information

If you have any questions, please do not hesitate to contact us. Our contact details are:

Researcher Name: Luis Rene Montana Gonzalez
Researcher Email: lrmontanagonzalez1@sheffield.ac.uk

Supervisor Name: Dr Steve Maddock
Supervisor Email: s.maddock@sheffield.ac.uk

Address:
Department of Computer Science
University of Sheffield
Regent Court
211 Portobello
S1 4DP

Thank you very much for your time to read this information sheet.

You will be given a copy of this information sheet and a signed consent form to keep.

The University Of Sheffield.

Participant Consent Form

Sketching for Real-time Control of Crowd Simulations Consent Form

Please tick the appropriate boxes	Yes	No
Taking Part in the Project		
I have read and understood the project information sheet dated [ ] or the project has been fully explained to me. (If you will answer No to this question please do not proceed with this consent form until you are fully aware of what your participation in the project will mean.)	☐	☐
I have been given the opportunity to ask questions about the project.	☐	☐
I agree to take part in the project. I understand that taking part in the project will include: - Being recorded while performing a list of tasks using a piece of software - Completing questionnaires	☐ ☐	☐ ☐
I understand that my taking part is voluntary and that I can withdraw from the study at any time; I do not have to give any reasons for why I no longer want to take part and there will be no adverse consequences if I choose to withdraw.	☐	☐
I understand that I will be paid on completion of the experiment.	☐	☐
How my information will be used during and after the project		
I understand my personal details such as name will not be revealed to people outside the project.	☐	☐
I understand and agree that my words may be quoted in publications, reports, web pages, and other research outputs. Anonymous participant IDs will be used to refer to people.	☐	☐
I understand and agree that, if I permit to store my data, other authorised researchers will have access to the recorded video and questionnaires only if they agree to preserve the confidentiality of the information as requested in this form.	☐	☐
I understand and agree that, if I permit to store my data, other authorised researchers may use my data in publications, reports, web pages, and other research outputs, only if they agree to preserve the confidentiality of the information as requested in this form.	☐	☐
I give permission for the recorded video and questionnaires to be deposited in The University of Sheffield Research Data Catalogue and Repository (ORDA).	☐	☐
So that the information you provide can be used legally by the researchers		
I agree to assign the copyright I hold in any materials generated as part of this project to The University of Sheffield.	☐	☐

Name of participant [printed]: Signature Date

Name of Researcher [printed]: Signature Date

Project contact details for further information:

Researcher: Luis Rene Montana Gonzalez Researcher Email: lrmontanagonzalez1@sheffield.ac.uk

Supervisor: Dr Steve Maddock Supervisor Email: s.maddock@sheffield.ac.uk

Instruction Sheet

Scenario 1

This scenario is divided into two parts. The first part only includes one entrance and one exit. The goal is to spawn 1,000 pedestrians which will find the shortest path to the exit. In the second part of the scenario, a barrier is added to block the path followed by the crowd. This obstacle will force pedestrians to find a new route. Figure 1 shows the first part of the scenario in MassMotion.

Figure 1 Part A of the scenario in MassMotion.

This scenario is divided into the following tasks:

Part A

1. Create and rename the entrance
2. Create and rename the exit
3. Spawn 1,000 pedestrians at a 10 pedestrians/s rate
4. Run the simulation
5. Stop the simulation when all agents exit the simulation

Part B (Figure 2)

6. Create the barrier
7. Run the simulation
8. Stop the simulation when all agents exit the simulation

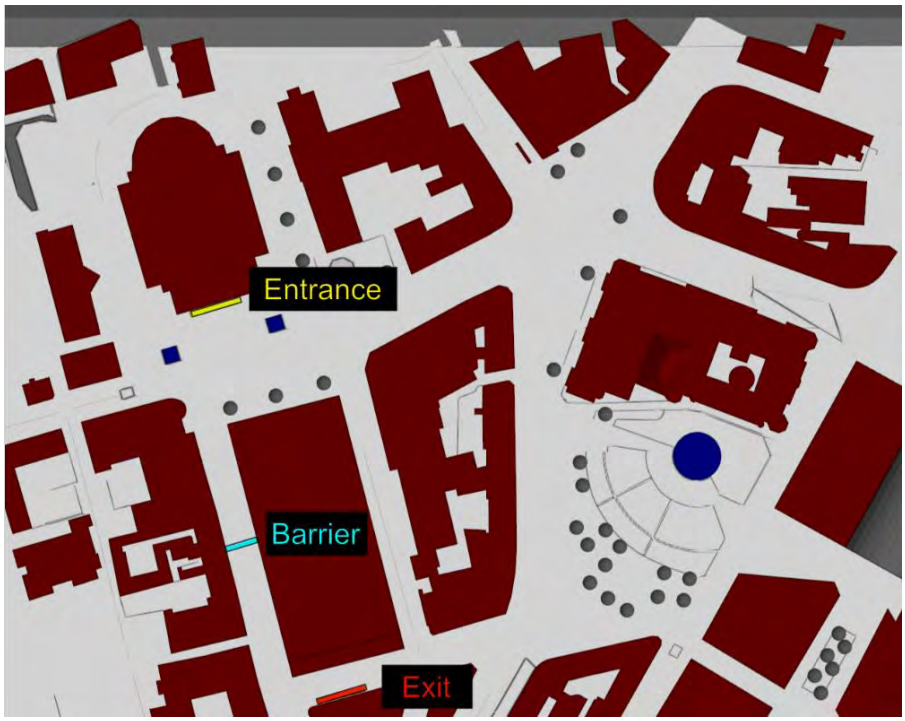


Figure 2 Part B of the scenario in MassMotion.

Figure 3 shows both parts of the scenario using the Sketching system.

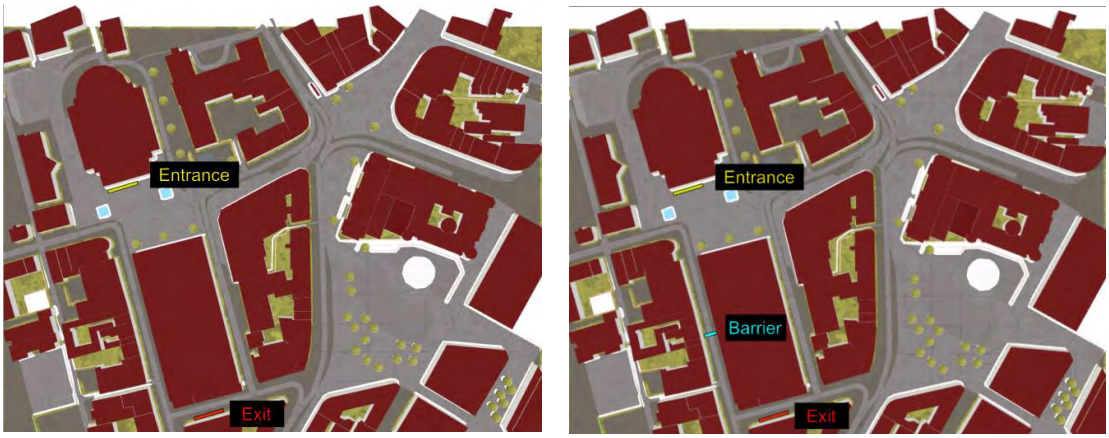


Figure 3 Part A and B of the scenario using the Sketching system.

Scenario 2

This scenario consists of three entrances and three exits. The pedestrians spawning from each entrance are divided into three equal groups. Each group should move towards a different exit. Figure 4 shows the location of the entrances and exits in MassMotion.

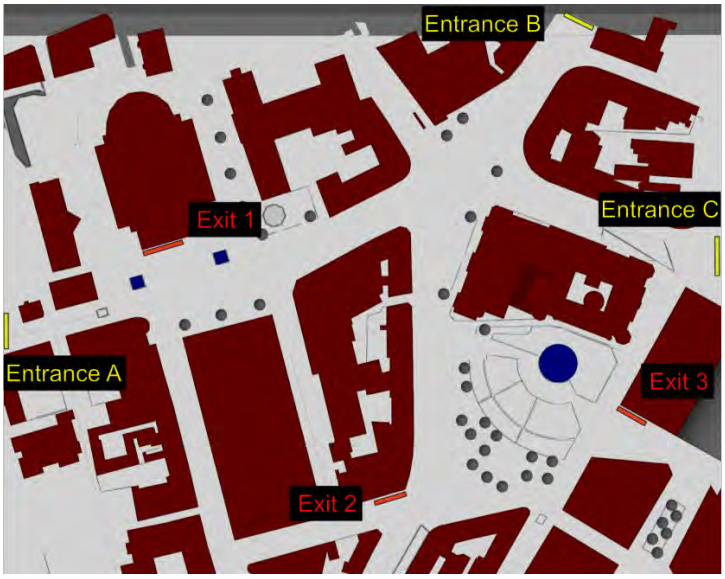


Figure 4 Scenario 2 in MassMotion

This scenario is divided into the following tasks:

1. Create and rename the entrances
2. Create and rename the exits
3. Spawn 900 pedestrians per entrance at a 3 pedestrians/s rate
4. For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit. For example, the groups from Entrance A would be: 300 agents to Exit 1, 300 agents to Exit 2 and 300 agents to Exit 3.
5. Run the simulation
6. Stop the simulation when all agents exit the simulation

Figure 5 shows the layout of the scenario in the Sketching system.

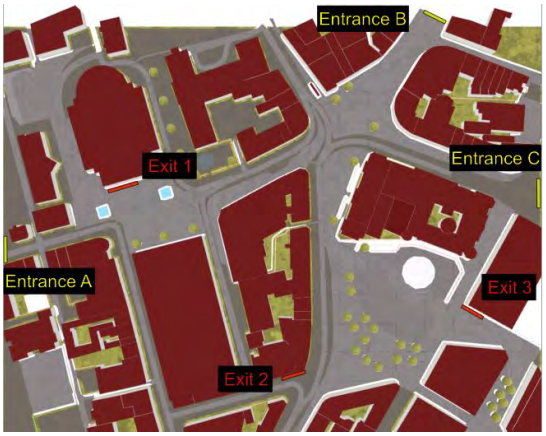


Figure 5 Scenario 2 in the Sketching system

Scenario 3

The last scenario includes two entrances, one exit and two areas, used as waypoints, where pedestrians wait before continuing to their destination. The goal of the scenario is to specify the journey and actions of the crowd. This scenario consists of two stages. First, the crowd spawning from Entrance A visits Area 1 and then walks towards the Exit, while pedestrians from Entrance B go to Area 2 before continuing to the Exit.

After running the simulation, the journeys are modified to complete the second stage. Now the crowd from Entrance A visits both areas. The pedestrians from Entrance B go directly to the Exit. Figure 6 shows the first part of the scenario in MassMotion.

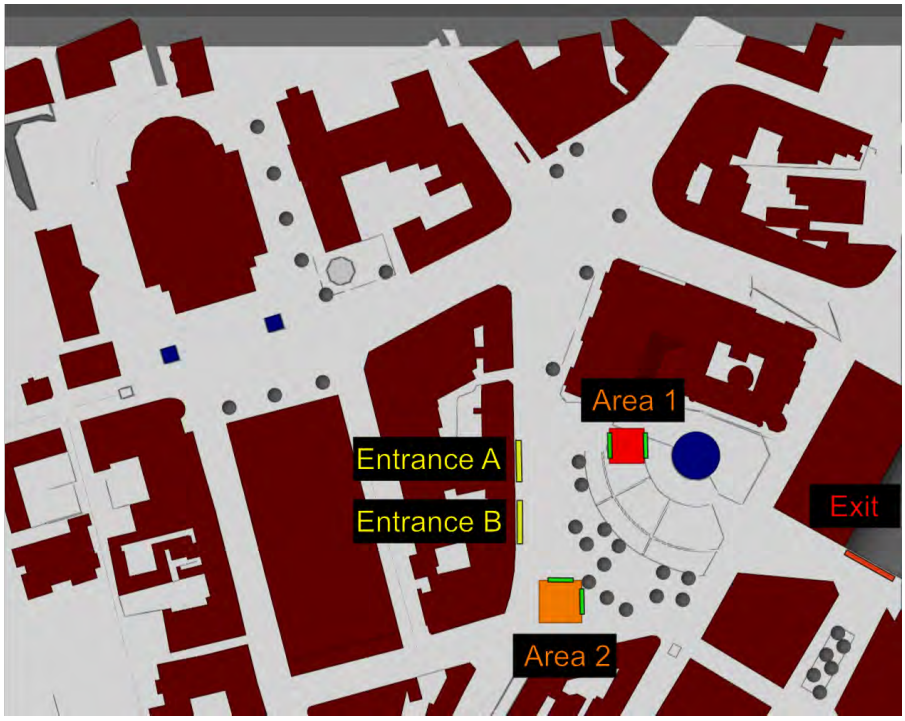


Figure 6 First part of Scenario 3 in MassMotion

This scenario is divided into the following tasks:

Part A

1. Create and rename entrances
2. Create and rename exit
3. Create and rename areas
4. Spawn 900 pedestrians per entrance at a 3 pedestrians/s rate
5. Make pedestrians from Entrance A visit Area 1 and wait 10 seconds inside then move to the exit
6. Make pedestrians from Entrance B visit Area 2 and wait 10 seconds inside then move to the exit
7. Run the simulation
8. Stop the simulation when all agents exit the simulation

Part B (Figure 7)

9. Change the journey of Entrance A to visit and wait in both areas before going to the exit
10. Change the journey of Entrance B to move directly to the exit
11. Create a barrier
12. Run the simulation
13. Stop the simulation when all agents exit the simulation

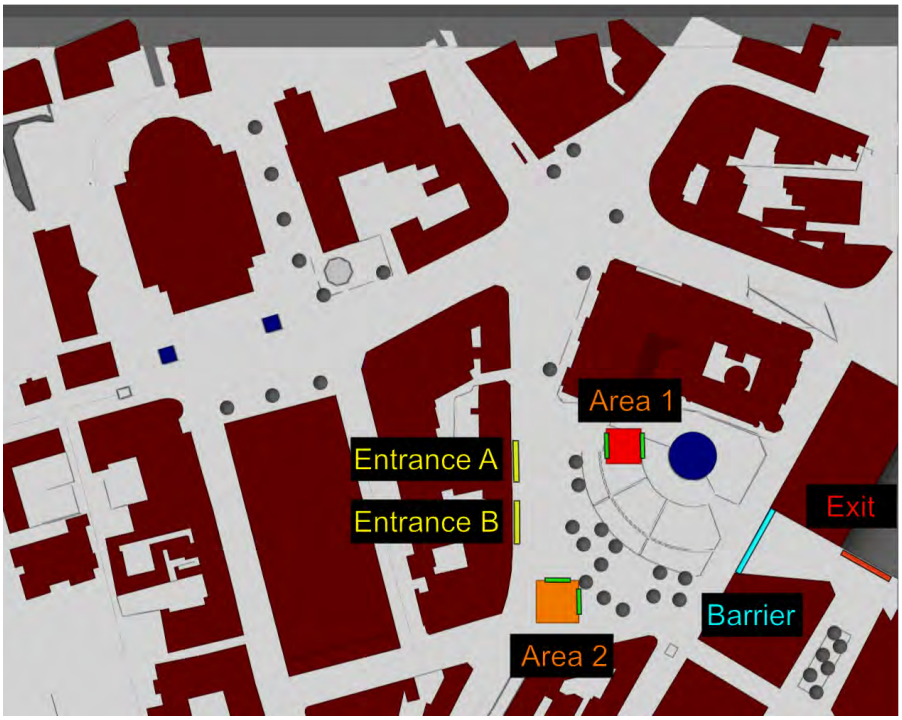


Figure 7 Part B of the third scenario in MassMotion

Figure 8 shows both parts of the scenario using the Sketching system.

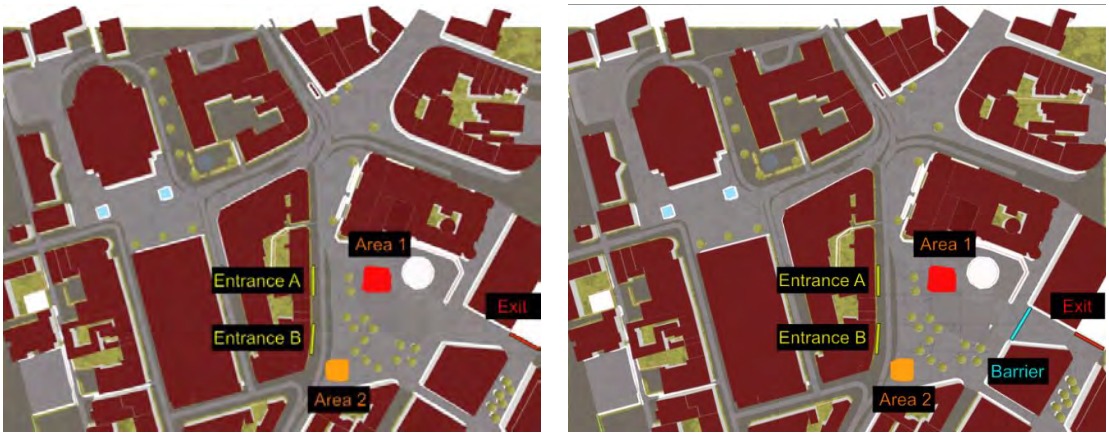


Figure 8 Part A and B of Scenario 3 in the Sketching system.

Participant ID: **System Usability Survey** **Date:**

System: MassMotion | Sketching

Instructions
For each of the following statements, mark one box that best describes your reactions to the system today.

	Strongly disagree				Strongly agree
1. I think that I would like to use this system frequently if I was creating crowd simulations	1	2	3	4	5
2. I found the system unnecessarily complex	1	2	3	4	5
3. I thought the system was easy to use	1	2	3	4	5
4. I think that I would need the support of a technical person to be able to use this system	1	2	3	4	5
5. I found the various functions in this system were well integrated	1	2	3	4	5
6. I thought there was too much inconsistency in this system	1	2	3	4	5
7. I would imagine that most people would learn to use this system very quickly	1	2	3	4	5
8. I found the system very cumbersome to use	1	2	3	4	5
9. I felt very confident using the system	1	2	3	4	5
10. I needed to learn a lot of things before I could get going with this system	1	2	3	4	5

Expert KLM Results

This section shows the KLM analysis of the expert user completing the three scenarios. Figure B.1 shows the description of the KLM operators used. Figures B.2 and B.3 summarises the KLM results and time of the expert user for all the scenarios in both systems. Then, detailed data for each scenario is presented individually.

Operators	Description
K	Keystroke or button press
VS	Visual searching for item in interface
P	Pointing to a target with mouse
H	Hand movement between mouse/keyboard
D	Drag or sketch
M	Mentally preparing for executing actions
R	Response of the system

Figure B.1: KLM operator description

	Operators	
Scenario	Massmotion	Sketching
1	89K 37VS 63P 5H 25D 27M 2R	64K 16VS 26P 5H 4D 20M
2	193K 76VS 118P 12H 54D 51M 1R	173K 38VS 62P 21H 6D 41M
3	336K 156VS 246P 16H 71D 80M 2R	175K 43VS 78P 13H 12D 54M

Figure B.2: Summary of KLM analysis of the expert user for all the scenarios using MassMotion and the sketching system.

	Time (s)							
	Setup		**Compile**		**Run**		**Total**	
Scenario	MM	Sketching	MM	Sketching	MM	Sketching	MM	Sketching
1	195	63	91	0	126	108	412	171
2	330	147	88	0	63	73	481	220
3	566	141	165	0	86	98	817	239

Figure B.3: Completion time divided into setup, compilation and run time for every scenario in MassMotion and the sketching approach.

Expert Scenario 1 MassMotion

Elements	
1	Entrance
1	Exit
1	Barrier

Tasks
Create entrance
Create exit
Spawn 1000 pedestrians at a 10 pedestrians/sec rate
Run/Stop simulation
Create a barrier blocking the way followed by the crowd
Run/Stop simulation

Figure B.4: Simulation elements included in Scenario 1 and the tasks needed to complete the scenario.

Create Entrance/Exit	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	1	0	0	0	2	1	0	1K 2D 1M
Go to Scene tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Position portal	3	3	8	0	5	1	0	3K 3VS 8P 5D 1M
Rename portal	11	2	3	2	1	1	0	11K 2VS 3P 2H 1D 1M
	17	7	13	2	8	5	0	17K 7VS 13P 2H 8D 5M

Figure B.5: KLM operators used by the actions required to create an entrance or exit.

Spawn pedestrians	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Go to Activitites tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set No. pedestrians	7	1	1	2	1	1	0	7K 1VS 1P 2H 1D 1M
Set rate/time	6	1	1	2	1	1	0	6K 1VS 1P 2H 1D 1M
Select Origin	4	2	3	0	0	1	0	4K 2VS 3P 1M
Select Destination	5	2	4	0	0	1	0	5K 2VS 4P 1M
	24	8	11	4	2	6	0	24K 8VS 11P 4H 2D 6M

Figure B.6: KLM operators used by the actions required to specify the numbers of pedestrians to spawn and the spawn rate.

Run Simulation	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Go to Simulation tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Run simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Run	5	2	5	0	0	1	1	5K 2VS 5P 1M 1R
	7	4	7	0	0	3	1	7K 4VS 7P 3M 1R

Figure B.7: KLM operators used by the actions required to run and stop the simulation.

Create Barrier	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	1	0	0	0	2	1	0	1K 2D 1M
Go to Scene tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Position Barrier	3	3	7	0	4	1	0	3K 3VS 7P 4D 1M
Rename barrier	11	2	3	2	1	1	0	11K 2VS 3P 2H 1D 1M
	17	7	12	2	7	5	0	17K 7VS 12P 2H 7D 5M

Figure B.8: KLM operators used by the actions required to create a barrier.

Task	K	VS	P	H	D	M	R
Create entrance	17	7	13	1	8	5	0
Create exit	17	7	13	1	8	5	0
Spawn 1000 pedestrians at a 10 pedestrians/sec rate	24	8	11	2	2	6	0
Run simulation	7	4	7	0	0	3	1
Create a barrier blocking the way followed by the crowd	17	7	12	1	7	5	0
Run simulation	7	4	7	0	0	3	1
	89	37	63	5	25	27	2

Figure B.9: KLM summary of the Scenario 1 tasks.

Task	KLM
Create entrance	17K 7VS 13P 1H 8D 5M
Create exit	17K 7VS 13P 1H 8D 5M
Spawn 1000 pedestrians at a 10 pedestrians/sec rate	24K 8VS 11P 2H 2D 6M
Run simulation	7K 4VS 7P 3M 1R
Create a barrier blocking the way followed by the crowd	17K 7VS 12P 1H 7D 5M
Run simulation	7K 4VS 7P 3M 1R
	89K 37VS 63P 5H 25D 27M 2R

Figure B.10: KLM total of the Scenario 1 tasks.

Time (s)			
Setup	Compilation	Run	Total
195	91	126	412

Figure B.11: Scenario 1 Completion time divided into setup, compilation and run time.

Expert Scenario 1 Sketching

Create Entrance/Exit	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	4	0	0	0	1	1	0	4K 1D 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Rename Entrance/Exit	10	2	4	1	0	1	0	10K 2VS 4P 1H 1M
	16	3	6	1	1	4	0	16K 3VS 6P 1H 1D 4M

Figure B.12: KLM operators used by the actions required to create an entrance or exit.

Spawn pedestrians	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Click Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Entrance	1	0	1	0	0	1	0	1K 1P 1M
Set No. pedestrians	6	1	1	1	0	1	0	6K 1VS 1P 1H 1M
Set rate/time	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
	13	3	4	2	0	4	0	13K 3VS 4P 2H 4M

Figure B.13: KLM operators used by the actions required to specify the numbers of pedestrians to spawn and the spawn rate.

Run simulation	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	2	2	2	0	0	2	0	2K 2VS 2P 2M

Figure B.14: KLM operators used by the actions required to run and stop the simulation.

Create Barrier	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	4	0	0	0	1	1	0	4K 1D 1M
Select Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Sketch Barrier	0	0	1	0	1	1	0	1P 1D 1M
Rename Barrier	10	2	4	1	0	1	0	10K 2VS 4P 1H 1M
	15	3	6	1	2	4	0	15K 3VS 6P 1H 2D 4M

Figure B.15: KLM operators used by the actions required to create a barrier.

Task	K	VS	P	H	D	M	R
Create entrance	16	3	6	1	1	4	0
Create exit	16	3	6	1	1	4	0
Spawn 1000 pedestrians at a 10 pedestrians/sec rate	13	3	4	2	0	4	0
Run simulation	2	2	2	0	0	2	0
Create a barrier blocking the way followed by the crowd	15	3	6	1	2	4	0
Run simulation	2	2	2	0	0	2	0
	64	16	26	5	4	20	0

Figure B.16: KLM summary of the Scenario 1 tasks.

Task	KLM
Create entrance	16K 3VS 6P 1H 1D 4M
Create exit	16K 3VS 6P 1H 1D 4M
Spawn 1000 pedestrians at a 10 pedestrians/sec rate	13K 3VS 4P 2H 4M
Run simulation	2K 2VS 2P 2M
Create a barrier blocking the way followed by the crowd	15K 3VS 6P 1H 2D 4M
Run simulation	2K 2VS 2P 2M
	64K 16VS 26P 5H 4D 20M

Figure B.17: KLM total of the Scenario 1 tasks.

Time (s)		
Setup	Run	Total
63	108	171

Figure B.18: Scenario 1 Completion time divided into setup and run time.

Expert Scenario 2 MassMotion

Elements	
3	Entrances
3	Exits
3	Storyboards

Tasks
Create entrances
Create exits
Spawn 900 pedestrians at a 3 peds/s rate
Divide the crowd in 3 groups
Repeat for all entrances
Run/Stop simulation

Figure B.19: Simulation elements included in Scenario 2 and the tasks needed to complete the scenario.

Create Entrance/Exit	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	1	0	0	0	2	1	0	1K 2D 1M
Go to Scene tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Position portal	3	3	8	0	5	1	0	3K 3VS 8P 5D 1M
Rename portal	11	2	3	1	1	1	0	11K 2VS 3P 1H 1D 1M
	17	7	13	1	8	5	0	17K 7VS 13P 1H 8D 5M

Figure B.20: KLM operators used by the actions required to create an entrance or exit.

Spawn/Divide crowd	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Go to Activitites tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set No. pedestrians	7	1	1	1	1	1	0	7K 1VS 1P 1H 1D 1M
Set rate/time	6	1	1	1	1	1	0	6K 1VS 1P 1H 1D 1M
Select Origin	4	2	3	0	0	1	0	4K 2VS 3P 1M
Select Destination	9	4	4	0	0	1	0	9K 4VS 4P 1M
	28	10	11	2	2	6	0	28K 10VS 11P 2H 2D 6M

Figure B.21: KLM operators used by the actions required to specify the numbers of pedestrians to spawn, the spawn rate and the exit to be followed by each crowd group.

Run Simulation	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Go to Simulation tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Run simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Run	5	2	5	0	0	1	1	5K 2VS 5P 1M 1R
	7	4	7	0	0	3	1	7K 4VS 7P 3M 1R

Figure B.22: KLM operators used by the actions required to run and stop the simulation.

Tasks	K	VS	P	H	D	M	R
Create entrances	51	21	39	3	24	15	0
Create exits	51	21	39	3	24	15	0
Spawn/Divide crowd	84	30	33	6	6	18	0
Run Simulation	7	4	7	0	0	3	1
	193	76	118	12	54	51	1

Figure B.23: KLM summary of the Scenario 2 tasks.

Tasks	KLM
Create entrances	51K 21VS 39P 3H 24D 15M
Create exits	51K 21VS 39P 3H 24D 15M
Spawn/Divide crowd	84K 30VS 33P 6H 6D 18M
Run Simulation	7K 4VS 7P 3M 1R
	193K 76VS 118P 12H 54D 51M 1R

Figure B.24: KLM total of the Scenario 2 tasks.

Time (s)			
Setup	Compilation	Run	Total
330	88	63	481

Figure B.25: Scenario 2 Completion time divided into setup, compilation and run time.

Expert Scenario 2 Sketching

Create Entrance/Exit	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	4	0	0	0	1	1	0	4K 1D 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Rename Entrance/Exit	10	2	4	1	0	1	0	10K 2VS 4P 1H 1M
	16	3	6	1	1	4	0	16K 3VS 6P 1H 1D 4M

Figure B.26: KLM operators used by the actions required to create an entrance or exit.

Spawn/Divide crowd	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Click Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Entrance	1	0	1	0	0	1	0	1K 1P 1M
Set No. pedestrians	6	1	1	1	0	1	0	6K 1VS 1P 1H 1M
Set rate/time	4	1	1	1	0	1	0	4K 1VS 1P 1H 1M
Set Exit percentages	13	3	4	3	0	1	0	13K 3VS 4P 3H 1M
	25	6	8	5	0	5	0	25K 6VS 8P 5H 5M

Figure B.27: KLM operators used by the actions required to specify the numbers of pedestrians to spawn, the spawn rate and the exit to be followed by each crowd group.

Run simulation	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	2	2	2	0	0	2	0	2K 2VS 2P 2M

Figure B.28: KLM operators used by the actions required to run and stop the simulation.

Task	K	VS	P	H	D	M	R
Create entrance	48	9	18	3	3	12	0
Create exit	48	9	18	3	3	12	0
Spawn/Divide crowd	75	18	24	15	0	15	0
Run simulation	2	2	2	0	0	2	0
	173	38	62	21	6	41	0

Figure B.29: KLM summary of the Scenario 2 tasks.

Task	KLM
Create entrance	48K 9VS 18P 3H 3D 12M
Create exit	48K 9VS 18P 3H 3D 12M
Spawn/Divide crowd	75K 18VS 24P 15H 15M
Run simulation	2K 2VS 2P 2M
	173K 38VS 62P 21H 6D 41M

Figure B.30: KLM total of the Scenario 2 tasks.

Time (s)		
Setup	Run	Total
147	73	220

Figure B.31: Scenario 2 Completion time divided into setup and run time.

Expert Scenario 3 MassMotion

Elements	
2	Entrances
1	Exit
2	Areas
1	Barrier
3	Storyboards

Tasks
Create 2 entrances
Create exit
Create 2 areas
Spawn 900 pedestrians at a 3 peds/s rate
Crowd journey EntranceA -> AreaA->Exit
Crowd journey EntranceB -> AreaB->Exit
Run/Stop simulation
Crowd journey EntranceA -> AreaA->AreaB->Exit
Crowd journey EntranceB->Exit
Create a barrier blocking the exit
Run/Stop simulation

Figure B.32: Simulation elements included in Scenario 3 and the tasks needed to complete the scenario.

Create Entrance/Exit	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	1	0	0	0	2	1	0	1K 2D 1M
Go to Scene tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Position portal	3	3	8	0	5	1	0	3K 3VS 8P 5D 1M
Rename portal	11	2	3	1	1	1	0	11K 2VS 3P 1H 1D 1M
	17	7	13	1	8	5	0	17K 7VS 13P 1H 8D 5M

Figure B.33: KLM operators used by the actions required to create an entrance or exit.

Create Area	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	1	0	0	0	2	1	0	1K 2D 1M
Go to Scene tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Floor	1	1	1	0	0	1	0	1K 1VS 1P 1M
Position floor	2	2	4	0	2	1	0	2K 2VS 4P 2D 1M
Rename floor	11	2	3	1	1	1	0	11K 2VS 3P 1H 1D 1M
Select Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Position link	6	6	16	0	10	2	0	6K 6VS 16P 10D 2M
Rename link	22	4	6	2	2	2	0	22K 4VS 6P 2H 2D 2M
	45	17	32	3	17	10	0	45K 17VS 32P 3H 17D 10M

Figure B.34: KLM operators used by the actions required to create an area.

Spawn pedestrians	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Go to Activitites tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set No. pedestrians	7	1	1	1	1	1	0	7K 1VS 1P 1H 1D 1M
Set rate/time	6	1	1	1	1	1	0	6K 1VS 1P 1H 1D 1M
Select Origin	4	2	3	0	0	1	0	4K 2VS 3P 1M
Select Destination	5	2	4	0	0	1	0	5K 2VS 4P 1M
	24	8	11	2	2	6	0	24K 8VS 11P 2H 2D 6M

Figure B.35: KLM operators used by the actions required to specify the numbers of pedestrians to spawn and the spawn rate.

Action 1 Area	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Go to Activitites tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Action	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select list	2	2	2	0	0	1	0	2K 2VS 2P 1M
Add actions to list	2	1	1	0	0	1	0	2K 1VS 1P 1M
Configure actions	27	16	22	1	1	1	0	27K 16VS 22P 1H 1D 1M
Add action to journey	9	4	7	0	0	1	0	9K 4VS 7P 1M
	42	25	34	1	1	6	0	42K 25VS 34P 1H 1D 6M

Figure B.36: KLM operators used by the actions required to create an action to visit one area and wait inside for 10 seconds before continuing the journey.

Run Simulation	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Go to Simulation tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Run simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Run	5	2	5	0	0	1	1	5K 2VS 5P 1M 1R
	7	4	7	0	0	3	1	7K 4VS 7P 3M 1R

Figure B.37: KLM operators used by the actions required to run and stop the simulation.

Remove Action	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Go to Activitites tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Open journey	2	1	1	0	0	1	0	2K 1VS 1P 1M
Go to Actions tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Edit Action	3	2	3	0	0	1	0	3K 2VS 3P 1M
	7	5	6	0	0	4	0	7K 5VS 6P 4M

Figure B.38: KLM operators used by the actions required to delete an action.

Action 2 Areas	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Go to Activitites tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Action	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select list	2	2	2	0	0	1	0	2K 2VS 2P 1M
Add actions to list	1	1	1	0	0	1	0	1K 1VS 1P 1M
Configure actions	11	6	9	0	0	1	0	11K 6VS 9P 1M
Add action to journey	9	4	7	0	0	1	0	9K 4VS 7P 1M
	25	15	21	0	0	6	0	25K 15VS 21P 6M

Figure B.39: KLM operators used by the actions required to create an action to visit two areas and wait inside each for 10 seconds before continuing the journey.

Create Barrier	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	1	0	0	0	2	1	0	1K 2D 1M
Go to Scene tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Position Barrier	3	3	7	0	4	1	0	3K 3VS 7P 4D 1M
Rename barrier	11	2	3	1	1	1	0	11K 2VS 3P 1H 1D 1M
	17	7	12	1	7	5	0	17K 7VS 12P 1H 7D 5M

Figure B.40: KLM operators used by the actions required to create a barrier.

Task	K	VS	P	H	D	M	R
Create 2 entrances	34	14	26	2	16	10	0
Create exit	17	7	13	1	8	5	0
Create 2 areas	90	34	64	6	34	20	0
Spawn 900 pedestrians at a 3 peds/s rate	48	16	22	4	4	12	0
Crowd journey EntranceA -> AreaA->Exit	42	25	34	1	1	6	0
Crowd journey EntranceB -> AreaB->Exit	42	25	34	1	1	6	0
Run simulation	7	4	7	0	0	3	1
Crowd journey EntranceA -> AreaA->AreaB->Exit	25	15	21	0	0	6	0
Crowd journey EntranceB->Exit	7	5	6	0	0	4	0
Create a barrier blocking the exit	17	7	12	1	7	5	0
Run simulation	7	4	7	0	0	3	1
	336	156	246	16	71	80	2

Figure B.41: KLM summary of the Scenario 3 tasks.

Task	KLM
Create 2 entrances	34K 14VS 26P 2H 16D 10M
Create exit	17K 7VS 13P 1H 8D 5M
Create 2 areas	90K 34VS 64P 6H 34D 20M
Spawn 900 pedestrians at a 3 peds/s rate	48K 16VS 22P 4H 4D 12M
Crowd journey EntranceA -> AreaA->Exit	42K 25VS 34P 1H 1D 6M
Crowd journey EntranceB -> AreaB->Exit	42K 25VS 34P 1H 1D 6M
Run simulation	7K 4VS 7P 3M 1R
Crowd journey EntranceA -> AreaA->AreaB->Exit	25K 15VS 21P 6M
Crowd journey EntranceB->Exit	7K 5VS 6P 4M
Create a barrier blocking the exit	17K 7VS 12P 1H 7D 5M
Run simulation	7K 4VS 7P 3M 1R
	336K 156VS 246P 16H 71D 80M 2R

Figure B.42: KLM total of the Scenario 3 tasks.

Time (s)			
Setup	Compilation	Run	Total
566	165	86	817

Figure B.43: Scenario 3 Completion time divided into setup, compilation and run time.

Expert Scenario 3 Sketching

Create Entrance/Exit	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	4	0	0	0	1	1	0	4K 1D 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Rename Entrance/Exit	10	2	4	1	0	1	0	10K 2VS 4P 1H 1M
	16	3	6	1	1	4	0	16K 3VS 6P 1H 1D 4M

Figure B.44: KLM operators used by the actions required to create an entrance or exit.

Create Area	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	4	0	0	0	1	1	0	4K 1D 1M
Select Area	1	1	1	0	0	1	0	1K 1VS 1P 1M
Sketch Area	0	0	1	0	1	1	0	1P 1D 1M
Rename Area	10	2	4	1	0	1	0	10K 2VS 4P 1H 1M
	15	3	6	1	2	4	0	15K 3VS 6P 1H 2D 4M

Figure B.45: KLM operators used by the actions required to create an area.

Spawn pedestrians	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Click Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Entrance	1	0	1	0	0	1	0	1K 1P 1M
Set No. pedestrians	6	1	1	1	0	1	0	6K 1VS 1P 1H 1M
Set rate/time	4	1	1	1	0	1	0	4K 1VS 1P 1H 1M
	12	3	4	2	0	4	0	12K 3VS 4P 2H 4M

Figure B.46: KLM operators used by the actions required to specify the numbers of pedestrians to spawn and the spawn rate.

Storyboard 1 Area	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Move Camera	4	0	0	0	1	1	0	4K 1D 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Entrance	2	1	2	0	0	1	0	2K 1VS 2P 1M
Create Storyboard	4	1	4	0	0	1	0	4K 1VS 4P 1M
Set percentage	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
	16	4	8	1	1	5	0	16K 4VS 8P 1H 1D 5M

Figure B.47: KLM operators used by the actions required to create a storyboard to visit one area and wait inside for 10 seconds before continuing the journey.

Run simulation	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	2	2	2	0	0	2	0	2K 2VS 2P 2M

Figure B.48: KLM operators used by the actions required to run and stop the simulation.

Delete Storyboard	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Entrance	2	1	2	0	0	1	0	2K 1VS 2P 1M
Delete Storyboard	1	0	1	0	0	1	0	1K 1P 1M
	4	2	4	0	0	3	0	4K 2VS 4P 3M

Figure B.49: KLM operators used by the actions required to delete a storyboard.

Storyboard 2 Areas	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Move Camera	4	0	0	0	1	1	0	4K 1D 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Entrance	2	1	2	0	0	1	0	2K 1VS 2P 1M
Create Storyboard	6	2	6	0	0	1	0	6K 2VS 6P 1M
Set percentage	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
	18	5	10	1	1	5	0	18K 5VS 10P 1H 1D 5M

Figure B.50: KLM operators used by the actions required to create a storyboard to visit two areas and wait inside each for 10 seconds before continuing the journey.

Create Barrier	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	4	0	0	0	1	1	0	4K 1D 1M
Select Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Sketch Barrier	0	0	1	0	1	1	0	1P 1D 1M
Rename Barrier	10	2	4	1	0	1	0	10K 2VS 4P 1H 1M
	15	3	6	1	2	4	0	15K 3VS 6P 1H 2D 4M

Figure B.51: KLM operators used by the actions required to create a barrier.

Task	K	VS	P	H	D	M	R
Create 2 entrances	32	6	12	2	2	8	0
Create exit	16	3	6	1	1	4	0
Create 2 areas	30	6	12	2	4	8	0
Spawn 900 pedestrians at a 3 peds/s rate	24	6	8	4	0	8	0
Crowd journey EntranceA -> AreaA->Exit	16	4	8	1	1	5	0
Crowd journey EntranceB -> AreaB->Exit	16	4	8	1	1	5	0
Run simulation	2	2	2	0	0	2	0
Crowd journey EntranceA -> AreaA->AreaB->Exit	18	5	10	1	1	5	0
Crowd journey EntranceB->Exit	4	2	4	0	0	3	0
Create a barrier blocking the exit	15	3	6	1	2	4	0
Run simulation	2	2	2	0	0	2	0
	175	43	78	13	12	54	0

Figure B.52: KLM summary of the Scenario 3 tasks.

Task	KLM
Create 2 entrances	32K 6VS 12P 2H 2D 8M
Create exit	16K 3VS 6P 1H 1D 4M
Create 2 areas	30K 6VS 12P 2H 4D 8M
Spawn 900 pedestrians at a 3 peds/s rate	24K 6VS 8P 4H 8M
Crowd journey EntranceA -> AreaA->Exit	16K 4VS 8P 1H 1D 5M
Crowd journey EntranceB -> AreaB->Exit	16K 4VS 8P 1H 1D 5M
Run simulation	2K 2VS 2P 2M
Crowd journey EntranceA -> AreaA->AreaB->Exit	18K 5VS 10P 1H 1D 5M
Crowd journey EntranceB->Exit	4K 2VS 4P 3M
Create a barrier blocking the exit	15K 3VS 6P 1H 2D 4M
Run simulation	2K 2VS 2P 2M
	175K 43VS 78P 13H 12D 54M

Figure B.53: KLM total of the Scenario 3 tasks.

Time (s)		
Setup	Run	Total
141	98	239

Figure B.54: Scenario 3 Completion time divided into setup and run time.

Participant KLM Results

This section shows the KLM analysis of the user study participants completing the three scenarios. Figures B.55, B.56 and B.57 summarises the KLM results, completion time and metrics of the user study participants for all the scenarios in both systems. Then, detailed data for each participant and scenario is presented individually.

		Operators							
		K	VS	P	H	D	M	R	KLM
1	MassMotion	1154	492	695	39	310	216	6	1154K 492VS 695P 39H 310D 216M 6R
	Sketching	642	130	165	50	77	98	0	642K 130VS 165P 50H 77D 98M 0R
2	MassMotion	1674	642	994	41	475	249	6	1674K 642VS 994P 41H 475D 249M 6R
	Sketching	524	102	141	33	119	84	0	524K 102VS 141P 33H 119D 84M 0R
3	MassMotion	1526	404	650	35	365	219	6	1526K 404VS 650P 35H 365D 219M 6R
	Sketching	685	168	216	38	21	108	0	685K 168VS 216P 38H 21D 108M 0R
4	MassMotion	1202	354	596	24	301	207	6	1202K 354VS 596P 24H 301D 207M 6R
	Sketching	642	134	187	39	34	94	0	642K 134VS 187P 39H 34D 94M 0R
5	MassMotion	806	315	502	20	276	191	5	806K 315VS 502P 20H 276D 191M 5R
	Sketching	489	135	174	41	47	96	0	489K 135VS 174P 41H 47D 96M 0R
6	MassMotion	1356	344	534	24	301	201	6	1356K 344VS 534P 24H 301D 201M 6R
	Sketching	624	139	192	41	49	89	0	624K 139VS 192P 41H 49D 89M 0R
7	MassMotion	763	398	563	39	220	194	6	763K 398VS 563P 39H 220D 194M 6R
	Sketching	533	118	158	24	5	87	0	533K 118VS 158P 24H 5D 87M 0R
8	MassMotion	1082	350	475	36	245	173	5	1082K 350VS 475P 36H 245D 173M 5R
	Sketching	481	115	167	21	76	87	0	481K 115VS 167P 21H 76D 87M 0R
Avg	MassMotion	1195	412	626	32	312	206	5.8	1195K 412VS 626P 32H 311D 206M 5R
	Sketching	578	130	175	36	54	93	0	577K 130VS 175P 35H 53D 92M 0R

Figure B.55: KLM summary of participants for all the scenarios using MassMotion and the sketching system.

		Time (s)			
		Setup	Compile	Run	Total
1	MassMotion	30:39	08:26	03:31	42:36
	Sketching	09:07	00:00	07:44	16:51
2	MassMotion	41:34	09:53	03:53	55:20
	Sketching	11:11	00:00	07:23	18:34
3	MassMotion	37:37	07:37	04:07	49:21
	Sketching	11:15	00:00	09:57	21:12
4	MassMotion	32:15	07:07	03:06	42:28
	Sketching	12:41	00:00	07:27	20:08
5	MassMotion	28:31	05:43	04:28	38:42
	Sketching	11:26	00:00	07:06	18:32
6	MassMotion	29:51	06:17	08:28	44:36
	Sketching	13:00	00:00	07:08	20:08
7	MassMotion	40:46	07:22	02:41	50:49
	Sketching	11:11	00:00	07:29	18:40
8	MassMotion	23:54	06:50	02:12	32:56
	Sketching	08:48	00:00	07:32	16:20
Avg	MassMotion	33:08	07:24	04:03	44:36
	Sketching	11:05	00:00	07:43	18:48

Figure B.56: Participant completion time divided into setup, compilation and run for every scenario and system.

		Metrics						
		Distance (m)	Keystrokes	Left	Right	Middle	Double	Wheel
1	MassMotion	231.24	246	867	4	98	118	762
	Sketching	105.73	469	226	27	0	18	203
2	MassMotion	318.43	685	1224	20	131	125	1522
	Sketching	89.95	426	210	110	0	32	0
3	MassMotion	178.97	590	851	27	120	123	519
	Sketching	90.88	445	302	11	1	31	144
4	MassMotion	165.44	504	798	2	84	104	499
	Sketching	133.41	413	267	26	3	36	214
5	MassMotion	169.65	340	644	0	71	81	193
	Sketching	89.17	280	221	39	1	49	59
6	MassMotion	147.77	340	658	0	117	67	920
	Sketching	100.02	424	244	42	0	15	0
7	MassMotion	242.21	200	680	7	40	70	191
	Sketching	79.31	369	237	1	0	29	6
8	MassMotion	139.61	466	623	23	138	120	679
	Sketching	89.43	336	211	75	0	24	0
Avg	MassMotion	199.165	421.375	793.125	10.375	99.875	101	660.625
	Sketching	97.2375	395.25	239.75	41.375	0.625	29.25	78.25

Figure B.57: Participant metrics summary for every scenario and system.

Participant 1 Scenario 1 MassMotion

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:10	05:03	05:03	05:51	05:51	06:28	05:51	06:19
	06:28	07:37	07:37	08:26	08:26	08:49	08:26	08:44
	Total	06:02	Total	01:37	Total	01:00	Total	00:46
2	00:04	09:38	09:38	10:56	10:56	11:46	10:57	11:39
	Total	09:34	Total	01:18	Total	00:50	Total	00:42
3	00:39	11:01	11:01	12:24	12:24	12:56	12:26	12:56
	12:56	14:08	14:08	15:28	15:28	16:08	15:29	15:56
	16:08	18:28	18:28	19:45				
	19:45	20:54	20:54	22:25	22:25	22:54	22:27	22:54
	Total	15:03	Total	05:31	Total	01:41	Total	01:24

Figure B.58: Time (mm:ss) needed by participant 1 to complete each scenario in MassMotion. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	1	100
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.59: Scenario 1 tasks, number of attempts made by participant 1 and efficiency percentage using the MassMotion system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	9	0	0	0	3	1	0	9K 3D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	5	1	2	0	1	1	0	5K 1VS 2P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	3	1	3	0	4	1	0	3K 1VS 3P 4D 1M
Move Portal	6	2	4	0	2	1	0	6K 2VS 4P 2D 1M
Create Portal	5	1	1	0	0	1	0	5K 1VS 1P 1M
Rename Portal	32	3	4	1	1	1	0	32K 3VS 4P 1H 1D 1M
Move Portal	15	1	7	0	8	1	0	15K 1VS 7P 8D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	1	1	7	0	6	1	0	1K 1VS 7P 6D 1M
Move Portal	33	2	11	0	23	1	0	33K 2VS 11P 23D 1M
Rename Portal	12	4	5	2	1	1	0	12K 4VS 5P 2H 1D 1M
Select Activities Tab	4	1	1	0	0	1	0	4K 1VS 1P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	9	8	8	0	0	1	0	9K 8VS 8P 1M
Set rate	6	3	3	1	1	1	0	6K 3VS 3P 1H 1D 1M
Set Origin	5	4	4	0	0	1	0	5K 4VS 4P 1M
Set Destination	6	5	5	0	0	1	0	6K 5VS 5P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	12	3	3	0	4	1	0	12K 3VS 3P 4D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Scene Tab	6	1	1	0	2	1	0	6K 1VS 1P 2D 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Barrier	15	1	2	0	4	1	0	15K 1VS 2P 4D 1M
Scale Barrier	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Barrier	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Barrier	21	2	4	0	5	1	0	21K 2VS 4P 5D 1M
Select Sim Tab	8	2	2	0	0	1	0	8K 2VS 2P 1M
Compile Simulation	5	5	5	0	0	1	1	5K 5VS 5P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	9	0	0	0	3	1	0	9K 3D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	244	68	104	4	73	36	2	244K 68VS 104P 4H 73D 36M 2R

Figure B.60: Actions performed by participant 1 to complete Scenario 1 in MassMotion and the KLM operators required by each action.

Participant 1 Scenario 1 Sketching

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:04	00:54			00:54	01:45	00:54	01:43
	01:45	02:08			02:08	03:02	02:08	02:58
	Total	01:13	Total	00:00	Total	01:45	Total	01:39
2	00:12	04:43			04:43	06:41	04:43	06:40
	Total	04:31	Total	00:00	Total	01:58	Total	01:57
3	00:09	02:30			02:30	04:44	02:30	04:40
	04:44	05:46			05:46	07:33	05:46	07:33
	Total	03:23	Total	00:00	Total	04:01	Total	03:57

Figure B.61: Time (mm:ss) needed by participant 1 to complete each scenario in the sketching system. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	3	33.333333
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.62: Scenario 1 tasks, number of attempts made by participant 1 and efficiency percentage using the sketching system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Camera	9	0	0	1	0	1	0	9K 1H 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Click Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # pedestrians	17	2	4	6	0	3	0	17K 2VS 4P 6H 3M
Set rate	6	1	1	2	0	1	0	6K 1VS 1P 2H 1M
Set exit %	5	1	1	2	0	1	0	5K 1VS 1P 2H 1M
Rename Entrance	0	0	0	0	0	0	0	
Rename Exit	0	0	0	0	0	0	0	
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch simulation	35	1	1	3	4	1	0	35K 1VS 1P 3H 4D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Barrier	4	1	1	0	0	1	0	4K 1VS 1P 1M
Move Camera	7	0	0	0	0	1	0	7K 1M
Create Barrier	0	0	0	0	1	1	0	1D 1M
Start Simulation	9	1	1	0	0	1	0	9K 1VS 1P 1M
Watch simulation	10	0	0	1	0	1	0	10K 1H 1M
Stop Simulation	1	1	1	1	0	1	0	1K 1VS 1P 1H 1M
	111	14	18	16	5	21	0	111K 14VS 18P 16H 5D 21M

Figure B.63: Actions performed by participant 1 to complete Scenario 1 in the sketching system and the KLM operators required by each action.

Participant 1 Scenario 2 MassMotion

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	4	75
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.64: Scenario 2 tasks, number of attempts made by participant 1 and efficiency percentage using the MassMotion system.

Scenario 2	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	1	0	0	0	1	1	0	1K 1D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	5	1	1	0	0	1	0	5K 1VS 1P 1M
Move Portal	16	1	4	0	5	1	0	16K 1VS 4P 5D 1M
Scale Portal	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Move Portal	3	1	4	0	3	1	0	3K 1VS 4P 3D 1M
Create Portal	13	1	1	0	0	1	0	13K 1VS 1P 1M
Rename Portal	22	2	3	2	1	1	0	22K 2VS 3P 2H 1D 1M
Rename Portal	16	3	4	4	1	1	0	16K 3VS 4P 4H 1D 1M
Move Portal	16	1	7	0	10	1	0	16K 1VS 7P 10D 1M
Scale Portal	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	7	1	3	0	4	1	0	7K 1VS 3P 4D 1M
Rotate Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Create Portal	8	1	1	0	0	1	0	8K 1VS 1P 1M
Move Portal	11	1	5	0	6	1	0	11K 1VS 5P 6D 1M
Scale Portal	2	1	2	0	1	1	0	2K 1VS 2P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Move Portal	9	1	5	0	4	1	0	9K 1VS 5P 4D 1M
Rename Portal	15	4	4	3	1	1	0	15K 4VS 4P 3H 1D 1M
Create Portal	5	1	1	0	1	1	0	5K 1VS 1P 1D 1M
Move Portal	4	1	3	0	2	1	0	4K 1VS 3P 2D 1M
Scale Portal	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	3	1	4	0	3	1	0	3K 1VS 4P 3D 1M
Rename Portal	23	4	5	3	1	1	0	23K 4VS 5P 3H 1D 1M
Create Portal	7	1	1	0	3	1	0	7K 1VS 1P 3D 1M
Move Portal	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Scale Portal	6	1	6	0	6	1	0	6K 1VS 6P 6D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M

Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Portal	17	4	5	2	1	1	0	17K 4VS 5P 2H 1D 1M
Create Portal	3	1	1	0	2	1	0	3K 1VS 1P 2D 1M
Move Portal	7	1	3	0	4	1	0	7K 1VS 3P 4D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Portal	13	4	5	2	1	1	0	13K 4VS 5P 2H 1D 1M
Select Activities Tab	7	3	3	0	0	1	0	7K 3VS 3P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Rate	4	1	1	1	1	1	0	4K 1VS 1P 1H 1D 1M
Set Origin	8	3	3	0	0	1	0	8K 3VS 3P 1M
Set Destination	10	7	7	0	0	1	0	10K 7VS 7P 1M
Set # Agents	23	5	9	1	1	1	0	23K 5VS 9P 1H 1D 1M
Set Origin	6	3	3	0	0	1	0	6K 3VS 3P 1M
Set Destination	5	5	5	0	0	1	0	5K 5VS 5P 1M
Create Journey	2	2	2	0	0	1	0	2K 2VS 2P 1M
Set Destination	12	8	8	0	0	1	0	12K 8VS 8P 1M
Set Origin	9	6	6	0	0	1	0	9K 6VS 6P 1M
Set Destination	8	5	5	0	0	1	0	8K 5VS 5P 1M
Set # Agents	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Rate	5	2	2	1	1	1	0	5K 2VS 2P 1H 1D 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Rate	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Origin	4	3	3	0	0	1	0	4K 3VS 3P 1M
Set Destination	9	6	6	0	0	1	0	9K 6VS 6P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	29	3	3	0	9	1	0	29K 3VS 3P 9D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	415	134	211	23	110	70	1	415K 134VS 211P 23H 110D 70M 1R

Figure B.65 (cont.): Actions performed by participant 1 to complete Scenario 2 in MassMotion and the KLM operators required by each action.

Participant 1 Scenario 2 Sketching

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	4	75
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.66: Scenario 2 tasks, number of attempts made by participant 1 and efficiency percentage using the sketching system.

Scenario 2	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	11	1	1	0	0	1	0	11K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	18	1	2	1	0	1	0	18K 1VS 2P 1H 1M
Select Entrance/Exit	6	1	1	0	0	1	0	6K 1VS 1P 1M
Create Entrance/Exit	7	0	1	0	0	1	0	7K 1P 1M
Select Select	4	1	1	0	0	1	0	4K 1VS 1P 1M
Rename Entrance	13	1	2	1	0	1	0	13K 1VS 2P 1H 1M
Set # pedestrians	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Select Entrance/Exit	17	3	4	0	0	1	0	17K 3VS 4P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	11	2	3	2	0	1	0	11K 2VS 3P 2H 1M
Select Entrance/Exit	20	1	1	0	0	1	0	20K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	13	2	3	1	0	1	0	13K 2VS 3P 1H 1M
Select Entrance/Exit	10	1	1	0	0	1	0	10K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	19	2	3	2	0	1	0	19K 2VS 3P 2H 1M
Set # pedestrians	6	3	3	2	0	1	0	6K 3VS 3P 2H 1M
Set rate	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Set exit %	17	4	4	4	0	1	0	17K 4VS 4P 4H 1M
Select Select	3	1	1	0	1	1	0	3K 1VS 1P 1D 1M
Set exit %	20	4	5	3	0	1	0	20K 4VS 5P 3H 1M
Move Camera	25	1	1	0	2	1	0	25K 1VS 1P 2D 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set rate	10	1	2	1	0	1	0	10K 1VS 2P 1H 1M
Set # pedestrians	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Set exit %	13	4	4	3	0	1	0	13K 4VS 4P 3H 1M
Move Camera	3	1	2	0	1	1	0	3K 1VS 2P 1D 1M
Start Simulation	1	1	1	0	1	1	0	1K 1VS 1P 1D 1M
Watch Simulation	38	1	1	0	30	1	0	38K 1VS 1P 30D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	315	48	63	24	35	38	0	315K 48VS 63P 24H 35D 38M

Figure B.67: Actions performed by participant 1 to complete Scenario 2 in the sketching system and the KLM operators required by each action.

Participant 1 Scenario 3 MassMotion

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	4	50
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	1	100
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	2	50
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.68: Scenario 3 tasks, number of attempts made by participant 1 and efficiency percentage using the MassMotion system.

Scenario 3	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Move Camera	4	0	0	0	1	1	0	4K 1D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	3	1	2	0	2	1	0	3K 1VS 2P 2D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	5	0	4	0	5	1	0	5K 4P 5D 1M
Move Portal	3	1	3	0	3	2	0	3K 1VS 3P 3D 2M
Rename Portal	20	4	5	1	1	1	0	20K 4VS 5P 1H 1D 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	2	1	5	0	6	1	0	2K 1VS 5P 6D 1M
Move Portal	3	1	3	0	3	1	0	3K 1VS 3P 3D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Portal	21	4	5	1	1	1	0	21K 4VS 5P 1H 1D 1M
Create Portal	8	1	1	0	4	1	0	8K 1VS 1P 4D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	4	2	6	0	4	1	0	4K 2VS 6P 4D 1M
Move Portal	1	1	4	0	4	1	0	1K 1VS 4P 4D 1M
Rename Portal	12	4	5	1	1	1	0	12K 4VS 5P 1H 1D 1M
Create Floor	5	1	1	0	1	1	0	5K 1VS 1P 1D 1M
Move Floor	5	2	6	0	4	1	0	5K 2VS 6P 4D 1M
Rotate Floor	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Floor	13	4	5	1	1	1	0	13K 4VS 5P 1H 1D 1M
Create Link	3	1	1	0	1	1	0	3K 1VS 1P 1D 1M
Move Link	5	1	4	0	6	1	0	5K 1VS 4P 6D 1M
Rotate Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Link	2	1	2	0	2	1	0	2K 1VS 2P 2D 1M
Scale Link	2	0	3	0	3	1	0	2K 3P 3D 1M
Move Link	4	1	4	0	5	1	0	4K 1VS 4P 5D 1M
Rotate Link	11	1	2	0	4	1	0	11K 1VS 2P 4D 1M
Move Floor	7	3	4	0	2	1	0	7K 3VS 4P 2D 1M
Rotate Link	5	2	3	0	3	1	0	5K 2VS 3P 3D 1M
Delete Link	4	4	4	0	0	1	0	4K 4VS 4P 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Link	2	2	3	0	1	1	0	2K 2VS 3P 1D 1M
Move Link	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Link	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Move Link	5	1	3	0	2	1	0	5K 1VS 3P 2D 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rotate Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Link	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Create Floor	6	1	1	0	1	1	0	6K 1VS 1P 1D 1M
Move Floor	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Floor	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Floor	11	3	4	1	1	1	0	11K 3VS 4P 1H 1D 1M
Create Link	2	1	1	0	0	1	0	2K 1VS 1P 1M
Scale Link	2	2	4	0	2	1	0	2K 2VS 4P 2D 1M
Move Link	3	1	2	0	2	1	0	3K 1VS 2P 2D 1M
Move Floor	2	2	3	0	1	1	0	2K 2VS 3P 1D 1M
Rotate Link	4	4	5	0	1	1	0	4K 4VS 5P 1D 1M
Move Link	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M

Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Link	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Move Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rotate Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Link	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Select Activities Tab	5	1	1	0	1	1	0	5K 1VS 1P 1D 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set Origin	5	4	4	0	0	1	0	5K 4VS 4P 1M
Set Destination	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set Destination	3	2	2	0	0	1	0	3K 2VS 2P 1M
Create Action	5	5	5	0	0	1	0	5K 5VS 5P 1M
Define Actions	37	32	32	0	0	1	0	37K 32VS 32P 1M
Set Action	10	9	9	0	0	1	0	10K 9VS 9P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Rate	4	1	1	1	1	1	0	4K 1VS 1P 1H 1D 1M
Set Origin	4	3	3	0	0	1	0	4K 3VS 3P 1M
Set Destination	4	3	3	0	0	1	0	4K 3VS 3P 1M
Set Action	6	6	6	0	0	1	0	6K 6VS 6P 1M
Set # Agents	6	3	3	1	1	1	0	6K 3VS 3P 1H 1D 1M
Set Rate	9	2	1	1	1	1	0	9K 2VS 1P 1H 1D 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	0	4K 4VS 4P 1M
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	3	3	3	0	0	1	0	3K 3VS 3P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Activities Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Action	3	2	2	0	0	1	0	3K 2VS 2P 1M
Define Actions	33	29	29	0	0	1	0	33K 29VS 29P 1M
Set Action	8	6	6	0	0	1	0	8K 6VS 6P 1M
Select Sim Tab	8	6	6	0	0	1	0	8K 6VS 6P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Modify Actions	34	26	27	1	1	1	0	34K 26VS 27P 1H 1D 1M
Modify Actions	5	3	3	1	1	1	0	5K 3VS 3P 1H 1D 1M
Remove Action	10	6	6	1	1	1	0	10K 6VS 6P 1H 1D 1M
Select Scene Tab	6	5	5	0	0	1	0	6K 5VS 5P 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Barrier	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Scale Barrier	1	1	6	0	5	1	0	1K 1VS 6P 5D 1M
Rotate Barrier	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Barrier	6	1	2	0	2	1	0	6K 1VS 2P 2D 1M
Define Actions	0	0	0	0	0	1	0	1M
Select Sim Tab	2	2	2	0	0	1	0	2K 2VS 2P 1M
Compile Simulation	5	5	5	0	0	1	1	5K 5VS 5P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Activities Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Modify Actions	15	9	10	0	1	1	0	15K 9VS 10P 1D 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	10	0	0	0	0	1	0	10K 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	495	290	380	12	127	110	3	495K 290VS 380P 12H 127D 110M 3R

Figure B.69 (cont.): Actions performed by participant 1 to complete Scenario 3 in MassMotion and the KLM operators required by each action.

Participant 1 Scenario 3 Sketching

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	2	100
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	2	50
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	2	50
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.70: Scenario 3 tasks, number of attempts made by participant 1 and efficiency percentage using the sketching system.

Scenario 3	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	2	1	1	0	1	1	0	2K 1VS 1P 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Entrance/Exit	4	1	1	0	3	1	0	4K 1VS 1P 3D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Area	2	1	1	0	1	1	0	2K 1VS 1P 1D 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Select Select	2	2	2	0	0	1	0	2K 2VS 2P 1M
Set rate	4	2	2	1	0	1	0	4K 2VS 2P 1H 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Set rate	4	1	1	1	0	1	0	4K 1VS 1P 1H 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Rename Entrance	16	4	4	1	0	1	0	16K 4VS 4P 1H 1M
Rename Entrance	15	2	3	1	0	1	0	15K 2VS 3P 1H 1M
Rename Exit	9	2	3	1	0	1	0	9K 2VS 3P 1H 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Storyboard	12	4	7	1	0	1	0	12K 4VS 7P 1H 1M
Create Storyboard	10	4	7	1	0	1	0	10K 4VS 7P 1H 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	9	2	2	0	0	1	0	9K 2VS 2P 1M
Stop Simulation	4	4	4	0	0	1	0	4K 4VS 4P 1M
Set behaviour	4	3	4	0	0	1	0	4K 3VS 4P 1M
Set behaviour	4	3	4	0	0	1	0	4K 3VS 4P 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	25	1	1	0	8	1	0	25K 1VS 1P 8D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Delete Storyboard	3	3	3	0	0	1	0	3K 3VS 3P 1M
Delete Storyboard	3	3	3	0	0	1	0	3K 3VS 3P 1M
Create Storyboard	8	1	4	0	0	1	0	8K 1VS 4P 1M
Select Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	11	0	0	0	3	1	0	11K 3D 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Change Storyboard	13	10	10	1	0	1	0	13K 10VS 10P 1H 1M
Watch Simulation	27	0	0	0	19	1	0	27K 19D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	216	68	84	10	37	39	0	216K 68VS 84P 10H 37D 39M

Figure B.71: Actions performed by participant 1 to complete Scenario 3 in the sketching system and the KLM operators required by each action.

Participant 2 Scenario 1 MassMotion

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:05	06:44	06:44	07:19				
	07:19	07:51	07:51	08:53	08:53	09:55	08:55	09:50
	09:55	11:04	11:04	11:58	11:58	12:36	12:04	12:28
	Total	08:20	Total	02:31	Total	01:40	Total	01:19
2	00:04	11:07	11:07	11:56				
	11:56	12:56	12:56	14:18	14:18	15:24	14:21	15:20
	Total	12:03	Total	02:11	Total	01:06	Total	00:59
3	00:04	16:03	16:03	17:19				
	17:19	19:22	19:22	20:42	20:42	21:12	20:44	21:08
	21:12	24:00	24:00	25:03				
	25:03	25:24	25:24	26:56	26:56	27:33	26:57	27:25
	Total	21:11	Total	05:11	Total	01:07	Total	00:52

Figure B.72: Time (mm:ss) needed by participant 2 to complete each scenario in MassMotion. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	2	50
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.73: Scenario 1 tasks, number of attempts made by participant 2 and efficiency percentage using the MassMotion system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	21	0	0	0	1	1	0	21K 1D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	7	1	1	0	0	1	0	7K 1VS 1P 1M
Move Portal	7	1	2	0	1	1	0	7K 1VS 2P 1D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Portal	44	6	6	1	12	1	0	44K 6VS 6P 1H 12D 1M
Create Portal	31	1	1	0	3	1	0	31K 1VS 1P 3D 1M
Move Portal	26	6	14	1	15	1	0	26K 6VS 14P 1H 15D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Move Portal	10	2	6	0	3	1	0	10K 2VS 6P 3D 1M
Scale Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	3	1	4	0	3	1	0	3K 1VS 4P 3D 1M
Rename Portal	10	4	4	1	1	1	0	10K 4VS 4P 1H 1D 1M
Move Portal	15	1	5	0	4	1	0	15K 1VS 5P 4D 1M
Scale Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	8	3	7	0	5	1	0	8K 3VS 7P 5D 1M
Rotate Portal	3	2	3	0	1	1	0	3K 2VS 3P 1D 1M
Select Activities Tab	9	1	1	0	0	1	0	9K 1VS 1P 1M
Create Action	13	10	10	0	0	1	0	13K 10VS 10P 1M
Create Journey	3	1	1	0	0	1	0	3K 1VS 1P 1M
Set # Agents	8	4	4	0	0	1	0	8K 4VS 4P 1M
Set Rate	4	1	1	1	1	1	0	4K 1VS 1P 1H 1D 1M
Set Origin	6	5	5	0	0	1	0	6K 5VS 5P 1M
Set Destination	16	14	14	0	0	1	0	16K 14VS 14P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	7	7	7	0	0	1	1	7K 7VS 7P 1M 1R
Delete Action	7	5	5	1	0	1	0	7K 5VS 5P 1H 1M
Select Sim Tab	2	2	2	0	0	1	0	2K 2VS 2P 1M
Compile Simulation	4	4	4	0	0	1	0	4K 4VS 4P 1M
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	3	1	1	0	0	1	0	3K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Barrier	6	1	5	0	6	1	0	6K 1VS 5P 6D 1M
Scale Barrier	3	1	4	0	8	1	0	3K 1VS 4P 8D 1M
Move Barrier	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rotate Barrier	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Barrier	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Barrier	4	1	2	0	2	1	0	4K 1VS 2P 2D 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	7	4	4	0	0	1	1	7K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	9	2	2	0	3	1	0	9K 2VS 2P 3D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	316	112	159	5	83	49	2	316K 112VS 159P 5H 83D 49M 2R

Figure B.74: Actions performed by participant 2 to complete Scenario 1 in MassMotion and the KLM operators required by each action.

Participant 2 Scenario 1 Sketching

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:53	03:03			03:03	03:54	03:03	03:51
	03:54	04:38			04:38	05:32	04:38	05:29
	Total	02:54	Total	00:00	Total	01:45	Total	01:39
2	00:05	04:12			04:12	06:09	04:12	06:06
	Total	04:07	Total	00:00	Total	01:57	Total	01:54
3	00:20	03:28			03:28	05:08	03:28	05:01
	05:08	06:10			06:10	08:11	06:10	08:03
	Total	04:10	Total	00:00	Total	03:41	Total	03:26

Figure B.75: Time (mm:ss) needed by participant 2 to complete each scenario in the sketching system. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	2	50
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	3	33.333333
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.76: Scenario 1 tasks, number of attempts made by participant 2 and efficiency percentage using the sketching system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	15	1	1	0	8	1	0	15K 1VS 1P 8D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	23	1	2	1	0	1	0	23K 1VS 2P 1H 1M
Set # pedestrians	4	1	1	1	0	1	0	4K 1VS 1P 1H 1M
Set rate	5	2	2	1	0	1	0	5K 2VS 2P 1H 1M
Select Entrance/Exit	12	1	1	0	15	1	0	12K 1VS 1P 15D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	2	1	1	0	0	1	0	2K 1VS 1P 1M
Rename Exit	8	1	2	1	0	1	0	8K 1VS 2P 1H 1M
Set # pedestrians	12	3	4	1	9	1	0	12K 3VS 4P 1H 9D 1M
Set Exit %	7	1	1	1	0	1	0	7K 1VS 1P 1H 1M
Start Simulation	3	2	2	0	3	1	0	3K 2VS 2P 3D 1M
Watch simulation	7	1	1	0	11	1	0	7K 1VS 1P 11D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Barrier	2	1	1	0	7	1	0	2K 1VS 1P 7D 1M
Create Barrier	3	0	1	0	5	1	0	3K 1P 5D 1M
Start Simulation	5	1	1	0	0	1	0	5K 1VS 1P 1M
Watch simulation	0	0	0	0	0	0	0	
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	113	20	26	6	58	19	0	113K 20VS 26P 6H 58D 19M

Figure B.77: Actions performed by participant 2 to complete Scenario 1 in the sketching system and the KLM operators required by each action.

Participant 2 Scenario 2 MassMotion

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	6	50
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.78: Scenario 2 tasks, number of attempts made by participant 2 and efficiency percentage using the MassMotion system.

Scenario 2	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	8	0	0	0	3	1	0	8K 3D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Rename Portal	22	5	5	1	0	1	0	22K 5VS 5P 1H 1M
Create Portal	23	1	1	0	4	1	0	23K 1VS 1P 4D 1M
Move Portal	2	2	3	0	1	1	0	2K 2VS 3P 1D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	5	1	9	0	8	1	0	5K 1VS 9P 8D 1M
Scale Portal	4	1	2	0	1	1	0	4K 1VS 2P 1D 1M
Move Portal	5	1	13	0	14	1	0	5K 1VS 13P 14D 1M
Scale Portal	9	1	5	0	4	1	0	9K 1VS 5P 4D 1M
Rotate Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Scale Portal	2	2	3	0	1	1	0	2K 2VS 3P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Delete Portal	2	1	1	1	0	1	0	2K 1VS 1P 1H 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Portal	2	2	5	0	3	1	0	2K 2VS 5P 3D 1M
Move Portal	1	1	7	0	6	1	0	1K 1VS 7P 6D 1M
Scale Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	5	1	8	0	9	1	0	5K 1VS 8P 9D 1M
Rename Portal	25	6	6	2	0	1	0	25K 6VS 6P 2H 1M
Create Portal	17	1	1	0	10	1	0	17K 1VS 1P 10D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	14	2	6	0	4	1	0	14K 2VS 6P 4D 1M
Scale Portal	4	4	5	0	1	1	0	4K 4VS 5P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	2	1	5	0	4	1	0	2K 1VS 5P 4D 1M

Rename Portal	25	4	4	1	0	1	0	25K 4VS 4P 1H 1M
Create Portal	22	1	1	0	7	1	0	22K 1VS 1P 7D 1M
Move Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	3	1	3	0	2	1	0	3K 1VS 3P 2D 1M
Scale Portal	8	2	5	0	3	1	0	8K 2VS 5P 3D 1M
Move Portal	2	2	3	0	1	1	0	2K 2VS 3P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	2	1	2	0	1	1	0	2K 1VS 2P 1D 1M
Rename Portal	28	7	7	1	0	1	0	28K 7VS 7P 1H 1M
Create Portal	20	1	1	0	3	1	0	20K 1VS 1P 3D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Move Portal	8	3	5	0	3	1	0	8K 3VS 5P 3D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Portal	13	3	3	1	0	1	0	13K 3VS 3P 1H 1M
Create Portal	19	1	1	0	3	1	0	19K 1VS 1P 3D 1M
Move Portal	19	3	11	0	16	1	0	19K 3VS 11P 16D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	3	1	2	0	1	1	0	3K 1VS 2P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Portal	19	4	4	1	0	1	0	19K 4VS 4P 1H 1M
Select Activities Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set Origin	9	9	9	0	0	1	0	9K 9VS 9P 1M
Set Destination	15	11	11	0	0	1	0	15K 11VS 11P 1M
Set # Agents	9	4	4	2	0	1	0	9K 4VS 4P 2H 1M
Set Rate	9	6	6	1	1	1	0	9K 6VS 6P 1H 1D 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	6	6	6	0	0	1	1	6K 6VS 6P 1M 1R
Move Portal	67	6	8	0	10	1	0	67K 6VS 8P 10D 1M
Select Sim Tab	9	3	3	0	0	1	0	9K 3VS 3P 1M
Compile Simulation	4	4	4	0	0	1	0	4K 4VS 4P 1M
Run Simulation	2	2	2	0	0	1	0	2K 2VS 2P 1M
Watch Simulation	12	0	0	0	3	1	0	12K 3D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	516	149	257	11	163	73	1	516K 149VS 257P 11H 163D 73M 1R

Figure B.79 (cont.): Actions performed by participant 2 to complete Scenario 2 in MassMotion and the KLM operators required by each action.

Participant 2 Scenario 2 Sketching

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.80: Scenario 2 tasks, number of attempts made by participant 2 and efficiency percentage using the sketching system.

Scenario 2	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	2	1	1	0	4	1	0	2K 1VS 1P 4D 1M
Create Entrance/Exit	3	0	1	0	4	1	0	3K 1P 4D 1M
Create Entrance/Exit	9	0	1	0	5	1	0	9K 1P 5D 1M
Create Entrance/Exit	8	0	1	0	2	1	0	8K 1P 2D 1M
Select Entrance/Exit	9	1	1	0	4	1	0	9K 1VS 1P 4D 1M
Create Entrance/Exit	4	0	1	0	3	1	0	4K 1P 3D 1M
Create Entrance/Exit	10	0	1	0	0	1	0	10K 1P 1M
Create Entrance/Exit	11	0	1	0	5	1	0	11K 1P 5D 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	11	2	3	1	0	1	0	11K 2VS 3P 1H 1M
Rename Exit	19	2	3	1	1	1	0	19K 2VS 3P 1H 1D 1M
Rename Exit	15	2	3	0	1	1	0	15K 2VS 3P 1D 1M
Rename Entrance	26	0	1	1	2	1	0	26K 1P 1H 2D 1M
Set rate	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Set Exit %	12	4	4	3	0	3	0	12K 4VS 4P 3H 3M
Rename Entrance	36	1	2	1	6	1	0	36K 1VS 2P 1H 6D 1M
Set rate	2	1	1	0	0	1	0	2K 1VS 1P 1M
Set # pedestrians	9	2	2	2	0	1	0	9K 2VS 2P 2H 1M
Set Exit %	9	4	4	0	0	1	0	9K 4VS 4P 1M
Rename Entrance	21	1	2	1	5	1	0	21K 1VS 2P 1H 5D 1M
Set rate	2	1	1	0	0	1	0	2K 1VS 1P 1M
Set # pedestrians	9	2	2	2	0	1	0	9K 2VS 2P 2H 1M
Start Simulation	8	1	1	0	9	1	0	8K 1VS 1P 9D 1M
Watch Simulation	4	1	1	0	0	1	0	4K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	253	32	44	14	51	28	0	253K 32VS 44P 14H 51D 28M

Figure B.81: Actions performed by participant 2 to complete Scenario 2 in the sketching system and the KLM operators required by each action.

Participant 2 Scenario 3 MassMotion

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	4	50
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	2	50
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	2	50
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.82: Scenario 3 tasks, number of attempts made by participant 2 and efficiency percentage using the MassMotion system.

Scenario 3	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	16	0	0	0	0	1	0	16K 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	3	1	3	0	2	1	0	3K 1VS 3P 2D 1M
Move Portal	6	1	3	0	2	1	0	6K 1VS 3P 2D 1M
Rename Portal	21	4	5	1	1	1	0	21K 4VS 5P 1H 1D 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	4	2	6	0	4	1	0	4K 2VS 6P 4D 1M
Move Portal	8	2	4	0	3	1	0	8K 2VS 4P 3D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	2	2	6	0	4	1	0	2K 2VS 6P 4D 1M
Rename Portal	25	5	5	1	0	1	0	25K 5VS 5P 1H 1M
Create Portal	12	1	1	0	7	1	0	12K 1VS 1P 7D 1M
Scale Portal	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Move Portal	5	1	3	0	3	1	0	5K 1VS 3P 3D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	5	0	4	1	0	1K 1VS 5P 4D 1M
Rename Portal	10	4	4	1	1	1	0	10K 4VS 4P 1H 1D 1M
Create Floor	3	1	1	0	0	1	0	3K 1VS 1P 1M
Move Floor	3	1	7	0	6	1	0	3K 1VS 7P 6D 1M
Rotate Floor	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Floor	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Link	5	1	5	0	4	1	0	5K 1VS 5P 4D 1M
Scale Link	3	1	4	0	3	1	0	3K 1VS 4P 3D 1M
Move Link	5	1	4	0	6	1	0	5K 1VS 4P 6D 1M
Rotate Link	2	1	2	0	1	1	0	2K 1VS 2P 1D 1M
Move Link	4	1	6	0	5	1	0	4K 1VS 6P 5D 1M
Rotate Link	24	4	4	0	0	1	0	24K 4VS 4P 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Link	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Scale Link	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Move Link	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Link	2	2	3	0	1	1	0	2K 2VS 3P 1D 1M
Move Link	1	1	14	0	13	1	0	1K 1VS 14P 13D 1M
Rotate Link	7	2	7	0	4	1	0	7K 2VS 7P 4D 1M
Move Link	2	1	9	0	7	1	0	2K 1VS 9P 7D 1M
Move Floor	4	1	4	1	1	1	0	4K 1VS 4P 1H 1D 1M
Move Link	8	2	12	0	9	1	0	8K 2VS 12P 9D 1M
Delete Link	7	2	2	3	0	1	0	7K 2VS 2P 3H 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Link	1	1	6	0	5	1	0	1K 1VS 6P 5D 1M
Scale Link	1	1	5	0	4	1	0	1K 1VS 5P 4D 1M
Rotate Link	2	2	3	0	1	1	0	2K 2VS 3P 1D 1M
Move Link	6	1	4	0	3	1	0	6K 1VS 4P 3D 1M
Scale Link	0	0	3	0	3	1	0	3P 3D 1M
Move Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rotate Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Link	2	1	4	1	3	1	0	2K 1VS 4P 1H 3D 1M
Create Floor	7	1	1	0	0	1	0	7K 1VS 1P 1M
Rename Floor	14	4	5	1	2	1	0	14K 4VS 5P 1H 2D 1M
Delete Floor	5	3	3	1	0	1	0	5K 3VS 3P 1H 1M
Create Floor	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Floor	3	1	6	0	5	1	0	3K 1VS 6P 5D 1M
Rotate Floor	4	1	2	0	1	1	0	4K 1VS 2P 1D 1M
Move Floor	3	2	3	0	1	1	0	3K 2VS 3P 1D 1M
Create Link	4	4	4	0	0	1	0	4K 4VS 4P 1M
Move Link	3	1	6	0	5	1	0	3K 1VS 6P 5D 1M

Scale Link	15	1	5	0	7	1	0	15K 1VS 5P 7D 1M
Move Link	4	1	4	0	3	1	0	4K 1VS 4P 3D 1M
Move Floor	2	1	3	0	1	1	0	2K 1VS 3P 1D 1M
Move Link	3	1	4	0	2	1	0	3K 1VS 4P 2D 1M
Create Link	2	1	1	0	0	1	0	2K 1VS 1P 1M
Move Link	1	1	6	0	5	1	0	1K 1VS 6P 5D 1M
Scale Link	3	1	3	0	2	1	0	3K 1VS 3P 2D 1M
Move Link	3	1	3	0	2	1	0	3K 1VS 3P 2D 1M
Rename Link	31	3	3	3	0	1	0	31K 3VS 3P 3H 1M
Rename Link	32	4	5	1	0	1	0	32K 4VS 5P 1H 1M
Rename Floor	10	2	3	1	1	1	0	10K 2VS 3P 1H 1D 1M
Rename Link	17	2	2	1	1	1	0	17K 2VS 2P 1H 1D 1M
Rename Link	8	3	5	0	1	1	0	8K 3VS 5P 1D 1M
Rename Floor	6	2	3	1	0	1	0	6K 2VS 3P 1H 1M
Rename Link	8	4	4	0	2	1	0	8K 4VS 4P 2D 1M
Rename Link	17	2	3	1	1	1	0	17K 2VS 3P 1H 1D 1M
Rename Link	9	4	4	1	0	1	0	9K 4VS 4P 1H 1M
Select Activities Tab	4	2	2	0	0	1	0	4K 2VS 2P 1M
Create Action	1	1	1	0	0	1	0	1K 1VS 1P 1M
Define Actions	57	38	38	1	0	1	0	57K 38VS 38P 1H 1M
Create Action	1	1	1	0	0	1	0	1K 1VS 1P 1M
Define Actions	43	26	26	1	1	1	0	43K 26VS 26P 1H 1D 1M
Create Journey	3	1	1	0	0	1	0	3K 1VS 1P 1M
Set # Agents	7	3	3	1	0	1	0	7K 3VS 3P 1H 1M
Set Rate	19	1	1	1	0	1	0	19K 1VS 1P 1H 1M
Set Origin	5	5	5	0	0	1	0	5K 5VS 5P 1M
Set Destination	5	5	5	0	0	1	0	5K 5VS 5P 1M
Set Action	26	24	24	0	0	1	0	26K 24VS 24P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	8	6	7	0	0	1	1	8K 6VS 7P 1M 1R
Run Simulation	0	1	1	0	1	1	0	1VS 1P 1D 1M
Modify Actions	35	24	25	0	6	1	0	35K 24VS 25P 6D 1M
Modify Journey	12	10	10	0	0	1	0	12K 10VS 10P 1M
Create Journey	3	2	2	0	0	1	0	3K 2VS 2P 1M
Set # Agents	9	6	6	1	1	1	0	9K 6VS 6P 1H 1D 1M
Set Rate	15	1	1	1	0	1	0	15K 1VS 1P 1H 1M
Set Origin	4	4	4	0	0	1	0	4K 4VS 4P 1M
Set Destination	6	6	6	0	0	1	0	6K 6VS 6P 1M
Set Action	7	7	7	0	0	1	0	7K 7VS 7P 1M
Modify Journey	15	12	12	0	0	1	0	15K 12VS 12P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	8	1	1	0	0	1	0	8K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Modify Journey	18	15	15	0	0	1	0	18K 15VS 15P 1M
Modify Journey	8	6	6	0	0	1	0	8K 6VS 6P 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Barrier	3	3	8	0	5	1	0	3K 3VS 8P 5D 1M
Move Barrier	1	1	9	0	8	1	0	1K 1VS 9P 8D 1M
Scale Barrier	3	3	4	0	1	1	0	3K 3VS 4P 1D 1M
Move Barrier	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rotate Barrier	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Barrier	1	1	9	0	8	1	0	1K 1VS 9P 8D 1M
Scale Barrier	11	2	4	0	2	1	0	11K 2VS 4P 2D 1M
Rotate Barrier	2	2	3	0	1	1	0	2K 2VS 3P 1D 1M
Move Barrier	14	2	5	0	10	1	0	14K 2VS 5P 10D 1M
Select Sim Tab	13	1	1	0	8	1	0	13K 1VS 1P 8D 1M
Compile Simulation	5	5	5	0	0	1	1	5K 5VS 5P 1M 1R
Modify Journey	11	10	10	0	0	1	0	11K 10VS 10P 1M
Compile Simulation	4	4	4	0	0	1	0	4K 4VS 4P 1M
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	842	381	578	25	229	127	3	842K 381VS 578P 25H 229D 127M 3R

Figure B.83 (cont.): Actions performed by participant 2 to complete Scenario 3 in MassMotion and the KLM operators required by each action.

Participant 2 Scenario 3 Sketching

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	2	100
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	1	100
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.84: Scenario 3 tasks, number of attempts made by participant 2 and efficiency percentage using the sketching system.

Scenario 3	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Select Entrance/Exit	5	1	1	0	1	1	0	5K 1VS 1P 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Area	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	17	1	2	1	0	1	0	17K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Rename Entrance	14	1	2	1	0	1	0	14K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Rename Exit	8	1	2	1	0	1	0	8K 1VS 2P 1H 1M
Rename Area	10	1	2	1	0	1	0	10K 1VS 2P 1H 1M
Set behaviour	2	2	2	0	0	1	0	2K 2VS 2P 1M
Rename Area	10	1	2	1	0	1	0	10K 1VS 2P 1H 1M
Set behaviour	3	3	3	0	0	1	0	3K 3VS 3P 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Storyboard	12	4	7	1	2	1	0	12K 4VS 7P 1H 2D 1M
Create Storyboard	13	4	7	1	2	1	0	13K 4VS 7P 1H 2D 1M
Select Area	3	2	3	0	0	1	0	3K 2VS 3P 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	4	1	1	0	0	1	0	4K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	0	0	0	0	1	1	0	1D 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Delete Storyboard	3	3	3	0	0	1	0	3K 3VS 3P 1M
Create Storyboard	11	2	6	1	0	1	0	11K 2VS 6P 1H 1M
Delete Storyboard	3	3	3	0	0	1	0	3K 3VS 3P 1M
Create Storyboard	7	2	4	1	0	1	0	7K 2VS 4P 1H 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	2	2	2	0	1	1	0	2K 2VS 2P 1D 1M
Stop Simulation	1	1	1	0	1	1	0	1K 1VS 1P 1D 1M
	158	50	71	13	10	37	0	158K 50VS 71P 13H 10D 37M

Figure B.85: Actions performed by participant 2 to complete Scenario 3 in the sketching system and the KLM operators required by each action.

Participant 3 Scenario 1 MassMotion

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:23	09:44	09:44	10:35	10:35	11:03	10:37	11:01
	11:03	12:28	12:28	13:19	13:19	14:06	13:21	13:27
	Total	10:46	Total	01:42	Total	01:15	Total	00:30
2	00:13	10:41	10:41	11:59	11:59	12:39	12:06	12:38
	Total	10:28	Total	01:18	Total	00:40	Total	00:32
3	00:06	12:37	12:37	14:01	14:01	15:12	14:04	15:12
	15:12	17:32	17:32	19:14	19:14	19:34	19:17	19:34
	19:34	21:06	21:06	22:37	22:37	23:18	22:39	23:12
	Total	16:23	Total	04:37	Total	02:12	Total	01:58

Figure B.86: Time (mm:ss) needed by participant 3 to complete each scenario in MassMotion. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	3	33.333333
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	1	100
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.87: Scenario 1 tasks, number of attempts made by participant 3 and efficiency percentage using the MassMotion system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	9	0	0	0	1	1	0	9K 1D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Portal	21	3	3	1	0	1	0	21K 3VS 3P 1H 1M
Create Portal	46	5	5	0	3	1	0	46K 5VS 5P 3D 1M
Move Portal	4	2	3	0	1	1	0	4K 2VS 3P 1D 1M
Delete Portal	19	3	3	0	0	1	0	19K 3VS 3P 1M
Move Portal	3	0	3	0	2	1	0	3K 3P 2D 1M
Rename Portal	13	3	3	1	1	1	0	13K 3VS 3P 1H 1D 1M
Rotate Portal	2	1	3	0	1	1	0	2K 1VS 3P 1D 1M
Move Portal	15	1	6	0	6	1	0	15K 1VS 6P 6D 1M
Scale Portal	2	1	4	0	3	1	0	2K 1VS 4P 3D 1M
Move Portal	3	0	5	0	5	1	0	3K 5P 5D 1M
Scale Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Rotate Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Move Portal	4	0	3	0	3	1	0	4K 3P 3D 1M
Rotate Portal	3	0	1	0	1	1	0	3K 1P 1D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Delete Portal	4	1	1	0	0	1	0	4K 1VS 1P 1M
Create Portal	6	4	5	0	0	1	0	6K 4VS 5P 1M
Scale Portal	2	0	3	0	3	1	0	2K 3P 3D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Scale Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Rotate Portal	12	0	1	0	4	1	0	12K 1P 4D 1M
Move Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Create Portal	26	1	1	0	7	1	0	26K 1VS 1P 7D 1M
Rename Portal	17	4	4	2	1	1	0	17K 4VS 4P 2H 1D 1M
Rename Portal	8	3	3	1	1	1	0	8K 3VS 3P 1H 1D 1M
Move Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Scale Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Move Portal	16	0	4	0	7	1	0	16K 4P 7D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	37	0	9	0	15	1	0	37K 9P 15D 1M
Select Activities Tab	6	2	2	0	1	1	0	6K 2VS 2P 1D 1M
Create Journey	1	1	1	0	2	1	0	1K 1VS 1P 2D 1M
Set # Agents	2	1	1	0	3	1	0	2K 1VS 1P 3D 1M
Set Rate	5	1	1	1	1	1	0	5K 1VS 1P 1H 1D 1M
Move Portal	39	35	35	0	0	1	0	39K 35VS 35P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	3	3	3	0	0	1	0	3K 3VS 3P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Barrier	9	0	2	0	4	1	0	9K 2P 4D 1M
Scale Barrier	13	0	3	0	3	1	0	13K 3P 3D 1M
Rotate Barrier	5	0	1	0	1	1	0	5K 1P 1D 1M
Select Sim Tab	15	2	2	0	6	1	0	15K 2VS 2P 6D 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	0	0	0	0	0	1	0	1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	409	94	158	6	103	55	2	409K 94VS 158P 6H 103D 55M 2R

Figure B.88: Actions performed by participant 3 to complete Scenario 1 in MassMotion and the KLM operators required by each action.

Participant 3 Scenario 1 Sketching

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:05	01:41			01:41	02:28	01:41	02:27
	02:28	03:04			03:04	03:56	03:04	03:54
	Total	02:12	Total	00:00	Total	01:39	Total	01:36
2	00:05	04:06			04:06	06:24	04:06	05:57
	Total	04:01	Total	00:00	Total	02:18	Total	01:51
3	00:05	02:40			02:40	05:13	02:40	05:13
	05:13	05:57			05:57	07:25	05:57	07:25
	07:25	09:08			09:08	11:07	09:08	10:55
	Total	05:02	Total	00:00	Total	06:00	Total	05:48

Figure B.89: Time (mm:ss) needed by participant 3 to complete each scenario in the sketching system. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	1	100
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.90: Scenario 1 tasks, number of attempts made by participant 3 and efficiency percentage using the sketching system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	12	0	0	0	1	1	0	12K 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	14	1	2	1	0	1	0	14K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	7	2	2	1	0	1	0	7K 2VS 2P 1H 1M
Select Entrance/Exit	18	1	1	0	0	1	0	18K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	14	3	4	1	0	1	0	14K 3VS 4P 1H 1M
Start Simulation	6	6	6	0	0	1	0	6K 6VS 6P 1M
Watch simulation	4	1	1	0	0	1	0	4K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Barrier	2	2	2	0	0	1	0	2K 2VS 2P 1M
Create Barrier	19	0	0	0	6	1	0	19K 6D 1M
Start Simulation	7	1	1	0	4	1	0	7K 1VS 1P 4D 1M
Watch simulation	20	0	0	0	1	1	0	20K 1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	132	22	26	4	12	18	0	132K 22VS 26P 4H 12D 18M

Figure B.91: Actions performed by participant 3 to complete Scenario 1 in the sketching system and the KLM operators required by each action.

Participant 3 Scenario 2 MassMotion

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.92: Scenario 2 tasks, number of attempts made by participant 3 and efficiency percentage using the MassMotion system.

Scenario 2	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Move Camera	10	0	0	0	2	1	0	10K 2D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	4	0	2	0	2	1	0	4K 2P 2D 1M
Scale Portal	5	0	3	0	3	1	0	5K 3P 3D 1M
Move Portal	27	4	7	0	11	1	0	27K 4VS 7P 11D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Scale Portal	2	0	3	0	3	1	0	2K 3P 3D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	2	0	6	0	6	1	0	2K 6P 6D 1M
Scale Portal	13	2	4	0	5	1	0	13K 2VS 4P 5D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rotate Portal	2	2	2	0	0	1	0	2K 2VS 2P 1M
Move Portal	17	0	2	0	6	1	0	17K 2P 6D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	4	0	3	0	3	1	0	4K 3P 3D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rename Portal	32	5	5	1	1	1	0	32K 5VS 5P 1H 1D 1M
Duplicate Portal	2	2	2	0	0	1	0	2K 2VS 2P 1M
Move Portal	12	0	4	0	6	1	0	12K 4P 6D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	4	0	1	0	1	1	0	4K 1P 1D 1M
Rotate Portal	6	0	2	0	3	1	0	6K 2P 3D 1M
Rename Portal	11	3	3	1	0	1	0	11K 3VS 3P 1H 1M
Duplicate Portal	14	7	7	0	2	1	0	14K 7VS 7P 2D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rename Portal	22	4	4	1	3	1	0	22K 4VS 4P 1H 3D 1M
Move Portal	2	0	3	0	3	1	0	2K 3P 3D 1M
Duplicate Portal	8	3	3	0	2	1	0	8K 3VS 3P 2D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	13	0	4	0	5	1	0	13K 4P 5D 1M
Rename Portal	11	3	3	1	0	1	0	11K 3VS 3P 1H 1M
Duplicate Portal	2	2	2	0	0	1	0	2K 2VS 2P 1M
Move Portal	7	0	9	0	10	1	0	7K 9P 10D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	8	0	4	0	5	1	0	8K 4P 5D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rename Portal	14	2	2	2	2	1	0	14K 2VS 2P 2H 2D 1M
Duplicate Portal	2	2	2	0	0	1	0	2K 2VS 2P 1M
Rename Portal	13	3	3	1	0	1	0	13K 3VS 3P 1H 1M
Move Portal	5	0	5	0	9	1	0	5K 5P 9D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Scale Portal	8	0	4	0	5	1	0	8K 4P 5D 1M
Move Portal	2	0	5	0	5	1	0	2K 5P 5D 1M
Select Activities Tab	17	4	4	0	4	1	0	17K 4VS 4P 4D 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	4	1	1	1	1	1	0	4K 1VS 1P 1H 1D 1M
Set Rate	8	1	2	1	0	1	0	8K 1VS 2P 1H 1M
Set Origin	11	8	8	0	0	1	0	11K 8VS 8P 1M
Set Destination	10	6	6	0	0	1	0	10K 6VS 6P 1M
Set # Agents	8	5	5	1	0	1	0	8K 5VS 5P 1H 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	2	2	2	0	0	1	0	2K 2VS 2P 1M
Watch Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	386	81	164	10	125	61	1	386K 81VS 164P 10H 125D 61M 1R

Figure B.93: Actions performed by participant 3 to complete Scenario 2 in MassMotion and the KLM operators required by each action.

Participant 3 Scenario 2 Sketching

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.94: Scenario 2 tasks, number of attempts made by participant 3 and efficiency percentage using the sketching system.

Scenario 2	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	13	1	1	0	0	1	0	13K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	18	1	2	1	0	1	0	18K 1VS 2P 1H 1M
Set rate	7	3	3	1	0	1	0	7K 3VS 3P 1H 1M
Set # pedestrians	8	2	3	1	0	1	0	8K 2VS 3P 1H 1M
Select Entrance/Exit	12	1	1	0	1	1	0	12K 1VS 1P 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	26	2	3	2	0	1	0	26K 2VS 3P 2H 1M
Set rate	3	1	1	0	0	1	0	3K 1VS 1P 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Select Entrance/Exit	4	1	1	0	4	1	0	4K 1VS 1P 4D 1M
Create Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	14	1	2	1	0	1	0	14K 1VS 2P 1H 1M
Set rate	3	1	1	0	0	1	0	3K 1VS 1P 1M
Set # pedestrians	6	2	2	0	1	1	0	6K 2VS 2P 1D 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	11	2	3	1	0	1	0	11K 2VS 3P 1H 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	4	0	1	0	0	1	0	4K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	15	2	3	1	0	1	0	15K 2VS 3P 1H 1M
Select Entrance/Exit	3	1	1	0	0	1	0	3K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	0	1	0	0	1	0	1K 1P 1M
Rename Exit	13	2	3	1	0	1	0	13K 2VS 3P 1H 1M
Set Exit %	37	10	12	1	0	1	0	37K 10VS 12P 1H 1M
Set Exit %	18	4	5	3	0	1	0	18K 4VS 5P 3H 1M
Set Exit %	21	4	5	3	0	1	0	21K 4VS 5P 3H 1M
Start Simulation	3	1	1	0	0	1	0	3K 1VS 1P 1M
Watch Simulation	4	1	1	0	0	1	0	4K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	263	54	71	17	6	36	0	263K 54VS 71P 17H 6D 36M

Figure B.95: Actions performed by participant 3 to complete Scenario 2 in the sketching system and the KLM operators required by each action.

Participant 3 Scenario 3 MassMotion

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	2	100
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	1	200
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	2	50
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.96: Scenario 3 tasks, number of attempts made by participant 3 and efficiency percentage using the MassMotion system.

Scenario 3	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	4	0	0	0	1	1	0	4K 1D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Scale Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Rotate Portal	6	0	1	0	1	1	0	6K 1P 1D 1M
Scale Portal	7	0	1	0	2	1	0	7K 1P 2D 1M
Move Portal	4	0	2	0	2	1	0	4K 2P 2D 1M
Create Portal	8	0	3	0	5	1	0	8K 3P 5D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rename Portal	26	3	3	1	3	1	0	26K 3VS 3P 1H 3D 1M
Duplicate Portal	2	2	2	0	0	1	0	2K 2VS 2P 1M
Rename Portal	17	3	3	1	0	1	0	17K 3VS 3P 1H 1M
Move Portal	17	0	4	0	8	1	0	17K 4P 8D 1M
Duplicate Portal	11	4	4	0	1	1	0	11K 4VS 4P 1D 1M
Rename Portal	12	3	3	1	0	1	0	12K 3VS 3P 1H 1M
Move Portal	4	0	5	0	6	1	0	4K 5P 6D 1M
Rotate Portal	5	0	1	0	3	1	0	5K 1P 3D 1M
Move Portal	6	0	6	0	6	1	0	6K 6P 6D 1M
Rotate Portal	3	0	1	0	3	1	0	3K 1P 3D 1M
Move Portal	4	0	3	0	3	1	0	4K 3P 3D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	3	0	3	0	3	1	0	3K 3P 3D 1M
Create Floor	10	1	1	0	5	1	0	10K 1VS 1P 5D 1M
Move Floor	4	0	5	0	5	1	0	4K 5P 5D 1M
Rename Floor	19	3	3	1	0	1	0	19K 3VS 3P 1H 1M
Scale Floor	2	0	1	0	1	1	0	2K 1P 1D 1M
Duplicate Floor	2	2	2	0	0	1	0	2K 2VS 2P 1M
Move Floor	2	0	2	0	2	1	0	2K 2P 2D 1M
Rename Floor	11	3	3	1	0	1	0	11K 3VS 3P 1H 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Link	9	0	4	0	6	1	0	9K 4P 6D 1M
Scale Link	7	0	2	0	2	1	0	7K 2P 2D 1M
Move Link	18	0	6	0	9	1	0	18K 6P 9D 1M
Move Floor	3	1	2	0	1	1	0	3K 1VS 2P 1D 1M
Rename Link	33	3	3	1	1	1	0	33K 3VS 3P 1H 1D 1M
Duplicate Link	2	2	2	0	0	1	0	2K 2VS 2P 1M
Move Link	2	0	1	0	1	1	0	2K 1P 1D 1M
Rename Link	11	3	3	1	0	1	0	11K 3VS 3P 1H 1M
Duplicate Link	4	3	3	0	0	1	0	4K 3VS 3P 1M
Move Link	8	0	3	0	4	1	0	8K 3P 4D 1M
Rotate Link	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Link	7	0	3	0	4	1	0	7K 3P 4D 1M
Move Floor	3	0	2	0	1	1	0	3K 2P 1D 1M
Move Link	5	0	2	0	1	1	0	5K 2P 1D 1M
Create Link	6	0	5	0	3	1	0	6K 5P 3D 1M
Rename Link	17	3	4	1	0	1	0	17K 3VS 4P 1H 1M
Duplicate Link	2	2	2	0	0	1	0	2K 2VS 2P 1M

Rename Link	15	4	4	1	0	1	0	15K 4VS 4P 1H 1M
Move Link	3	0	2	0	2	1	0	3K 2P 2D 1M
Rotate Link	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Link	2	0	1	0	1	1	0	2K 1P 1D 1M
Select Activities Tab	9	1	1	0	2	1	0	9K 1VS 1P 2D 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	7	1	1	1	0	1	0	7K 1VS 1P 1H 1M
Set Rate	12	1	1	1	0	1	0	12K 1VS 1P 1H 1M
Set Origin	4	3	3	0	0	1	0	4K 3VS 3P 1M
Set Destination	5	4	4	0	0	1	0	5K 4VS 4P 1M
Create Action	1	1	1	0	0	1	0	1K 1VS 1P 1M
Define Actions	54	33	33	1	1	1	0	54K 33VS 33P 1H 1D 1M
Create Action	1	1	1	0	0	1	0	1K 1VS 1P 1M
Define Actions	46	26	26	2	0	1	0	46K 26VS 26P 2H 1M
Set Action	10	7	7	0	0	1	0	10K 7VS 7P 1M
Create Journey	21	2	2	1	1	1	0	21K 2VS 2P 1H 1D 1M
Set # Agents	4	2	2	1	1	1	0	4K 2VS 2P 1H 1D 1M
Set Rate	6	1	1	1	0	1	0	6K 1VS 1P 1H 1M
Set Origin	4	3	3	0	0	1	0	4K 3VS 3P 1M
Set Destination	4	3	3	0	0	1	0	4K 3VS 3P 1M
Set Action	7	6	6	0	0	1	0	7K 6VS 6P 1M
Modify Journey	10	5	5	1	1	1	0	10K 5VS 5P 1H 1D 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	12	3	3	0	1	1	0	12K 3VS 3P 1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Remove Action	19	15	15	0	0	1	0	19K 15VS 15P 1M
Modify Actions	42	31	31	0	0	1	0	42K 31VS 31P 1M
Modify Actions	9	3	3	1	0	1	0	9K 3VS 3P 1H 1M
Set Action	4	3	3	0	0	1	0	4K 3VS 3P 1M
Select Sim Tab	2	1	1	0	0	1	0	2K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	2	1	1	0	0	1	0	2K 1VS 1P 1M
Watch Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Barrier	2	0	1	0	1	1	0	2K 1P 1D 1M
Scale Barrier	6	0	2	0	2	1	0	6K 2P 2D 1M
Move Barrier	6	0	1	0	1	1	0	6K 1P 1D 1M
Scale Barrier	2	0	1	0	1	1	0	2K 1P 1D 1M
Rotate Barrier	2	0	1	0	1	1	0	2K 1P 1D 1M
Scale Barrier	5	0	7	0	7	1	0	5K 7P 7D 1M
Move Barrier	3	0	1	0	2	1	0	3K 1P 2D 1M
Rotate Barrier	2	0	1	0	1	1	0	2K 1P 1D 1M
Scale Barrier	7	0	2	0	4	1	0	7K 2P 4D 1M
Select Sim Tab	2	1	1	0	0	1	0	2K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	10	1	1	0	3	1	0	10K 1VS 1P 3D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	731	229	328	19	137	103	3	731K 229VS 328P 19H 137D 103M 3R

Figure B.97 (cont.): Actions performed by participant 3 to complete Scenario 3 in MassMotion and the KLM operators required by each action.

Participant 3 Scenario 3 Sketching

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	3	66.66666667
Create and rename the exits	1	100
Create and rename the areas	2	100
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	2	50
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.98: Scenario 3 tasks, number of attempts made by participant 3 and efficiency percentage using the sketching system.

Scenario 3	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	12	1	2	1	0	1	0	12K 1VS 2P 1H 1M
Set rate	3	2	2	1	0	1	0	3K 2VS 2P 1H 1M
Set # pedestrians	8	2	2	1	0	1	0	8K 2VS 2P 1H 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	14	1	2	1	0	1	0	14K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Select Entrance/Exit	2	1	1	0	0	1	0	2K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	7	2	3	1	0	1	0	7K 2VS 3P 1H 1M
Select Area	6	3	3	0	0	1	0	6K 3VS 3P 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Area	9	1	2	1	0	1	0	9K 1VS 2P 1H 1M
Set behaviour	3	3	3	0	0	1	0	3K 3VS 3P 1M
Select Area	9	1	1	0	0	1	0	9K 1VS 1P 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Area	16	1	2	1	0	1	0	16K 1VS 2P 1H 1M
Set behaviour	3	3	3	0	0	1	0	3K 3VS 3P 1M
Select Storyboard	2	1	1	0	0	1	0	2K 1VS 1P 1M
Create Storyboard	14	6	9	1	0	1	0	14K 6VS 9P 1H 1M
Create Storyboard	12	4	7	1	0	1	0	12K 4VS 7P 1H 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	26	3	4	0	0	1	0	26K 3VS 4P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Delete Entrance	4	2	4	0	0	1	0	4K 2VS 4P 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	20	1	2	1	0	1	0	20K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	7	2	2	1	0	1	0	7K 2VS 2P 1H 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Storyboard	18	4	7	1	0	1	0	18K 4VS 7P 1H 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	10	10	10	0	0	1	0	10K 10VS 10P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Delete Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Delete Storyboard	3	3	3	0	0	1	0	3K 3VS 3P 1M
Select Barrier	4	4	4	0	0	1	0	4K 4VS 4P 1M
Create Barrier	0	0	0	0	1	1	0	1D 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Storyboard	11	4	6	1	0	1	0	11K 4VS 6P 1H 1M
Create Storyboard	13	4	7	1	0	1	0	13K 4VS 7P 1H 1M
Start Simulation	2	2	2	0	0	1	0	2K 2VS 2P 1M
Watch Simulation	19	0	0	0	0	1	0	19K 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	290	92	119	17	3	54	0	290K 92VS 119P 17H 3D 54M

Figure B.99: Actions performed by participant 3 to complete Scenario 3 in the sketching system and the KLM operators required by each action.

Participant 4 Scenario 1 MassMotion

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:05	04:16	04:16	05:05	05:05	05:44	05:06	05:30
	05:44	06:34	06:34	07:52	07:52	08:14	07:53	08:14
	08:14	08:39	08:39	09:28	09:28	09:47	09:29	09:45
	Total	05:26	Total	02:56	Total	01:20	Total	01:01
2	00:04	09:25	09:25	10:44	10:44	11:25	10:45	11:22
	Total	09:21	Total	01:19	Total	00:41	Total	00:37
3	00:05	13:07	13:07	14:29	14:29	15:02	14:30	15:01
	15:02	19:28	19:28	20:58	20:58	21:30	20:59	21:27
	Total	17:28	Total	02:52	Total	01:05	Total	00:59

Figure B.100: Time (mm:ss) needed by participant 4 to complete each scenario in MassMotion. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	1	100
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.101: Scenario 1 tasks, number of attempts made by participant 4 and efficiency percentage using the MassMotion system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	18	0	0	0	6	1	0	18K 6D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	7	1	3	0	2	1	0	7K 1VS 3P 2D 1M
Scale Portal	1	0	5	0	7	1	0	1K 5P 7D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	3	0	1	0	1	1	0	3K 1P 1D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Create Portal	11	1	1	0	11	1	0	11K 1VS 1P 11D 1M
Move Portal	9	0	4	0	6	1	0	9K 4P 6D 1M
Scale Portal	5	0	4	0	4	1	0	5K 4P 4D 1M
Rotate Portal	3	0	1	0	1	1	0	3K 1P 1D 1M
Move Portal	4	0	1	0	1	1	0	4K 1P 1D 1M
Rename Portal	10	4	4	1	1	1	0	10K 4VS 4P 1H 1D 1M
Rename Portal	17	5	5	1	0	1	0	17K 5VS 5P 1H 1M
Select Activities Tab	7	2	2	0	1	1	0	7K 2VS 2P 1D 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	11	6	8	1	0	1	0	11K 6VS 8P 1H 1M
Set Rate	6	1	1	1	0	1	0	6K 1VS 1P 1H 1M
Set Origin	4	4	4	0	0	1	0	4K 4VS 4P 1M
Set Destination	5	5	5	0	0	1	0	5K 5VS 5P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	10	2	2	0	3	1	0	10K 2VS 2P 3D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Scene Tab	4	1	1	0	2	1	0	4K 1VS 1P 2D 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Barrier	3	0	3	0	4	1	0	3K 3P 4D 1M
Scale Barrier	4	0	3	0	4	1	0	4K 3P 4D 1M
Move Barrier	1	0	1	0	1	1	0	1K 1P 1D 1M
Select Sim Tab	12	1	1	0	0	1	0	12K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	24	0	0	0	1	1	0	24K 1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Barrier	10	0	2	0	5	1	0	10K 2P 5D 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	4	4	4	0	0	1	0	4K 4VS 4P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	218	60	91	4	63	41	3	218K 60VS 91P 4H 63D 41M 3R

Figure B.102: Actions performed by participant 4 to complete Scenario 1 in MassMotion and the KLM operators required by each action.

Participant 4 Scenario 1 Sketching

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:08	02:02			02:02	03:01	02:02	02:49
	03:01	03:38			03:38	04:30	03:38	04:29
	Total	02:31	Total	00:00	Total	01:51	Total	01:38
2	00:20	05:24			05:24	07:18	05:24	07:17
	Total	05:04	Total	00:00	Total	01:54	Total	01:53
3	00:37	03:45			03:45	05:15	03:45	05:13
	05:15	07:13			07:13	09:25	07:13	09:35
	Total	05:06	Total	00:00	Total	03:42	Total	03:50

Figure B.103: Time (mm:ss) needed by participant 4 to complete each scenario in the sketching system. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	1	100
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.104: Scenario 1 tasks, number of attempts made by participant 4 and efficiency percentage using the sketching system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	7	1	1	0	0	1	0	7K 1VS 1P 1M
Create Entrance/Exit	4	0	1	0	0	1	0	4K 1P 1M
Select Select	3	3	3	0	0	1	0	3K 3VS 3P 1M
Rename Entrance	13	1	2	1	0	1	0	13K 1VS 2P 1H 1M
Set # pedestrians	12	3	3	2	0	1	0	12K 3VS 3P 2H 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	12	0	1	0	1	1	0	12K 1P 1D 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	16	2	3	1	0	1	0	16K 2VS 3P 1H 1M
Set rate	11	4	5	1	0	1	0	11K 4VS 5P 1H 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch simulation	11	1	1	0	0	1	0	11K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	7	0	0	0	3	1	0	7K 3D 1M
Start Simulation	2	1	1	0	0	1	0	2K 1VS 1P 1M
Watch simulation	10	0	0	0	2	1	0	10K 2D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	114	22	27	5	6	18	0	114K 22VS 27P 5H 6D 18M

Figure B.105: Actions performed by participant 4 to complete Scenario 1 in the sketching system and the KLM operators required by each action.

Participant 4 Scenario 2 MassMotion

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.106: Scenario 2 tasks, number of attempts made by participant 4 and efficiency percentage using the MassMotion system.

Scenario 2	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Move Camera	7	0	0	0	2	1	0	7K 2D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	7	0	10	0	11	1	0	7K 10P 11D 1M
Scale Portal	1	0	7	0	7	1	0	1K 7P 7D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	4	0	4	0	4	1	0	4K 4P 4D 1M
Scale Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rename Portal	38	3	3	1	0	1	0	38K 3VS 3P 1H 1M
Create Portal	7	1	1	0	4	1	0	7K 1VS 1P 4D 1M
Move Portal	1	0	3	0	3	1	0	1K 3P 3D 1M
Scale Portal	1	0	5	0	5	1	0	1K 5P 5D 1M
Move Portal	3	0	3	0	3	1	0	3K 3P 3D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rename Portal	15	4	4	1	0	1	0	15K 4VS 4P 1H 1M
Create Portal	7	1	1	0	3	1	0	7K 1VS 1P 3D 1M
Move Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Scale Portal	5	0	4	0	4	1	0	5K 4P 4D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	8	0	2	0	4	1	0	8K 2P 4D 1M
Rename Portal	13	3	3	1	0	1	0	13K 3VS 3P 1H 1M
Create Portal	8	1	1	0	2	1	0	8K 1VS 1P 2D 1M
Move Portal	3	0	3	0	3	1	0	3K 3P 3D 1M
Scale Portal	1	0	4	0	4	1	0	1K 4P 4D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	12	0	1	0	2	1	0	12K 1P 2D 1M
Rename Portal	12	3	3	1	0	1	0	12K 3VS 3P 1H 1M
Create Portal	12	1	1	0	3	1	0	12K 1VS 1P 3D 1M
Move Portal	1	0	4	0	4	1	0	1K 4P 4D 1M
Scale Portal	1	0	3	0	3	1	0	1K 3P 3D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	3	0	1	0	1	1	0	3K 1P 1D 1M
Move Portal	3	0	3	0	3	1	0	3K 3P 3D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	4	0	1	0	1	1	0	4K 1P 1D 1M
Rename Portal	20	3	3	1	0	1	0	20K 3VS 3P 1H 1M
Create Portal	5	1	1	0	2	1	0	5K 1VS 1P 2D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Portal	12	0	4	0	4	1	0	12K 4P 4D 1M
Move Portal	4	0	3	0	3	1	0	4K 3P 3D 1M
Rename Portal	30	3	3	1	0	1	0	30K 3VS 3P 1H 1M
Move Portal	15	0	3	0	2	1	0	15K 3P 2D 1M
Select Activities Tab	22	5	6	0	3	1	0	22K 5VS 6P 3D 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	13	2	2	2	0	1	0	13K 2VS 2P 2H 1M
Set Rate	9	2	3	1	0	1	0	9K 2VS 3P 1H 1M
Set Origin	12	12	12	0	0	1	0	12K 12VS 12P 1M
Set Destination	9	9	9	0	0	1	0	9K 9VS 9P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	5	2	2	0	1	1	0	5K 2VS 2P 1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	362	66	153	9	108	60	1	362K 66VS 153P 9H 108D 60M 1R

Figure B.107: Actions performed by participant 4 to complete Scenario 2 in MassMotion and the KLM operators required by each action.

Participant 4 Scenario 2 Sketching

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	4	75
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.108: Scenario 2 tasks, number of attempts made by participant 4 and efficiency percentage using the sketching system.

Scenario 2	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Create Entrance/Exit	4	0	2	0	0	1	0	4K 2P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Delete Entrance	3	1	3	0	0	1	0	3K 1VS 3P 1M
Rename Entrance	27	2	3	2	0	1	0	27K 2VS 3P 2H 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	14	2	3	1	0	1	0	14K 2VS 3P 1H 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	3	1	1	0	0	1	0	3K 1VS 1P 1M
Rename Exit	37	2	3	1	0	1	0	37K 2VS 3P 1H 1M
Select Exit	2	1	1	0	0	1	0	2K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	24	2	3	2	0	1	0	24K 2VS 3P 2H 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	23	0	1	0	12	1	0	23K 1P 12D 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	25	2	3	2	0	1	0	25K 2VS 3P 2H 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	35	2	3	1	0	1	0	35K 2VS 3P 1H 1M
Set # pedestrians	5	1	2	1	0	1	0	5K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set Exit %	15	4	4	3	0	1	0	15K 4VS 4P 3H 1M
Set rate	6	1	2	1	0	1	0	6K 1VS 2P 1H 1M
Set # pedestrians	7	1	1	1	0	1	0	7K 1VS 1P 1H 1M
Set Exit %	14	4	4	3	0	1	0	14K 4VS 4P 3H 1M
Set rate	18	2	4	1	9	1	0	18K 2VS 4P 1H 9D 1M
Set # pedestrians	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Set Exit %	13	4	4	3	0	1	0	13K 4VS 4P 3H 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	4	1	1	0	4	1	0	4K 1VS 1P 4D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	303	46	66	24	25	37	0	303K 46VS 66P 24H 25D 37M

Figure B.109: Actions performed by participant 4 to complete Scenario 2 in the sketching system and the KLM operators required by each action.

Participant 4 Scenario 3 MassMotion

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	2	100
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	1	100
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	2	50
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.110: Scenario 3 tasks, number of attempts made by participant 4 and efficiency percentage using the MassMotion system.

Scenario 3	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Move Camera	4	0	0	0	1	1	0	4K 1D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	4	0	1	0	1	1	0	4K 1P 1D 1M
Scale Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	10	0	3	0	3	1	0	10K 3P 3D 1M
Rename Portal	17	3	3	2	0	1	0	17K 3VS 3P 2H 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Rotate Portal	3	0	1	0	1	1	0	3K 1P 1D 1M
Scale Portal	5	0	4	0	4	1	0	5K 4P 4D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	3	0	2	0	2	1	0	3K 2P 2D 1M
Scale Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Move Portal	4	0	2	0	2	1	0	4K 2P 2D 1M
Rename Portal	34	3	3	0	0	1	0	34K 3VS 3P 1M
Scale Portal	16	0	2	0	5	1	0	16K 2P 5D 1M
Create Portal	5	1	1	0	2	1	0	5K 1VS 1P 2D 1M
Move Portal	4	0	2	0	4	1	0	4K 2P 4D 1M
Scale Portal	3	0	2	0	2	1	0	3K 2P 2D 1M
Rotate Portal	4	0	1	0	1	1	0	4K 1P 1D 1M
Move Portal	8	0	4	0	4	1	0	8K 4P 4D 1M
Rename Portal	13	3	3	1	0	1	0	13K 3VS 3P 1H 1M
Create Floor	4	2	2	0	1	1	0	4K 2VS 2P 1D 1M
Move Floor	5	0	2	0	3	1	0	5K 2P 3D 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Link	2	0	2	0	2	1	0	2K 2P 2D 1M
Scale Link	2	0	3	0	3	1	0	2K 3P 3D 1M
Rotate Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Link	12	0	3	0	3	1	0	12K 3P 3D 1M
Move Floor	10	0	4	0	3	1	0	10K 4P 3D 1M
Move Link	21	4	11	0	8	1	0	21K 4VS 11P 8D 1M
Move Floor	10	0	2	0	1	1	0	10K 2P 1D 1M
Move Link	2	0	3	0	2	1	0	2K 3P 2D 1M
Scale Link	4	0	3	0	3	1	0	4K 3P 3D 1M
Move Link	3	0	1	0	1	1	0	3K 1P 1D 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Link	4	0	4	0	3	1	0	4K 4P 3D 1M
Rotate Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Link	3	0	2	0	2	1	0	3K 2P 2D 1M
Move Link	3	0	2	0	2	1	0	3K 2P 2D 1M
Scale Link	4	0	1	0	1	1	0	4K 1P 1D 1M
Create Floor	9	2	5	0	1	1	0	9K 2VS 5P 1D 1M
Rename Floor	12	3	3	1	0	1	0	12K 3VS 3P 1H 1M
Rename Floor	12	3	3	1	0	1	0	12K 3VS 3P 1H 1M
Move Floor	4	0	7	0	8	1	0	4K 7P 8D 1M
Create Link	4	1	1	0	1	1	0	4K 1VS 1P 1D 1M
Rotate Link	3	0	1	0	1	1	0	3K 1P 1D 1M
Scale Link	1	0	2	0	2	1	0	1K 2P 2D 1M
Move Link	6	0	4	0	4	1	0	6K 4P 4D 1M
Move Floor	5	0	2	0	1	1	0	5K 2P 1D 1M
Move Link	3	0	3	0	2	1	0	3K 3P 2D 1M

Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rotate Link	2	0	1	0	1	1	0	2K 1P 1D 1M
Scale Link	2	0	5	0	5	1	0	2K 5P 5D 1M
Move Link	5	0	3	0	3	1	0	5K 3P 3D 1M
Rotate Link	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Link	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Link	2	0	1	0	1	1	0	2K 1P 1D 1M
Select Activities Tab	4	1	1	0	0	1	0	4K 1VS 1P 1M
Create Action	1	1	1	0	0	1	0	1K 1VS 1P 1M
Define Actions	54	49	50	0	0	1	0	54K 49VS 50P 1M
Create Action	1	1	1	0	0	1	0	1K 1VS 1P 1M
Define Actions	31	29	31	0	0	1	0	31K 29VS 31P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set Rate	14	1	1	1	0	1	0	14K 1VS 1P 1H 1M
Set # Agents	7	1	1	1	0	1	0	7K 1VS 1P 1H 1M
Set Origin	4	4	4	0	0	1	0	4K 4VS 4P 1M
Set Destination	4	4	4	0	0	1	0	4K 4VS 4P 1M
Set Action	10	9	9	0	0	1	0	10K 9VS 9P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	6	1	1	1	0	1	0	6K 1VS 1P 1H 1M
Set Rate	6	1	1	1	0	1	0	6K 1VS 1P 1H 1M
Set Origin	4	4	4	0	0	1	0	4K 4VS 4P 1M
Set Destination	5	4	5	0	0	1	0	5K 4VS 5P 1M
Set Action	7	7	7	0	0	1	0	7K 7VS 7P 1M
Select Sim Tab	2	1	1	0	1	1	0	2K 1VS 1P 1D 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	4	2	2	0	0	1	0	4K 2VS 2P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Activities Tab	4	2	2	0	1	1	0	4K 2VS 2P 1D 1M
Modify Actions	34	29	29	0	0	1	0	34K 29VS 29P 1M
Delete Action	6	1	1	0	0	1	0	6K 1VS 1P 1M
Set Action	14	12	13	0	0	1	0	14K 12VS 13P 1M
Delete Journey	5	2	2	0	0	1	0	5K 2VS 2P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	7	2	2	1	0	1	0	7K 2VS 2P 1H 1M
Set Rate	7	1	2	1	0	1	0	7K 1VS 2P 1H 1M
Set Origin	5	4	4	0	0	1	0	5K 4VS 4P 1M
Set Destination	8	6	7	0	0	1	0	8K 6VS 7P 1M
Select Scene Tab	9	1	1	0	4	1	0	9K 1VS 1P 4D 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Barrier	7	0	3	0	1	1	0	7K 3P 1D 1M
Move Barrier	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Barrier	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Barrier	2	0	1	0	2	1	0	2K 1P 2D 1M
Rotate Barrier	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Barrier	8	0	1	0	4	1	0	8K 1P 4D 1M
Select Sim Tab	7	1	2	0	1	1	0	7K 1VS 2P 1D 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	0	0	0	0	0	1	0	1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	622	228	352	11	130	106	2	622K 228VS 352P 11H 130D 106M 2R

Figure B.111 (cont.): Actions performed by participant 4 to complete Scenario 3 in MassMotion and the KLM operators required by each action.

Participant 4 Scenario 3 Sketching

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	2	100
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	3	33.33333333
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	3	33.33333333
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.112: Scenario 3 tasks, number of attempts made by participant 4 and efficiency percentage using the sketching system.

Scenario 3	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	2	1	1	0	0	1	0	2K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	14	1	2	1	0	1	0	14K 1VS 2P 1H 1M
Set # pedestrians	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Set rate	4	2	2	1	0	1	0	4K 2VS 2P 1H 1M
Rename Entrance	15	1	2	1	0	1	0	15K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	2	0	1	0	0	1	0	2K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	23	2	3	0	0	1	0	23K 2VS 3P 1M
Select Area	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Storyboard	14	4	9	0	0	1	0	14K 4VS 9P 1M
Create Storyboard	6	3	6	0	0	1	0	6K 3VS 6P 1M
Delete Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Storyboard	8	1	5	0	0	1	0	8K 1VS 5P 1M
Select Select	2	2	2	0	0	1	0	2K 2VS 2P 1M
Set behaviour	4	2	4	0	0	1	0	4K 2VS 4P 1M
Set behaviour	4	2	4	0	0	1	0	4K 2VS 4P 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	4	1	1	0	0	1	0	4K 1VS 1P 1M
Update Storyboard	9	4	4	1	0	1	0	9K 4VS 4P 1H 1M
Update Storyboard	10	4	4	1	0	1	0	10K 4VS 4P 1H 1M
Set behaviour	6	5	6	0	0	1	0	6K 5VS 6P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Delete Storyboard	4	4	4	0	0	1	0	4K 4VS 4P 1M
Create Storyboard	13	2	6	1	0	1	0	13K 2VS 6P 1H 1M
Delete Storyboard	4	4	4	0	0	1	0	4K 4VS 4P 1M
Create Barrier	11	2	2	0	1	1	0	11K 2VS 2P 1D 1M
Start Simulation	4	1	1	0	0	1	0	4K 1VS 1P 1M
Watch Simulation	36	4	5	1	0	1	0	36K 4VS 5P 1H 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	225	66	94	10	3	39	0	225K 66VS 94P 10H 3D 39M

Figure B.113: Actions performed by participant 4 to complete Scenario 3 in the sketching system and the KLM operators required by each action.

Participant 5 Scenario 1 MassMotion

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:06	04:32	04:32	05:19	05:19	06:10	05:21	06:07
	06:10	07:14	07:14	08:02	08:02	08:44	08:06	08:37
	Total	05:30	Total	01:35	Total	01:33	Total	01:17
2	00:06	08:41	08:41	09:59	09:59	11:39	10:01	11:38
	Total	08:35	Total	01:18	Total	01:40	Total	01:37
3	00:05	12:01	12:01	13:20	13:20	14:00	13:23	13:59
	14:00	16:30	16:30	18:01	18:01	18:36	18:02	18:35
	Total	14:26	Total	02:50	Total	01:15	Total	01:09

Figure B.114: Time (mm:ss) needed by participant 5 to complete each scenario in MassMotion. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	1	100
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.115: Scenario 1 tasks, number of attempts made by participant 5 and efficiency percentage using the MassMotion system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	10	0	0	0	6	1	0	10K 6D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Portal	14	3	3	1	0	1	0	14K 3VS 3P 1H 1M
Move Portal	5	0	2	0	2	1	0	5K 2P 2D 1M
Scale Portal	1	0	5	0	5	1	0	1K 5P 5D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	3	0	2	0	4	1	0	3K 2P 4D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Create Portal	24	3	3	0	4	1	0	24K 3VS 3P 4D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Rotate Portal	3	0	1	0	1	1	0	3K 1P 1D 1M
Move Portal	3	0	5	0	5	1	0	3K 5P 5D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Select Activities Tab	3	1	1	0	4	1	0	3K 1VS 1P 4D 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Portal	16	3	3	1	2	1	0	16K 3VS 3P 1H 2D 1M
Set # Agents	6	4	4	1	0	1	0	6K 4VS 4P 1H 1M
Set Rate	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Origin	5	4	4	0	0	1	0	5K 4VS 4P 1M
Set Destination	6	5	5	0	0	1	0	6K 5VS 5P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	11	2	2	0	2	1	0	11K 2VS 2P 2D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Scene Tab	2	2	2	0	1	1	0	2K 2VS 2P 1D 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Barrier	6	0	1	0	1	1	0	6K 1P 1D 1M
Scale Barrier	1	0	2	0	2	1	0	1K 2P 2D 1M
Move Barrier	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Barrier	2	0	1	0	1	1	0	2K 1P 1D 1M
Select Sim Tab	4	1	1	0	0	1	0	4K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	1	0	0	0	3	1	0	1K 3D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	156	46	73	4	52	39	2	156K 46VS 73P 4H 52D 39M 2R

Figure B.116: Actions performed by participant 5 to complete Scenario 1 in MassMotion and the KLM operators required by each action.

Participant 5 Scenario 1 Sketching

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:11	02:25			02:25	03:14	02:25	03:12
	03:14	04:00			04:00	04:53	04:00	04:50
	Total	03:00	Total	00:00	Total	01:42	Total	01:37
2	00:08	04:23			04:23	06:17	04:23	06:17
	Total	04:15	Total	00:00	Total	01:54	Total	01:54
3	00:05	03:18			03:18	04:56	03:18	04:55
	04:56	05:54			05:54	07:46	05:54	07:45
	Total	04:11	Total	00:00	Total	03:30	Total	03:28

Figure B.117: Time (mm:ss) needed by participant 5 to complete each scenario in the sketching system. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	2	50
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	1	100
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.118: Scenario 1 tasks, number of attempts made by participant 5 and efficiency percentage using the sketching system.

Scenario 1	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Select Entrance/Exit	23	1	1	0	5	1	0	23K 1VS 1P 5D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	12	1	2	1	0	1	0	12K 1VS 2P 1H 1M
Set rate	4	1	1	1	0	1	0	4K 1VS 1P 1H 1M
Set # pedestrians	7	2	2	1	0	1	0	7K 2VS 2P 1H 1M
Select Entrance/Exit	14	1	1	0	9	1	0	14K 1VS 1P 9D 1M
Create Entrance/Exit	2	2	2	0	0	1	0	2K 2VS 2P 1M
Delete Exit	3	2	3	0	0	1	0	3K 2VS 3P 1M
Rename Exit	7	1	2	0	0	1	0	7K 1VS 2P 1M
Set Exit %	10	6	6	1	3	1	0	10K 6VS 6P 1H 3D 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch simulation	4	1	1	0	0	1	0	4K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Barrier	4	1	1	0	0	1	0	4K 1VS 1P 1M
Create Barrier	2	0	0	0	1	1	0	2K 1D 1M
Delete Barrier	3	2	3	0	0	1	0	3K 2VS 3P 1M
Create Barrier	1	1	1	0	1	1	0	1K 1VS 1P 1D 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch simulation	9	0	0	0	0	1	0	9K 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	111	27	32	4	19	21	0	111K 27VS 32P 4H 19D 21M

Figure B.119: Actions performed by participant 5 to complete Scenario 1 in the sketching system and the KLM operators required by each action.

Participant 5 Scenario 2 MassMotion

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.120: Scenario 2 tasks, number of attempts made by participant 5 and efficiency percentage using the MassMotion system.

Scenario 2	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	9	0	0	0	5	1	0	9K 5D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	4	0	2	0	2	1	0	4K 2P 2D 1M
Scale Portal	1	0	3	0	3	1	0	1K 3P 3D 1M
Rotate Portal	3	0	1	0	1	1	0	3K 1P 1D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	1	0	3	0	3	1	0	1K 3P 3D 1M
Rename Portal	17	3	4	1	1	1	0	17K 3VS 4P 1H 1D 1M
Create Portal	1	1	1	0	5	1	0	1K 1VS 1P 5D 1M
Move Portal	5	0	3	0	4	1	0	5K 3P 4D 1M
Scale Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rename Portal	25	3	3	1	1	1	0	25K 3VS 3P 1H 1D 1M
Create Portal	6	1	1	0	3	1	0	6K 1VS 1P 3D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Scale Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	4	0	3	0	3	1	0	4K 3P 3D 1M
Rename Portal	19	3	3	1	1	1	0	19K 3VS 3P 1H 1D 1M
Create Portal	6	1	1	0	6	1	0	6K 1VS 1P 6D 1M
Scale Portal	4	0	2	0	2	1	0	4K 2P 2D 1M
Move Portal	1	0	3	0	3	1	0	1K 3P 3D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rename Portal	13	3	3	1	1	1	0	13K 3VS 3P 1H 1D 1M
Create Portal	18	1	1	0	4	1	0	18K 1VS 1P 4D 1M
Move Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Scale Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	2	0	4	0	4	1	0	2K 4P 4D 1M
Rename Portal	10	3	3	1	1	1	0	10K 3VS 3P 1H 1D 1M
Create Portal	1	1	1	0	1	1	0	1K 1VS 1P 1D 1M
Move Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Scale Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	3	0	9	0	9	1	0	3K 9P 9D 1M
Rename Portal	10	3	4	1	2	1	0	10K 3VS 4P 1H 2D 1M
Select Activities Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Rate	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Origin	8	5	5	0	0	1	0	8K 5VS 5P 1M
Set Destination	8	5	5	0	0	1	0	8K 5VS 5P 1M
Set # Agents	7	8	8	0	4	1	0	7K 8VS 8P 4D 1M
Select Sim Tab	8	1	1	0	2	1	0	8K 1VS 1P 2D 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	3	2	2	0	0	1	0	3K 2VS 2P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	239	56	122	8	104	60	1	239K 56VS 122P 8H 104D 60M 1R

Figure B.121: Actions performed by participant 5 to complete Scenario 2 in MassMotion and the KLM operators required by each action.

Participant 5 Scenario 2 Sketching

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.122: Scenario 2 tasks, number of attempts made by participant 5 and efficiency percentage using the sketching system.

Scenario 2	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	7	1	1	0	0	1	0	7K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	13	1	2	1	0	1	0	13K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Select Entrance/Exit	11	1	1	0	8	1	0	11K 1VS 1P 8D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	14	1	2	1	0	1	0	14K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Select Entrance/Exit	3	1	1	0	1	1	0	3K 1VS 1P 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	17	1	2	1	0	1	0	17K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Select Entrance/Exit	4	1	1	0	1	1	0	4K 1VS 1P 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	8	2	3	1	0	1	0	8K 2VS 3P 1H 1M
Select Entrance/Exit	3	1	1	0	1	1	0	3K 1VS 1P 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	9	2	3	1	0	1	0	9K 2VS 3P 1H 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	9	2	3	1	0	1	0	9K 2VS 3P 1H 1M
Set Exit %	19	5	6	5	2	1	0	19K 5VS 6P 5H 2D 1M
Set Exit %	20	4	5	3	1	1	0	20K 4VS 5P 3H 1D 1M
Set Exit %	16	4	5	1	0	1	0	16K 4VS 5P 1H 1M
Start Simulation	6	1	1	0	1	1	0	6K 1VS 1P 1D 1M
Watch Simulation	12	1	1	0	7	1	0	12K 1VS 1P 7D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	212	46	61	21	22	36	0	212K 46VS 61P 21H 22D 36M

Figure B.123: Actions performed by participant 5 to complete Scenario 2 in the sketching system and the KLM operators required by each action.

Participant 5 Scenario 3 MassMotion

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	2	100
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	3	66.66666667
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	1	100
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.124: Scenario 3 tasks, number of attempts made by participant 5 and efficiency percentage using the MassMotion system.

Scenario 3	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	9	0	0	0	1	1	0	9K 1D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Scale Portal	1	0	4	0	4	1	0	1K 4P 4D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	7	0	3	0	3	1	0	7K 3P 3D 1M
Rename Portal	15	3	3	1	1	1	0	15K 3VS 3P 1H 1D 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Scale Portal	3	0	2	0	3	1	0	3K 2P 3D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	1	0	4	0	4	1	0	1K 4P 4D 1M
Rename Portal	27	4	4	1	1	1	0	27K 4VS 4P 1H 1D 1M
Create Floor	1	1	1	0	1	1	0	1K 1VS 1P 1D 1M
Move Floor	2	0	7	0	7	1	0	2K 7P 7D 1M
Rotate Floor	4	0	1	0	1	1	0	4K 1P 1D 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Link	1	0	2	0	2	1	0	1K 2P 2D 1M
Move Link	1	0	2	0	2	1	0	1K 2P 2D 1M
Rotate Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Link	1	0	2	0	2	1	0	1K 2P 2D 1M
Rename Floor	12	3	3	0	1	1	0	12K 3VS 3P 1D 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Link	1	0	2	0	2	1	0	1K 2P 2D 1M
Move Link	1	0	2	0	2	1	0	1K 2P 2D 1M
Rotate Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Floor	3	2	3	0	1	1	0	3K 2VS 3P 1D 1M
Move Link	5	1	4	0	3	1	0	5K 1VS 4P 3D 1M
Create Floor	8	1	1	0	4	1	0	8K 1VS 1P 4D 1M
Move Floor	1	0	4	0	4	1	0	1K 4P 4D 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Link	1	0	3	0	3	1	0	1K 3P 3D 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Link	1	0	3	0	3	1	0	1K 3P 3D 1M
Rotate Link	5	0	2	0	2	1	0	5K 2P 2D 1M
Move Link	5	0	11	0	11	1	0	5K 11P 11D 1M
Rename Floor	13	3	3	0	1	1	0	13K 3VS 3P 1D 1M
Create Portal	1	1	1	0	3	1	0	1K 1VS 1P 3D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M

Move Portal	1	0	6	0	6	1	0	1K 6P 6D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rename Portal	11	3	3	1	3	1	0	11K 3VS 3P 1H 3D 1M
Select Activities Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	5	3	3	1	1	1	0	5K 3VS 3P 1H 1D 1M
Set Rate	4	1	1	0	1	1	0	4K 1VS 1P 1D 1M
Set Origin	7	5	5	0	0	1	0	7K 5VS 5P 1M
Create Action	1	1	1	0	0	1	0	1K 1VS 1P 1M
Define Actions	3	3	3	0	0	1	0	3K 3VS 3P 1M
Delete Journey	3	2	2	0	0	1	0	3K 2VS 2P 1M
Create Action	1	1	1	0	0	1	0	1K 1VS 1P 1M
Define Actions	31	27	27	0	0	1	0	31K 27VS 27P 1M
Delete Action	13	7	7	0	0	1	0	13K 7VS 7P 1M
Create Action	35	30	30	0	0	1	0	35K 30VS 30P 1M
Modify Actions	10	7	7	0	0	1	0	10K 7VS 7P 1M
Create Journey	2	1	1	0	0	1	0	2K 1VS 1P 1M
Set # Agents	5	2	3	1	1	1	0	5K 2VS 3P 1H 1D 1M
Set Rate	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Origin	5	4	4	0	0	1	0	5K 4VS 4P 1M
Set Destination	4	3	3	0	0	1	0	4K 3VS 3P 1M
Set Action	8	8	8	0	0	1	0	8K 8VS 8P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Rate	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Origin	5	4	4	0	0	1	0	5K 4VS 4P 1M
Set Destination	5	4	4	0	0	1	0	5K 4VS 4P 1M
Set Action	8	6	6	0	0	1	0	8K 6VS 6P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	4	3	3	0	3	1	0	4K 3VS 3P 3D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Modify Actions	38	30	32	0	2	1	0	38K 30VS 32P 2D 1M
Remove Action	6	5	5	0	0	1	0	6K 5VS 5P 1M
Select Scene Tab	5	4	4	0	0	1	0	5K 4VS 4P 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Barrier	1	0	3	0	3	1	0	1K 3P 3D 1M
Move Barrier	1	0	2	0	2	1	0	1K 2P 2D 1M
Rotate Barrier	3	0	1	0	1	1	0	3K 1P 1D 1M
Move Barrier	2	0	4	0	4	1	0	2K 4P 4D 1M
Select Sim Tab	4	1	1	0	0	1	0	4K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	2	2	2	0	1	1	0	2K 2VS 2P 1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	411	213	307	8	120	92	2	411K 213VS 307P 8H 120D 92M 2R

Figure B.125 (cont.): Actions performed by participant 5 to complete Scenario 3 in MassMotion and the KLM operators required by each action.

Participant 5 Scenario 3 Sketching

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	2	100
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	1	100
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.126: Scenario 3 tasks, number of attempts made by participant 5 and efficiency percentage using the sketching system.

Scenario 3	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	3	1	1	0	1	1	0	3K 1VS 1P 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	15	1	2	1	0	1	0	15K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Rename Entrance	14	1	2	1	0	1	0	14K 1VS 2P 1H 1M
Set rate	2	1	1	1	0	1	0	2K 1VS 1P 1H 1M
Set # pedestrians	5	2	2	2	0	1	0	5K 2VS 2P 2H 1M
Select Area	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Select Entrance/Exit	3	2	2	0	0	1	0	3K 2VS 2P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	9	2	3	1	0	1	0	9K 2VS 3P 1H 1M
Rename Area	8	1	2	1	0	1	0	8K 1VS 2P 1H 1M
Set behaviour	3	3	3	0	0	1	0	3K 3VS 3P 1M
Rename Area	8	1	2	1	0	1	0	8K 1VS 2P 1H 1M
Set behaviour	5	5	5	0	0	1	0	5K 5VS 5P 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Storyboard	6	3	6	0	0	1	0	6K 3VS 6P 1M
Create Storyboard	16	4	7	1	1	1	0	16K 4VS 7P 1H 1D 1M
Update Storyboard	8	3	3	2	0	1	0	8K 3VS 3P 2H 1M
Start Simulation	4	4	4	0	0	1	0	4K 4VS 4P 1M
Watch Simulation	7	1	1	0	0	1	0	7K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Delete Storyboard	3	3	3	0	0	1	0	3K 3VS 3P 1M
Delete Storyboard	3	3	3	0	0	1	0	3K 3VS 3P 1M
Set Exit %	8	3	4	2	0	1	0	8K 3VS 4P 2H 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Storyboard	13	4	8	1	0	1	0	13K 4VS 8P 1H 1M
Select Barrier	2	2	2	0	0	1	0	2K 2VS 2P 1M
Create Barrier	0	0	0	0	1	1	0	1D 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	0	0	0	0	1	1	0	1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	166	62	81	16	6	39	0	166K 62VS 81P 16H 6D 39M

Figure B.127: Actions performed by participant 5 to complete Scenario 3 in the sketching system and the KLM operators required by each action.

Participant 6 Scenario 1 MassMotion

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:08	04:42	04:42	05:16				
	05:16	05:43	05:43	06:28	06:28	07:28	06:30	07:28
	07:28	08:32	08:32	09:21	09:21	10:25	09:22	10:25
	Total	06:05	Total	02:08	Total	02:04	Total	02:01
2	00:04	08:51	08:51	10:10	10:10	12:10	10:11	12:09
	Total	08:47	Total	01:19	Total	02:00	Total	01:58
3	00:05	12:26	12:26	13:46	13:46	16:16	13:47	16:14
	16:16	18:54	18:54	20:24	20:24	22:18	20:25	22:14
	Total	14:59	Total	02:50	Total	04:24	Total	04:16

Figure B.128: Time (mm:ss) needed by participant 6 to complete each scenario in MassMotion. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	2	50
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.129: Scenario 1 tasks, number of attempts made by participant 6 and efficiency percentage using the MassMotion system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	32	0	0	0	6	1	0	32K 6D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Scale Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Rotate Portal	4	0	1	0	1	1	0	4K 1P 1D 1M
Move Portal	5	0	2	0	3	1	0	5K 2P 3D 1M
Rename Portal	64	3	4	2	4	1	0	64K 3VS 4P 2H 4D 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Scale Portal	1	0	4	0	4	1	0	1K 4P 4D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	4	0	1	0	2	1	0	4K 1P 2D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rename Portal	12	3	4	1	0	1	0	12K 3VS 4P 1H 1M
Select Activities Tab	4	2	2	0	0	1	0	4K 2VS 2P 1M
Create Journey	12	9	10	0	0	1	0	12K 9VS 10P 1M
Set # Agents	8	4	4	2	1	1	0	8K 4VS 4P 2H 1D 1M
Set Rate	5	1	3	1	1	1	0	5K 1VS 3P 1H 1D 1M
Select Sim Tab	32	5	5	0	6	1	0	32K 5VS 5P 6D 1M
Compile Simulation	5	5	5	0	0	1	1	5K 5VS 5P 1M 1R
Set Origin	7	6	6	0	0	1	0	7K 6VS 6P 1M
Set Destination	6	5	5	0	0	1	0	6K 5VS 5P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	20	1	1	0	4	1	0	20K 1VS 1P 4D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Scene Tab	4	1	1	0	1	1	0	4K 1VS 1P 1D 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Barrier	3	0	1	0	1	1	0	3K 1P 1D 1M
Scale Barrier	2	0	2	0	2	1	0	2K 2P 2D 1M
Rotate Barrier	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Barrier	21	0	2	0	6	1	0	21K 2P 6D 1M
Select Sim Tab	8	2	2	0	0	1	0	8K 2VS 2P 1M
Compile Simulation	5	5	5	0	0	1	1	5K 5VS 5P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	27	0	0	0	3	1	0	27K 3D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	323	65	98	6	60	46	3	323K 65VS 98P 6H 60D 46M 3R

Figure B.130: Actions performed by participant 6 to complete Scenario 1 in MassMotion and the KLM operators required by each action.

Participant 6 Scenario 1 Sketching

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:09	02:19			02:19	03:12	02:19	03:10
	03:12	04:04			04:04	04:55	04:04	04:53
	Total	03:02	Total	00:00	Total	01:44	Total	01:40
2	00:08	04:40			04:40	06:35	04:40	06:34
	Total	04:32	Total	00:00	Total	01:55	Total	01:54
3	00:05	03:39			03:39	05:13	03:39	05:13
	05:13	07:05			07:05	09:00	07:05	08:59
	Total	05:26	Total	00:00	Total	03:29	Total	03:28

Figure B.131: Time (mm:ss) needed by participant 6 to complete each scenario in the sketching system. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	4	25
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	1	100
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.132: Scenario 1 tasks, number of attempts made by participant 6 and efficiency percentage using the sketching system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	13	1	1	0	3	1	0	13K 1VS 1P 3D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Exit	9	1	1	0	6	1	0	9K 1VS 1P 6D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Create Entrance/Exit	3	1	2	0	3	1	0	3K 1VS 2P 3D 1M
Delete Exit	7	4	7	0	0	1	0	7K 4VS 7P 1M
Rename Entrance	17	1	3	2	2	1	0	17K 1VS 3P 2H 2D 1M
Rename Exit	21	1	2	1	0	1	0	21K 1VS 2P 1H 1M
Set rate	5	1	2	1	0	1	0	5K 1VS 2P 1H 1M
Set # pedestrians	7	2	2	1	0	1	0	7K 2VS 2P 1H 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch simulation	4	2	2	0	1	1	0	4K 2VS 2P 1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	10	0	0	0	3	1	0	10K 3D 1M
Start Simulation	5	2	2	0	0	1	0	5K 2VS 2P 1M
Watch simulation	1	0	0	0	0	1	0	1K 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	108	20	30	5	18	18	0	108K 20VS 30P 5H 18D 18M

Figure B.133: Actions performed by participant 6 to complete Scenario 1 in the sketching system and the KLM operators required by each action.

Participant 6 Scenario 2 MassMotion

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.134: Scenario 2 tasks, number of attempts made by participant 6 and efficiency percentage using the MassMotion system.

Scenario 2	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Move Camera	29	0	0	0	6	1	0	29K 6D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Portal	7	0	1	0	1	1	0	7K 1P 1D 1M
Move Portal	2	0	1	0	2	1	0	2K 1P 2D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rename Portal	22	3	2	1	1	1	0	22K 3VS 2P 1H 1D 1M
Create Portal	16	1	1	0	2	1	0	16K 1VS 1P 2D 1M
Scale Portal	4	0	1	0	1	1	0	4K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	33	0	2	0	8	1	0	33K 2P 8D 1M
Rename Portal	30	5	7	1	5	1	0	30K 5VS 7P 1H 5D 1M
Move Portal	5	0	2	0	1	1	0	5K 2P 1D 1M
Create Portal	15	1	1	0	3	1	0	15K 1VS 1P 3D 1M
Scale Portal	1	0	3	0	3	1	0	1K 3P 3D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rename Portal	11	3	4	1	1	1	0	11K 3VS 4P 1H 1D 1M
Create Portal	18	1	1	0	4	1	0	18K 1VS 1P 4D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Scale Portal	2	0	2	0	2	1	0	2K 2P 2D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	8	0	2	0	3	1	0	8K 2P 3D 1M
Scale Portal	3	0	1	0	1	1	0	3K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	3	0	2	0	2	1	0	3K 2P 2D 1M
Rename Portal	18	4	6	1	1	1	0	18K 4VS 6P 1H 1D 1M
Create Portal	43	1	1	0	7	1	0	43K 1VS 1P 7D 1M
Scale Portal	6	0	2	0	2	1	0	6K 2P 2D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Rename Portal	15	3	3	1	1	1	0	15K 3VS 3P 1H 1D 1M
Create Portal	30	1	1	0	11	1	0	30K 1VS 1P 11D 1M
Scale Portal	6	0	2	0	2	1	0	6K 2P 2D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	18	0	3	0	7	1	0	18K 3P 7D 1M
Rename Portal	18	4	5	1	1	1	0	18K 4VS 5P 1H 1D 1M
Select Activities Tab	4	1	1	0	0	1	0	4K 1VS 1P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	7	4	4	1	1	1	0	7K 4VS 4P 1H 1D 1M
Set Rate	4	2	2	1	1	1	0	4K 2VS 2P 1H 1D 1M
Set Origin	11	9	11	0	0	1	0	11K 9VS 11P 1M
Set Destination	14	22	23	0	0	1	0	14K 22VS 23P 1M
Select Sim Tab	7	1	1	0	0	1	0	7K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	20	1	1	0	4	1	0	20K 1VS 1P 4D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	459	76	127	8	103	56	1	459K 76VS 127P 8H 103D 56M 1R

Figure B.135: Actions performed by participant 6 to complete Scenario 2 in MassMotion and the KLM operators required by each action.

Participant 6 Scenario 2 Sketching

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.136: Scenario 2 tasks, number of attempts made by participant 6 and efficiency percentage using the sketching system.

Scenario 2	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	11	1	1	0	1	1	0	11K 1VS 1P 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Create Entrance/Exit	7	0	1	0	0	1	0	7K 1P 1M
Create Entrance/Exit	6	0	1	0	2	1	0	6K 1P 2D 1M
Select Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	7	0	1	0	1	1	0	7K 1P 1D 1M
Create Entrance/Exit	34	0	1	0	10	1	0	34K 1P 10D 1M
Create Entrance/Exit	2	0	1	0	0	1	0	2K 1P 1M
Select Select	16	1	1	0	0	1	0	16K 1VS 1P 1M
Rename Entrance	14	2	3	2	0	1	0	14K 2VS 3P 2H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	5	1	1	2	0	1	0	5K 1VS 1P 2H 1M
Set Exit %	13	4	4	3	0	1	0	13K 4VS 4P 3H 1M
Rename Entrance	18	1	2	1	0	1	0	18K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Set Exit %	15	4	4	3	0	1	0	15K 4VS 4P 3H 1M
Rename Entrance	16	2	3	1	0	1	0	16K 2VS 3P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Set Exit %	13	4	4	3	0	1	0	13K 4VS 4P 3H 1M
Rename Exit	11	2	3	1	0	1	0	11K 2VS 3P 1H 1M
Rename Exit	22	2	3	1	2	1	0	22K 2VS 3P 1H 2D 1M
Rename Exit	15	2	3	1	2	1	0	15K 2VS 3P 1H 2D 1M
Start Simulation	10	4	7	0	0	1	0	10K 4VS 7P 1M
Watch Simulation	34	1	1	0	9	1	0	34K 1VS 1P 9D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	291	38	53	23	27	27	0	291K 38VS 53P 23H 27D 27M

Figure B.137: Actions performed by participant 6 to complete Scenario 2 in the sketching system and the KLM operators required by each action.

Participant 6 Scenario 3 MassMotion

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	3	66.66666667
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	3	33.33333333
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.138: Scenario 3 tasks, number of attempts made by participant 6 and efficiency percentage using the MassMotion system.

Scenario 3	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Move Camera	26	0	0	0	7	1	0	26K 7D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Portal	8	0	2	0	4	1	0	8K 2P 4D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	20	0	2	0	5	1	0	20K 2P 5D 1M
Rename Portal	19	3	4	1	2	1	0	19K 3VS 4P 1H 2D 1M
Create Portal	5	1	1	0	1	1	0	5K 1VS 1P 1D 1M
Scale Portal	3	0	2	0	2	1	0	3K 2P 2D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	6	0	2	0	4	1	0	6K 2P 4D 1M
Rename Portal	16	2	4	1	1	1	0	16K 2VS 4P 1H 1D 1M
Create Portal	7	1	1	0	2	1	0	7K 1VS 1P 2D 1M
Scale Portal	2	0	3	0	3	1	0	2K 3P 3D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	7	0	2	0	3	1	0	7K 2P 3D 1M
Rename Portal	9	3	3	1	1	1	0	9K 3VS 3P 1H 1D 1M
Create Floor	8	1	1	0	1	1	0	8K 1VS 1P 1D 1M
Move Floor	2	0	5	0	4	1	0	2K 5P 4D 1M
Rotate Floor	5	0	2	0	3	1	0	5K 2P 3D 1M
Move Floor	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Floor	20	6	13	0	5	1	0	20K 6VS 13P 5D 1M
Scale Floor	1	0	2	0	2	1	0	1K 2P 2D 1M
Move Floor	1	0	2	0	2	1	0	1K 2P 2D 1M
Delete Floor	5	4	5	0	0	1	0	5K 4VS 5P 1M
Create Floor	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Floor	4	0	2	0	3	1	0	4K 2P 3D 1M
Rotate Floor	2	0	2	0	2	1	0	2K 2P 2D 1M
Rename Floor	12	6	7	2	2	1	0	12K 6VS 7P 2H 2D 1M
Rename Floor	15	5	6	1	1	1	0	15K 5VS 6P 1H 1D 1M
Select Activities Tab	6	1	2	0	1	1	0	6K 1VS 2P 1D 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Scene Tab	11	11	11	0	0	1	0	11K 11VS 11P 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Link	8	0	5	0	5	1	0	8K 5P 5D 1M
Rotate Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Link	6	0	1	0	2	1	0	6K 1P 2D 1M
Scale Link	2	0	2	0	2	1	0	2K 2P 2D 1M
Rotate Link	4	0	1	0	2	1	0	4K 1P 2D 1M
Move Link	1	0	2	0	2	1	0	1K 2P 2D 1M
Create Link	2	1	2	0	0	1	0	2K 1VS 2P 1M
Scale Link	3	0	2	0	2	1	0	3K 2P 2D 1M
Rotate Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Link	1	0	2	0	2	1	0	1K 2P 2D 1M
Create Link	1	0	0	0	0	1	0	1K 1M

Select Activities Tab	4	1	1	0	0	1	0	4K 1VS 1P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Delete Journey	4	4	4	0	0	1	0	4K 4VS 4P 1M
Define Actions	36	24	27	0	2	1	0	36K 24VS 27P 2D 1M
Set Origin	5	4	4	0	0	1	0	5K 4VS 4P 1M
Set Destination	4	3	3	0	0	1	0	4K 3VS 3P 1M
Set # Agents	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Rate	5	2	3	1	1	1	0	5K 2VS 3P 1H 1D 1M
Create Link	16	2	2	0	3	1	0	16K 2VS 2P 3D 1M
Scale Link	15	0	2	0	3	1	0	15K 2P 3D 1M
Rotate Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Link	1	0	2	0	2	1	0	1K 2P 2D 1M
Scale Link	2	0	2	0	2	1	0	2K 2P 2D 1M
Move Link	13	0	4	0	5	1	0	13K 4P 5D 1M
Rotate Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Link	5	0	1	0	2	1	0	5K 1P 2D 1M
Move Floor	3	0	2	0	1	1	0	3K 2P 1D 1M
Move Link	2	0	1	0	1	1	0	2K 1P 1D 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Link	3	0	2	0	2	1	0	3K 2P 2D 1M
Rotate Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Link	1	0	2	0	2	1	0	1K 2P 2D 1M
Set Origin	1	0	0	0	0	1	0	1K 1M
Select Activities Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Rate	4	1	2	1	1	1	0	4K 1VS 2P 1H 1D 1M
Set Origin	8	6	6	0	0	1	0	8K 6VS 6P 1M
Set Destination	5	4	4	0	0	1	0	5K 4VS 4P 1M
Define Actions	28	23	25	0	2	1	0	28K 23VS 25P 2D 1M
Select Sim Tab	11	1	1	0	1	1	0	11K 1VS 1P 1D 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	4	2	2	0	0	1	0	4K 2VS 2P 1M
Modify Actions	10	8	9	0	1	1	0	10K 8VS 9P 1D 1M
Modify Actions	9	5	9	0	1	1	0	9K 5VS 9P 1D 1M
Stop Simulation	2	2	2	0	0	1	0	2K 2VS 2P 1M
Select Scene Tab	9	1	1	0	2	1	0	9K 1VS 1P 2D 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Barrier	1	0	2	0	2	1	0	1K 2P 2D 1M
Rotate Barrier	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Barrier	1	0	2	0	2	1	0	1K 2P 2D 1M
Select Activities Tab	12	3	3	0	4	1	0	12K 3VS 3P 4D 1M
Modify Actions	38	25	25	0	2	1	0	38K 25VS 25P 2D 1M
Delete Action	10	8	8	0	0	1	0	10K 8VS 8P 1M
Select Sim Tab	8	5	5	0	0	1	0	8K 5VS 5P 1M
Compile Simulation	5	5	5	0	0	1	1	5K 5VS 5P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	0	0	0	0	0	1	0	1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	574	203	309	10	138	99	2	574K 203VS 309P 10H 138D 99M 2R

Figure B.139 (cont.): Actions performed by participant 6 to complete Scenario 3 in MassMotion and the KLM operators required by each action.

Participant 6 Scenario 3 Sketching

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	2	100
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	2	50
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	3	33.33333333
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.140: Scenario 3 tasks, number of attempts made by participant 6 and efficiency percentage using the sketching system.

Scenario 3	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	5	1	1	0	1	1	0	5K 1VS 1P 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	14	1	2	1	0	1	0	14K 1VS 2P 1H 1M
Rename Entrance	18	2	3	1	0	1	0	18K 2VS 3P 1H 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	9	2	3	1	0	1	0	9K 2VS 3P 1H 1M
Select Area	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Area	9	1	2	1	0	1	0	9K 1VS 2P 1H 1M
Set behaviour	3	3	3	0	0	1	0	3K 3VS 3P 1M
Rename Area	8	1	2	1	0	1	0	8K 1VS 2P 1H 1M
Set behaviour	3	3	3	0	0	1	0	3K 3VS 3P 1M
Set rate	9	2	3	1	0	1	0	9K 2VS 3P 1H 1M
Set # pedestrians	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Set rate	4	1	2	1	0	1	0	4K 1VS 2P 1H 1M
Set # pedestrians	6	2	2	1	0	1	0	6K 2VS 2P 1H 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Storyboard	6	3	6	0	0	1	0	6K 3VS 6P 1M
Create Storyboard	6	3	6	0	0	1	0	6K 3VS 6P 1M
Delete Storyboard	9	7	9	0	0	1	0	9K 7VS 9P 1M
Create Storyboard	4	1	4	0	0	1	0	4K 1VS 4P 1M
Start Simulation	4	3	4	0	0	1	0	4K 3VS 4P 1M
Watch Simulation	13	1	1	0	0	1	0	13K 1VS 1P 1M
Update Storyboard	9	4	4	1	0	1	0	9K 4VS 4P 1H 1M
Update Storyboard	8	3	3	1	0	1	0	8K 3VS 3P 1H 1M
Stop Simulation	5	3	3	0	0	1	0	5K 3VS 3P 1M
Select Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	0	0	0	0	1	1	0	1D 1M
Select Storyboard	12	1	1	0	0	1	0	12K 1VS 1P 1M
Delete Storyboard	4	4	4	0	0	1	0	4K 4VS 4P 1M
Create Storyboard	5	1	5	0	0	1	0	5K 1VS 5P 1M
Delete Storyboard	4	4	4	0	0	1	0	4K 4VS 4P 1M
Create Storyboard	9	2	4	1	0	1	0	9K 2VS 4P 1H 1M
Update Storyboard	10	5	5	1	0	1	0	10K 5VS 5P 1H 1M
Check Storyboards	6	6	6	0	0	1	0	6K 6VS 6P 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	6	1	1	0	0	1	0	6K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	225	81	109	13	4	44	0	225K 81VS 109P 13H 4D 44M

Figure B.141: Actions performed by participant 6 to complete Scenario 3 in the sketching system and the KLM operators required by each action.

Participant 7 Scenario 1 MassMotion

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:08	05:49	05:49	06:49	06:49	07:15	06:51	07:13
	07:15	09:03	09:03	09:57	09:57	10:19	09:58	10:14
	Total	07:29	Total	01:54	Total	00:48	Total	00:38
2	00:06	12:10	12:10	12:58				
	12:58	14:26	14:26	15:48	15:48	16:39	15:52	16:36
	Total	13:32	Total	02:10	Total	00:51	Total	00:44
3	00:09	15:42	15:42	17:04	17:04	17:36	17:12	17:35
	17:36	21:48	21:48	23:44	23:44	24:14	23:45	24:12
	Total	19:45	Total	03:18	Total	01:02	Total	00:50

Figure B.142: Time (mm:ss) needed by participant 7 to complete each scenario in MassMotion. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	1	100
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.143: Scenario 1 tasks, number of attempts made by participant 7 and efficiency percentage using the MassMotion system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	11	0	0	0	7	1	0	11K 7D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Portal	7	2	6	0	3	1	0	7K 2VS 6P 3D 1M
Move Portal	5	3	6	0	4	1	0	5K 3VS 6P 4D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	3	2	5	0	3	1	0	3K 2VS 5P 3D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Portal	13	1	2	1	0	1	0	13K 1VS 2P 1H 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Portal	6	1	4	0	3	1	0	6K 1VS 4P 3D 1M
Move Portal	12	4	6	0	3	1	0	12K 4VS 6P 3D 1M
Scale Portal	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Portal	8	1	1	2	0	1	0	8K 1VS 1P 2H 1M
Select Activities Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	2	1	1	0	0	1	0	2K 1VS 1P 1M
Set Rate	3	1	1	0	1	1	0	3K 1VS 1P 1D 1M
Set Origin	4	3	3	0	0	1	0	4K 3VS 3P 1M
Set Destination	5	4	4	0	0	1	0	5K 4VS 4P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	2	2	2	0	0	1	0	2K 2VS 2P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Barrier	10	1	3	0	2	1	0	10K 1VS 3P 2D 1M
Move Barrier	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Barrier	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Barrier	2	2	3	0	1	1	0	2K 2VS 3P 1D 1M
Select Sim Tab	3	3	3	0	0	1	0	3K 3VS 3P 1M
Compile Simulation	7	7	7	0	0	1	1	7K 7VS 7P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	2	0	0	0	0	1	0	2K 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	129	62	92	3	38	39	2	129K 62VS 92P 3H 38D 39M 2R

Figure B.144: Actions performed by participant 7 to complete Scenario 1 in MassMotion and the KLM operators required by each action.

Participant 7 Scenario 1 Sketching

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:11	01:40			01:40	02:27	01:40	02:26
	02:27	02:45			02:45	03:54	02:45	03:35
	Total	01:47	Total	00:00	Total	01:56	Total	01:36
2	00:07	05:01			05:01	06:57	05:01	06:53
	Total	04:54	Total	00:00	Total	01:56	Total	01:52
3	00:07	02:45			02:45	04:28	02:45	04:26
	04:28	06:20			06:20	08:14	06:20	08:12
	Total	04:30	Total	00:00	Total	03:37	Total	03:33

Figure B.145: Time (mm:ss) needed by participant 7 to complete each scenario in the sketching system. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	1	100
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.146: Scenario 1 tasks, number of attempts made by participant 7 and efficiency percentage using the sketching system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	5	1	1	0	0	1	0	5K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	8	0	1	0	0	1	0	8K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	12	1	2	1	0	1	0	12K 1VS 2P 1H 1M
Rename Exit	14	1	2	1	0	1	0	14K 1VS 2P 1H 1M
Set # pedestrians	14	3	4	2	0	1	0	14K 3VS 4P 2H 1M
Set rate	6	3	3	1	0	1	0	6K 3VS 3P 1H 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch simulation	4	2	2	0	1	1	0	4K 2VS 2P 1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	0	0	0	0	1	1	0	1D 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch simulation	46	0	0	0	0	1	0	46K 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	117	18	23	5	2	17	0	117K 18VS 23P 5H 2D 17M

Figure B.147: Actions performed by participant 7 to complete Scenario 1 in the sketching system and the KLM operators required by each action.

Participant 7 Scenario 2 MassMotion

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	6	50
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.148: Scenario 2 tasks, number of attempts made by participant 7 and efficiency percentage using the MassMotion system.

Scenario 2	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	3	0	0	0	2	1	0	3K 2D 1M
Select Scene Tab	2	2	2	0	0	1	0	2K 2VS 2P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Portal	3	1	2	0	1	1	0	3K 1VS 2P 1D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	2	2	4	0	2	1	0	2K 2VS 4P 2D 1M
Move Portal	1	1	2	0	2	1	0	1K 1VS 2P 2D 1M
Rename Portal	14	1	1	2	0	1	0	14K 1VS 1P 2H 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Portal	3	1	3	0	3	1	0	3K 1VS 3P 3D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	2	2	7	0	5	1	0	2K 2VS 7P 5D 1M
Move Portal	2	2	3	0	1	1	0	2K 2VS 3P 1D 1M
Rename Portal	27	1	1	2	0	1	0	27K 1VS 1P 2H 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Move Portal	8	1	5	0	7	1	0	8K 1VS 5P 7D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Scale Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	4	4	6	0	2	1	0	4K 4VS 6P 2D 1M
Rename Portal	14	1	1	2	0	1	0	14K 1VS 1P 2H 1M
Create Portal	10	1	1	0	1	1	0	10K 1VS 1P 1D 1M
Scale Portal	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Rotate Portal	3	1	4	0	3	1	0	3K 1VS 4P 3D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	1	1	5	0	4	1	0	1K 1VS 5P 4D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Portal	9	1	1	1	0	1	0	9K 1VS 1P 1H 1M
Create Portal	1	1	1	0	1	1	0	1K 1VS 1P 1D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Scale Portal	3	3	4	0	1	1	0	3K 3VS 4P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	2	2	3	0	1	1	0	2K 2VS 3P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Portal	9	1	1	1	0	1	0	9K 1VS 1P 1H 1M
Create Portal	6	1	1	0	3	1	0	6K 1VS 1P 3D 1M
Scale Portal	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	1	1	5	0	4	1	0	1K 1VS 5P 4D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Portal	9	1	1	0	0	1	0	9K 1VS 1P 1M
Select Activities Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	3	1	1	1	1	1	0	3K 1VS 1P 1H 1D 1M
Set Rate	7	1	1	1	1	1	0	7K 1VS 1P 1H 1D 1M
Set Origin	15	12	12	0	1	1	0	15K 12VS 12P 1D 1M
Set Destination	10	10	10	0	0	1	0	10K 10VS 10P 1M
Set # Agents	6	3	3	1	1	1	0	6K 3VS 3P 1H 1D 1M
Select Sim Tab	2	2	2	0	0	1	0	2K 2VS 2P 1M
Compile Simulation	5	5	5	0	0	1	1	5K 5VS 5P 1M 1R
Move Portal	4	4	5	0	2	1	0	4K 4VS 5P 2D 1M
Move Portal	9	2	3	0	8	1	0	9K 2VS 3P 8D 1M
Move Portal	12	2	3	0	7	1	0	12K 2VS 3P 7D 1M
Select Sim Tab	7	1	1	0	2	1	0	7K 1VS 1P 2D 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	4	2	2	0	0	1	0	4K 2VS 2P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	256	111	175	11	96	66	2	256K 111VS 175P 11H 96D 66M 2R

Figure B.149: Actions performed by participant 7 to complete Scenario 2 in MassMotion and the KLM operators required by each action.

Participant 7 Scenario 2 Sketching

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.150: Scenario 2 tasks, number of attempts made by participant 7 and efficiency percentage using the sketching system.

Scenario 2	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	3	1	1	0	0	1	0	3K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	15	2	3	1	0	1	0	15K 2VS 3P 1H 1M
Select Entrance/Exit	3	3	3	0	0	1	0	3K 3VS 3P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	10	2	3	1	0	1	0	10K 2VS 3P 1H 1M
Select Entrance/Exit	7	1	1	0	0	1	0	7K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	11	2	3	1	0	1	0	11K 2VS 3P 1H 1M
Select Entrance/Exit	4	1	1	0	0	1	0	4K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	16	2	3	2	0	1	0	16K 2VS 3P 2H 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	9	0	1	0	0	1	0	9K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	15	2	3	1	0	1	0	15K 2VS 3P 1H 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	17	2	3	1	0	1	0	17K 2VS 3P 1H 1M
Set rate	8	1	2	1	0	1	0	8K 1VS 2P 1H 1M
Set # pedestrians	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Set Exit %	15	5	5	1	0	1	0	15K 5VS 5P 1H 1M
Set rate	6	1	2	0	0	1	0	6K 1VS 2P 1M
Set # pedestrians	5	1	1	0	0	1	0	5K 1VS 1P 1M
Set Exit %	14	4	4	1	0	1	0	14K 4VS 4P 1H 1M
Set rate	8	1	2	0	0	1	0	8K 1VS 2P 1M
Set # pedestrians	5	1	1	0	0	1	0	5K 1VS 1P 1M
Set Exit %	13	4	4	0	0	1	0	13K 4VS 4P 1M
Start Simulation	15	1	1	0	0	1	0	15K 1VS 1P 1M
Watch Simulation	20	1	1	0	0	1	0	20K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	238	48	63	11	0	36	0	238K 48VS 63P 11H 36M

Figure B.151: Actions performed by participant 7 to complete Scenario 2 in the sketching system and the KLM operators required by each action.

Participant 7 Scenario 3 MassMotion

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	2	100
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	1	100
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.152: Scenario 3 tasks, number of attempts made by participant 7 and efficiency percentage using the MassMotion system.

Scenario 3	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Move Camera	6	0	0	0	4	1	0	6K 4D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	4	1	2	0	1	1	0	4K 1VS 2P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Portal	18	1	1	1	0	1	0	18K 1VS 1P 1H 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rotate Portal	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Move Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rename Portal	13	1	1	1	0	1	0	13K 1VS 1P 1H 1M
Move Portal	2	2	3	0	1	1	0	2K 2VS 3P 1D 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	1	1	4	0	4	1	0	1K 1VS 4P 4D 1M
Rotate Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Scale Portal	1	1	9	0	8	1	0	1K 1VS 9P 8D 1M
Move Portal	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rename Portal	8	1	1	2	0	1	0	8K 1VS 1P 2H 1M
Create Floor	1	1	1	0	1	1	0	1K 1VS 1P 1D 1M
Move Floor	2	2	5	0	3	1	0	2K 2VS 5P 3D 1M
Rename Floor	9	1	2	1	0	1	0	9K 1VS 2P 1H 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Link	4	1	3	0	2	1	0	4K 1VS 3P 2D 1M
Move Link	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Rename Link	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Link	3	3	7	0	4	1	0	3K 3VS 7P 4D 1M
Scale Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Link	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Create Floor	4	1	1	0	1	1	0	4K 1VS 1P 1D 1M
Move Floor	3	1	2	0	1	1	0	3K 1VS 2P 1D 1M
Scale Floor	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Move Floor	1	1	3	0	2	1	0	1K 1VS 3P 2D 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Link	3	1	3	0	3	1	0	3K 1VS 3P 3D 1M
Scale Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Link	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Create Link	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rotate Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M

Move Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Link	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Move Link	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Rename Link	6	1	1	1	0	1	0	6K 1VS 1P 1H 1M
Select Activities Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	9	2	2	2	0	1	0	9K 2VS 2P 2H 1M
Set Rate	9	2	2	3	1	1	0	9K 2VS 2P 3H 1D 1M
Set Origin	6	5	5	0	0	1	0	6K 5VS 5P 1M
Set Destination	5	4	4	0	0	1	0	5K 4VS 4P 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Set Rate	4	1	1	1	0	1	0	4K 1VS 1P 1H 1M
Set Origin	4	3	3	0	0	1	0	4K 3VS 3P 1M
Set Destination	5	4	4	0	0	1	0	5K 4VS 4P 1M
Create Action	1	1	1	0	0	1	0	1K 1VS 1P 1M
Define Actions	46	41	41	0	3	1	0	46K 41VS 41P 3D 1M
Create Action	14	10	10	0	0	1	0	14K 10VS 10P 1M
Define Actions	30	24	24	1	0	1	0	30K 24VS 24P 1H 1M
Set Action	9	7	7	0	0	1	0	9K 7VS 7P 1M
Set Action	8	6	6	0	0	1	0	8K 6VS 6P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	6	1	1	0	1	1	0	6K 1VS 1P 1D 1M
Watch Simulation	1	1	1	0	1	1	0	1K 1VS 1P 1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Activities Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Modify Actions	29	22	22	0	0	1	0	29K 22VS 22P 1M
Delete Action	4	1	1	2	0	1	0	4K 1VS 1P 2H 1M
Remove Action	7	5	5	1	0	1	0	7K 5VS 5P 1H 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Barrier	6	1	2	0	2	1	0	6K 1VS 2P 2D 1M
Rotate Barrier	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Barrier	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Rotate Barrier	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Barrier	13	6	9	5	3	1	0	13K 6VS 9P 5H 3D 1M
Move Barrier	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Scale Barrier	1	1	4	0	3	1	0	1K 1VS 4P 3D 1M
Move Barrier	1	1	2	0	1	1	0	1K 1VS 2P 1D 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	5	5	5	0	0	1	1	5K 5VS 5P 1M 1R
Run Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	2	0	0	0	1	1	0	2K 1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	378	225	296	25	86	89	2	378K 225VS 296P 25H 86D 89M 2R

Figure B.153 (cont.): Actions performed by participant 7 to complete Scenario 3 in MassMotion and the KLM operators required by each action.

Participant 7 Scenario 3 Sketching

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	2	100
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	1	100
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.154: Scenario 3 tasks, number of attempts made by participant 7 and efficiency percentage using the sketching system.

Scenario 3	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	2	0	1	0	0	1	0	2K 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Entrance/Exit	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Entrance/Exit	2	0	1	0	0	1	0	2K 1P 1M
Select Area	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	18	2	3	1	0	1	0	18K 2VS 3P 1H 1M
Rename Entrance	28	2	2	1	0	1	0	28K 2VS 2P 1H 1M
Rename Area	9	1	2	1	0	1	0	9K 1VS 2P 1H 1M
Set behaviour	3	3	3	0	0	1	0	3K 3VS 3P 1M
Rename Area	8	1	2	1	0	1	0	8K 1VS 2P 1H 1M
Set behaviour	3	3	3	0	0	1	0	3K 3VS 3P 1M
Rename Exit	8	2	3	1	0	1	0	8K 2VS 3P 1H 1M
Set rate	4	1	2	0	0	1	0	4K 1VS 2P 1M
Set # pedestrians	11	4	4	0	0	1	0	11K 4VS 4P 1M
Set rate	4	1	2	0	0	1	0	4K 1VS 2P 1M
Set # pedestrians	6	2	2	0	0	1	0	6K 2VS 2P 1M
Select Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Storyboard	19	5	8	1	0	1	0	19K 5VS 8P 1H 1M
Create Storyboard	13	4	7	0	0	1	0	13K 4VS 7P 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	4	1	1	0	0	1	0	4K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Delete Storyboard	3	3	3	0	0	1	0	3K 3VS 3P 1M
Create Storyboard	10	2	5	1	0	1	0	10K 2VS 5P 1H 1M
Delete Storyboard	3	3	3	0	0	1	0	3K 3VS 3P 1M
Create Storyboard	9	2	4	1	0	1	0	9K 2VS 4P 1H 1M
Select Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	0	0	0	0	1	1	0	1D 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	178	52	72	8	3	34	0	178K 52VS 72P 8H 3D 34M

Figure B.155: Actions performed by participant 7 to complete Scenario 3 in the sketching system and the KLM operators required by each action.

Participant 8 Scenario 1 MassMotion

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:05	02:09	02:09	02:29				
	02:29	03:29	03:29	04:16	04:16	04:40	04:21	04:38
	04:40	05:48	05:48	06:37	06:37	06:49	06:39	06:54
	Total	04:12	Total	01:56	Total	00:36	Total	00:32
2	00:04	07:05	07:05	08:22	08:22	08:56	08:25	08:53
	Total	07:01	Total	01:17	Total	00:34	Total	00:28
3	00:05	09:29	09:29	10:18				
	10:18	11:17	11:17	12:36	12:36	13:02	12:39	13:01
	13:02	15:20	15:20	16:49	16:49	17:25	16:57	17:24
	Total	12:41	Total	03:37	Total	01:02	Total	00:49

Figure B.156: Time (mm:ss) needed by participant 8 to complete each scenario in MassMotion. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	2	50
Create and rename the exit	2	50
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	1	100
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.157: Scenario 1 tasks, number of attempts made by participant 8 and efficiency percentage using the MassMotion system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	6	0	0	0	6	1	0	6K 6D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Scale Portal	4	0	2	0	2	1	0	4K 2P 2D 1M
Move Portal	3	0	1	0	1	1	0	3K 1P 1D 1M
Rotate Portal	5	0	1	0	1	1	0	5K 1P 1D 1M
Scale Portal	1	0	1	0	3	1	0	1K 1P 3D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rename Portal	16	3	3	1	0	1	0	16K 3VS 3P 1H 1M
Duplicate Portal	2	2	2	0	0	1	0	2K 2VS 2P 1M
Rename Portal	11	3	3	1	0	1	0	11K 3VS 3P 1H 1M
Move Portal	7	0	5	0	10	1	0	7K 5P 10D 1M
Select Activities Tab	4	1	1	0	3	1	0	4K 1VS 1P 3D 1M
Create Journey	1	1	1	0	0	1	0	1K 1VS 1P 1M
Set # Agents	6	1	1	1	1	1	0	6K 1VS 1P 1H 1D 1M
Set Rate	11	2	2	2	0	1	0	11K 2VS 2P 2H 1M
Set Origin	4	3	3	0	0	1	0	4K 3VS 3P 1M
Set Destination	5	4	4	0	0	1	0	5K 4VS 4P 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	5	5	5	0	0	1	0	5K 5VS 5P 1M
Move Portal	15	1	4	0	12	1	0	15K 1VS 4P 12D 1M
Move Portal	15	0	2	0	9	1	0	15K 2P 9D 1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	5	4	4	0	0	1	1	5K 4VS 4P 1M 1R
Run Simulation	5	1	1	0	3	1	0	5K 1VS 1P 3D 1M
Watch Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Barrier	7	0	1	0	4	1	0	7K 1P 4D 1M
Scale Barrier	1	0	5	0	5	1	0	1K 5P 5D 1M
Rotate Barrier	4	0	1	0	1	1	0	4K 1P 1D 1M
Move Barrier	2	0	1	0	1	1	0	2K 1P 1D 1M
Scale Barrier	2	0	1	0	3	1	0	2K 1P 3D 1M
Select Sim Tab	6	2	2	0	1	1	0	6K 2VS 2P 1D 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	1	1	1	0	3	1	0	1K 1VS 1P 3D 1M
Watch Simulation	3	0	0	0	2	1	0	3K 2D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	172	47	72	5	72	39	2	172K 47VS 72P 5H 72D 39M 2R

Figure B.158: Actions performed by participant 8 to complete Scenario 1 in MassMotion and the KLM operators required by each action.

Participant 8 Scenario 1 Sketching

	Setup		Compilation		Total Run		Run only	
	Start	End	Start	End	Start	End	Start	End
1	00:06	00:55			00:55	01:43	00:55	01:42
	01:43	02:22			02:22	03:13	02:22	03:12
	Total	01:28	Total	00:00	Total	01:39	Total	01:37
2	00:07	03:13			03:13	05:09	03:13	05:07
	Total	03:06	Total	00:00	Total	01:56	Total	01:54
3	00:05	02:45			02:45	04:33	02:45	04:32
	04:33	06:07			06:07	08:16	06:07	08:15
	Total	04:14	Total	00:00	Total	03:57	Total	03:55

Figure B.159: Time (mm:ss) needed by participant 8 to complete each scenario in the sketching system. Time is divided into Setup, Compilation, Total Run (includes idle time after simulation finished) and Run only (simulation time).

Scenario 1 Tasks	Attempts	%
Create and rename the entrance	1	100
Create and rename the exit	1	100
Spawn 1,000 pedestrians at a 10 pedestrians/s rate	1	100
Run the simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Create the barrier	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.160: Scenario 1 tasks, number of attempts made by participant 8 and efficiency percentage using the sketching system.

Scenario 1	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	1	1	1	0	2	1	0	1K 1VS 1P 2D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Entrance/Exit	1	1	1	0	1	1	0	1K 1VS 1P 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	18	1	2	1	0	1	0	18K 1VS 2P 1H 1M
Set rate	4	1	1	1	0	1	0	4K 1VS 1P 1H 1M
Set # pedestrians	7	2	2	0	0	1	0	7K 2VS 2P 1M
Rename Exit	14	4	5	0	0	1	0	14K 4VS 5P 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch simulation	7	1	1	0	3	1	0	7K 1VS 1P 3D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Barrier	8	0	0	0	4	1	0	8K 4D 1M
Create Barrier	0	0	0	0	1	1	0	1D 1M
Delete Barrier	5	2	3	0	2	1	0	5K 2VS 3P 2D 1M
Create Barrier	1	1	1	0	1	1	0	1K 1VS 1P 1D 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch simulation	4	0	0	0	2	1	0	4K 2D 1M
Stop Simulation	1	1	1	0	0	0	0	1K 1VS 1P
	77	19	24	2	16	18	0	77K 19VS 24P 2H 16D 18M

Figure B.161: Actions performed by participant 8 to complete Scenario 1 in the sketching system and the KLM operators required by each action.

Participant 8 Scenario 2 MassMotion

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.162: Scenario 2 tasks, number of attempts made by participant 8 and efficiency percentage using the MassMotion system.

Scenario 2	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Move Camera	5	0	0	0	3	1	0	5K 3D 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Portal	15	3	3	1	0	1	0	15K 3VS 3P 1H 1M
Move Portal	7	0	3	0	3	1	0	7K 3P 3D 1M
Scale Portal	1	0	2	0	2	1	0	1K 2P 2D 1M
Move Portal	12	0	6	0	13	1	0	12K 6P 13D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Duplicate Portal	7	5	6	0	0	1	0	7K 5VS 6P 1M
Move Portal	12	0	8	0	22	1	0	12K 8P 22D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	11	0	1	0	2	1	0	11K 1P 2D 1M
Rename Portal	38	6	6	2	2	1	0	38K 6VS 6P 2H 2D 1M
Duplicate Portal	2	2	2	0	0	1	0	2K 2VS 2P 1M
Move Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Rotate Portal	6	0	2	0	2	1	0	6K 2P 2D 1M
Move Portal	8	0	3	0	3	1	0	8K 3P 3D 1M
Rename Portal	19	4	5	1	2	1	0	19K 4VS 5P 1H 2D 1M
Duplicate Portal	2	2	2	0	0	1	0	2K 2VS 2P 1M
Move Portal	1	0	1	0	2	1	0	1K 1P 2D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	9	0	1	0	4	1	0	9K 1P 4D 1M
Rename Portal	13	3	4	1	1	1	0	13K 3VS 4P 1H 1D 1M
Duplicate Portal	4	3	4	0	1	1	0	4K 3VS 4P 1D 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rename Portal	7	3	3	1	0	1	0	7K 3VS 3P 1H 1M
Rename Portal	9	3	3	1	0	1	0	9K 3VS 3P 1H 1M
Move Portal	8	0	1	0	1	1	0	8K 1P 1D 1M
Duplicate Portal	7	2	2	0	0	1	0	7K 2VS 2P 1M
Rename Portal	12	4	4	1	0	1	0	12K 4VS 4P 1H 1M
Move Portal	1	0	2	0	4	1	0	1K 2P 4D 1M
Select Activities Tab	3	2	2	0	0	1	0	3K 2VS 2P 1M
Create Journey	3	2	2	0	0	1	0	3K 2VS 2P 1M
Set # Agents	12	3	3	1	0	1	0	12K 3VS 3P 1H 1M
Set Rate	7	1	1	1	0	1	0	7K 1VS 1P 1H 1M
Set Origin	9	7	7	0	0	1	0	9K 7VS 7P 1M
Set Destination	9	6	6	0	0	1	0	9K 6VS 6P 1M
Duplicate Journey	12	4	4	1	0	1	0	12K 4VS 4P 1H 1M
Duplicate Journey	27	4	4	1	0	1	0	27K 4VS 4P 1H 1M
Set Origin	8	5	5	0	0	1	0	8K 5VS 5P 1M
Set Origin	8	5	5	0	0	1	0	8K 5VS 5P 1M
Rename Portal	6	4	4	0	0	1	0	6K 4VS 4P 1M
Select Sim Tab	4	2	2	0	0	1	0	4K 2VS 2P 1M
Compile Simulation	4	4	4	0	0	1	1	4K 4VS 4P 1M 1R
Run Simulation	2	2	2	0	0	1	0	2K 2VS 2P 1M
Watch Simulation	2	0	0	0	1	1	0	2K 1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	344	94	135	12	75	49	1	344K 94VS 135P 12H 75D 49M 1R

Figure B.163: Actions performed by participant 8 to complete Scenario 2 in MassMotion and the KLM operators required by each action.

Participant 8 Scenario 2 Sketching

Scenario 2 Tasks	Attempts	%
Create and rename the entrances	3	100
Create and rename the exits	3	100
Spawn 900 pedestrians per entrance at a 3 pedestrians/sec rate	3	100
For each entrance, equally divide the crowd into 3 groups. Each group should move towards a different exit	3	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.164: Scenario 2 tasks, number of attempts made by participant 8 and efficiency percentage using the sketching system.

Scenario 2	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Select Entrance/Exit	5	1	1	0	2	1	0	5K 1VS 1P 2D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Create Entrance/Exit	9	0	1	0	5	1	0	9K 1P 5D 1M
Create Entrance/Exit	2	0	1	0	4	1	0	2K 1P 4D 1M
Select Select	4	1	1	0	1	1	0	4K 1VS 1P 1D 1M
Rename Entrance	14	1	2	1	0	1	0	14K 1VS 2P 1H 1M
Rename Entrance	15	1	2	1	0	1	0	15K 1VS 2P 1H 1M
Set rate	3	1	1	1	0	1	0	3K 1VS 1P 1H 1M
Set # pedestrians	7	1	1	0	0	1	0	7K 1VS 1P 1M
Set rate	4	1	2	0	0	1	0	4K 1VS 2P 1M
Set # pedestrians	5	1	1	1	0	1	0	5K 1VS 1P 1H 1M
Rename Entrance	17	1	2	1	3	1	0	17K 1VS 2P 1H 3D 1M
Set rate	2	1	1	0	0	1	0	2K 1VS 1P 1M
Set # pedestrians	6	2	2	0	0	1	0	6K 2VS 2P 1M
Select Entrance/Exit	5	1	1	0	2	1	0	5K 1VS 1P 2D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Create Entrance/Exit	3	0	1	0	1	1	0	3K 1P 1D 1M
Create Entrance/Exit	3	0	1	0	2	1	0	3K 1P 2D 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	12	1	2	1	0	1	0	12K 1VS 2P 1H 1M
Rename Exit	11	1	2	1	1	1	0	11K 1VS 2P 1H 1D 1M
Rename Exit	20	1	2	1	3	1	0	20K 1VS 2P 1H 3D 1M
Set Exit %	24	4	5	2	6	1	0	24K 4VS 5P 2H 6D 1M
Set Exit %	20	7	8	0	6	1	0	20K 7VS 8P 6D 1M
Set Exit %	7	2	4	0	0	1	0	7K 2VS 4P 1M
Start Simulation	7	4	7	0	5	1	0	7K 4VS 7P 5D 1M
Watch Simulation	9	1	1	0	7	1	0	9K 1VS 1P 7D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	218	36	56	10	48	28	0	218K 36VS 56P 10H 48D 28M

Figure B.165: Actions performed by participant 8 to complete Scenario 2 in the sketching system and the KLM operators required by each action.

Participant 8 Scenario 3 MassMotion

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	3	66.66666667
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	1	100
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	1	100
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	1	100
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.166: Scenario 3 tasks, number of attempts made by participant 8 and efficiency percentage using the MassMotion system.

Scenario 3	Operators							
Actions	K	VS	P	H	D	M	R	KLM
Move Camera	6	0	0	0	0	1	0	6K 1M
Select Scene Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Portal	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Portal	4	0	2	0	2	1	0	4K 2P 2D 1M
Scale Portal	2	0	5	0	5	1	0	2K 5P 5D 1M
Rotate Portal	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Portal	6	0	3	0	7	1	0	6K 3P 7D 1M
Duplicate Portal	3	2	3	0	0	1	0	3K 2VS 3P 1M
Move Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Rename Portal	24	3	3	2	0	1	0	24K 3VS 3P 2H 1M
Rename Portal	17	3	3	1	0	1	0	17K 3VS 3P 1H 1M
Duplicate Portal	2	2	2	0	2	1	0	2K 2VS 2P 2D 1M
Rename Portal	13	3	3	1	0	1	0	13K 3VS 3P 1H 1M
Move Portal	3	0	1	0	2	1	0	3K 1P 2D 1M
Rotate Portal	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Portal	3	0	3	0	6	1	0	3K 3P 6D 1M
Create Floor	7	1	1	0	7	1	0	7K 1VS 1P 7D 1M
Move Floor	2	0	1	0	1	1	0	2K 1P 1D 1M
Create Link	2	1	1	0	0	1	0	2K 1VS 1P 1M
Scale Link	4	0	3	0	3	1	0	4K 3P 3D 1M
Move Link	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Link	2	0	1	0	1	1	0	2K 1P 1D 1M
Move Link	4	0	4	0	7	1	0	4K 4P 7D 1M
Move Floor	3	0	2	0	1	1	0	3K 2P 1D 1M
Move Link	10	1	4	2	2	1	0	10K 1VS 4P 2H 2D 1M
Rename Link	8	4	5	1	0	1	0	8K 4VS 5P 1H 1M
Duplicate Link	2	2	2	0	0	1	0	2K 2VS 2P 1M
Rename Link	29	4	4	2	0	1	0	29K 4VS 4P 2H 1M
Rename Floor	12	3	3	1	0	1	0	12K 3VS 3P 1H 1M
Duplicate Link	2	2	2	0	0	1	0	2K 2VS 2P 1M
Move Link	1	0	2	0	2	1	0	1K 2P 2D 1M
Rename Link	13	3	3	1	0	1	0	13K 3VS 3P 1H 1M
Duplicate Area	5	4	4	0	1	1	0	5K 4VS 4P 1D 1M
Move Area	1	0	1	0	1	1	0	1K 1P 1D 1M
Rotate Area	1	0	1	0	1	1	0	1K 1P 1D 1M
Rename Floor	11	3	3	1	0	1	0	11K 3VS 3P 1H 1M
Rename Link	19	3	3	1	0	1	0	19K 3VS 3P 1H 1M
Rename Link	19	3	3	1	0	1	0	19K 3VS 3P 1H 1M
Rotate Link	8	0	1	0	3	1	0	8K 1P 3D 1M
Move Link	3	0	4	0	6	1	0	3K 4P 6D 1M
Move Floor	2	1	2	0	1	1	0	2K 1VS 2P 1D 1M
Move Link	4	2	3	0	1	1	0	4K 2VS 3P 1D 1M

Select Activities Tab	7	1	1	0	2	1	0	7K 1VS 1P 2D 1M
Create Journey	4	2	2	1	0	1	0	4K 2VS 2P 1H 1M
Set # Agents	8	1	1	1	0	1	0	8K 1VS 1P 1H 1M
Set Origin	4	3	3	0	0	1	0	4K 3VS 3P 1M
Set Rate	4	1	1	1	1	1	0	4K 1VS 1P 1H 1D 1M
Set Destination	5	4	4	0	0	1	0	5K 4VS 4P 1M
Duplicate Journey	12	4	4	1	0	1	0	12K 4VS 4P 1H 1M
Set Origin	13	8	8	0	0	1	0	13K 8VS 8P 1M
Create Action	1	1	1	0	0	1	0	1K 1VS 1P 1M
Define Actions	34	27	27	0	0	1	0	34K 27VS 27P 1M
Set Action	9	6	6	0	0	1	0	9K 6VS 6P 1M
Duplicate Action	3	3	3	0	0	1	0	3K 3VS 3P 1M
Define Actions	44	24	24	1	0	1	0	44K 24VS 24P 1H 1M
Set Action	14	11	11	0	0	1	0	14K 11VS 11P 1M
Select Sim Tab	2	1	1	0	0	1	0	2K 1VS 1P 1M
Compile Simulation	4	4	4	0	0	1	0	4K 4VS 4P 1M
Move Floor	17	1	3	0	2	1	0	17K 1VS 3P 2D 1M
Move Link	4	2	3	0	1	1	0	4K 2VS 3P 1D 1M
Delete Link	3	2	2	0	0	1	0	3K 2VS 2P 1M
Move Link	4	2	3	0	1	1	0	4K 2VS 3P 1D 1M
Select Sim Tab	3	1	1	0	0	1	0	3K 1VS 1P 1M
Compile Simulation	4	3	3	0	0	1	1	4K 3VS 3P 1M 1R
Run Simulation	2	2	2	0	0	1	0	2K 2VS 2P 1M
Watch Simulation	5	0	0	0	1	1	0	5K 1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Delete Action	4	2	2	0	0	1	0	4K 2VS 2P 1M
Modify Actions	33	23	23	0	0	1	0	33K 23VS 23P 1M
Remove Action	10	8	8	0	0	1	0	10K 8VS 8P 1M
Select Scene Tab	1	1	1	0	2	1	0	1K 1VS 1P 2D 1M
Create Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Move Barrier	1	0	1	0	1	1	0	1K 1P 1D 1M
Scale Barrier	6	0	3	0	4	1	0	6K 3P 4D 1M
Rotate Barrier	5	0	2	0	2	1	0	5K 2P 2D 1M
Move Barrier	4	0	1	0	2	1	0	4K 1P 2D 1M
Scale Barrier	1	0	1	0	1	1	0	1K 1P 1D 1M
Move Barrier	9	0	1	0	7	1	0	9K 1P 7D 1M
Scale Barrier	2	0	1	0	1	1	0	2K 1P 1D 1M
Watch Simulation	0	0	0	0	0	1	0	1M
Select Sim Tab	1	1	1	0	0	1	0	1K 1VS 1P 1M
Compile Simulation	5	4	4	0	0	1	1	5K 4VS 4P 1M 1R
Run Simulation	6	1	1	0	4	1	0	6K 1VS 1P 4D 1M
Watch Simulation	1	0	0	0	0	1	0	1K 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	566	209	268	19	98	85	2	566K 209VS 268P 19H 98D 85M 2R

Figure B.167 (cont.): Scenario 3 tasks, number of attempts made by participant 8 and efficiency percentage using the MassMotion system.

Participant 8 Scenario 3 Sketching

Scenario 3 Tasks	Attempts	%
Create and rename the entrances	2	100
Create and rename the exits	1	100
Create and rename the areas	2	100
Spawn 900 pedestrians per entrance at a 5 pedestrians/sec rate	2	100
Make pedestrians from Entrance A visit Area 1 and wait 10 seconds then move to the exit	1	100
Make pedestrians from Entrance B visit Area 2 and wait 10 seconds then move to the exit	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100
Change the journey of Entrance A to visit and wait in both areas before going to the exit	2	50
Change the journey of Entrance B to move directly to the exit	1	100
Create the barrier	2	50
Run simulation	1	100
Stop the simulation when all agents exit the simulation	1	100

Figure B.168: Scenario 3 tasks, number of attempts made by participant 8 and efficiency percentage using the sketching system.

Scenario 3	Operators							
Actions	**K**	**VS**	**P**	**H**	**D**	**M**	**R**	**KLM**
Select Entrance/Exit	4	1	1	0	1	1	0	4K 1VS 1P 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Entrance	17	1	2	1	0	1	0	17K 1VS 2P 1H 1M
Rename Entrance	14	1	2	1	0	1	0	14K 1VS 2P 1H 1M
Set rate	2	1	1	0	0	1	0	2K 1VS 1P 1M
Set # pedestrians	6	2	2	0	0	1	0	6K 2VS 2P 1M
Set rate	3	1	2	0	0	1	0	3K 1VS 2P 1M
Set # pedestrians	6	2	2	0	0	1	0	6K 2VS 2P 1M
Select Entrance/Exit	1	1	1	0	1	1	0	1K 1VS 1P 1D 1M
Create Entrance/Exit	1	0	1	0	0	1	0	1K 1P 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Exit	11	2	3	1	0	1	0	11K 2VS 3P 1H 1M
Select Area	1	1	1	0	1	1	0	1K 1VS 1P 1D 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Create Area	0	0	0	0	1	1	0	1D 1M
Select Select	1	1	1	0	0	1	0	1K 1VS 1P 1M
Rename Area	10	1	2	1	0	1	0	10K 1VS 2P 1H 1M
Rename Area	10	1	2	0	0	1	0	10K 1VS 2P 1M
Set behaviour	2	2	2	0	0	1	0	2K 2VS 2P 1M
Set behaviour	3	2	3	0	0	1	0	3K 2VS 3P 1M
Update Area	7	2	4	0	0	1	0	7K 2VS 4P 1M
Select Storyboard	5	2	4	0	0	1	0	5K 2VS 4P 1M
Create Storyboard	12	4	7	1	0	1	0	12K 4VS 7P 1H 1M
Create Storyboard	9	4	4	1	1	1	0	9K 4VS 4P 1H 1D 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	4	1	1	0	0	1	0	4K 1VS 1P 1M
Update Area	5	4	5	0	0	1	0	5K 4VS 5P 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Select Storyboard	2	2	2	0	0	1	0	2K 2VS 2P 1M
Create Storyboard	11	4	7	1	0	1	0	11K 4VS 7P 1H 1M
Delete Storyboard	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Storyboard	10	2	6	1	0	1	0	10K 2VS 6P 1H 1M
Delete Storyboard	3	3	3	0	0	1	0	3K 3VS 3P 1M
Create Storyboard	9	2	4	1	0	1	0	9K 2VS 4P 1H 1M
Select Barrier	1	1	1	0	0	1	0	1K 1VS 1P 1M
Create Barrier	4	2	2	0	5	1	0	4K 2VS 2P 5D 1M
Start Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
Watch Simulation	3	0	0	0	1	1	0	3K 1D 1M
Stop Simulation	1	1	1	0	0	1	0	1K 1VS 1P 1M
	186	60	87	9	12	41	0	186K 60VS 87P 9H 12D 41M

Figure B.169: Actions performed by participant 8 to complete Scenario 3 in the sketching system and the KLM operators required by each action.

SUS MassMotion results

This section shows the results of the System Usability Survey answered by the participants after completing the experiment in MassMotion.

ID	1					
Question	Strongly Disagree				Strongly Agree	Score
1				x		3
2	x					4
3				x		3
4				x		1
5				x		3
6		x				3
7				x		3
8			x			2
9				x		3
10		x				3
					Score	70

Figure B.170: Participant 1 System Usability Survey results of MassMotion.

ID	2					
Question	Strongly Disagree				Strongly Agree	Score
1				x		3
2			x			2
3		x				1
4					x	0
5				x		3
6			x			2
7			x			2
8				x		1
9			x			2
10					x	0
					Score	40

Figure B.171: Participant 2 System Usability Survey results of MassMotion.

ID	3					
Question	Strongly Disagree				Strongly Agree	Score
1					x	4
2	x					4
3					x	4
4	x					4
5					x	4
6	x					4
7					x	4
8	x					4
9					x	4
10		x				3
					Score	97.5

Figure B.172: Participant 3 System Usability Survey results of MassMotion.

ID	4					
Question	Strongly Disagree				Strongly Agree	Score
1			x			2
2			x			2
3		x				1
4			x			2
5					x	4
6		x				3
7			x			2
8			x			2
9			x			2
10				x		1
					Score	52.5

Figure B.173: Participant 4 System Usability Survey results of MassMotion.

ID	5					
Question	Strongly Disagree				Strongly Agree	Score
1					x	4
2	x					4
3					x	4
4			x			2
5				x		3
6	x					4
7				x		3
8	x					4
9					x	4
10			x			2
					Score	85

Figure B.174: Participant 5 System Usability Survey results of MassMotion.

ID	6					
Question	Strongly Disagree				Strongly Agree	Score
1	x					0
2					x	0
3		x				1
4					x	0
5	x					0
6		x				3
7	x					0
8			x			2
9	x					0
10					x	0
					Score	15

Figure B.175: Participant 6 System Usability Survey results of MassMotion.

ID	7					
Question	Strongly Disagree				Strongly Agree	Score
1				x		3
2			x			2
3			x			2
4					x	0
5				x		3
6		x				3
7		x				1
8			x			2
9			x			2
10					x	0
					Score	45

Figure B.176: Participant 7 System Usability Survey results of MassMotion.

ID	8					
Question	Strongly Disagree				Strongly Agree	Score
1					x	4
2		x				3
3			x			2
4					x	0
5					x	4
6		x				3
7		x				1
8				x		1
9				x		3
10					x	0
					Score	52.5

Figure B.177: Participant 8 System Usability Survey results of MassMotion.

SUS Sketching results

This section shows the results of the System Usability Survey answered by the participants after completing the experiment in the sketching system.

ID	1					
Question	Strongly Disagree				Strongly Agree	Score
1				x		3
2		x				3
3					x	4
4	x					4
5				x		3
6		x				3
7					x	4
8		x				3
9				x		3
10	x					4
					Score	85

Figure B.178: Participant 1 System Usability Survey results of the sketching approach.

ID	2					
Question	Strongly Disagree				Strongly Agree	Score
1					x	4
2	x					4
3					x	4
4		x				3
5				x		3
6	x					4
7					x	4
8	x					4
9					x	4
10		x				3
					Score	92.5

Figure B.179: Participant 2 System Usability Survey results of the sketching approach.

ID	3					
Question	Strongly Disagree				Strongly Agree	Score
1					x	4
2	x					4
3					x	4
4	x					4
5					x	4
6	x					4
7					x	4
8	x					4
9					x	4
10	x					4
					Score	100

Figure B.180: Participant 3 System Usability Survey results of the sketching approach.

ID	4					
Question	Strongly Disagree				Strongly Agree	Score
1				x		3
2				x		1
3					x	4
4		x				3
5				x		3
6	x					4
7				x		3
8			x			2
9				x		3
10	x					4
					Score	75

Figure B.181: Participant 4 System Usability Survey results of the sketching approach.

ID	5					
Question	Strongly Disagree				Strongly Agree	Score
1				x		3
2	x					4
3					x	4
4		x				3
5				x		3
6		x				3
7					x	4
8	x					4
9					x	4
10	x					4
					Score	90

Figure B.182: Participant 5 System Usability Survey results of the sketching approach.

ID	6					
Question	Strongly Disagree				Strongly Agree	Score
1					x	4
2	x					4
3				x		3
4			x			2
5				x		3
6	x					4
7					x	4
8			x			2
9					x	4
10		x				3
					Score	82.5

Figure B.183: Participant 6 System Usability Survey results of the sketching approach.

ID	7					
Question	Strongly Disagree				Strongly Agree	Score
1					x	4
2	x					4
3					x	4
4		x				3
5				x		3
6	x					4
7				x		3
8		x				3
9					x	4
10	x					4
					Score	90

Figure B.184: Participant 7 System Usability Survey results of the sketching approach.

ID	8					
Question	Strongly Disagree				Strongly Agree	Score
1				x		3
2	x					4
3				x		3
4		x				3
5				x		3
6		x				3
7					x	4
8			x			2
9			x			2
10	x					4
					Score	77.5

Figure B.185: Participant 8 System Usability Survey results of the sketching approach.