IMMERSIVE REAL-TIME MULTI-USER INTERACTION WITH COMPUTER SIMULATED PEDESTRIANS DURING EMERGENCIES

SIMO HANNU HAASANEN

A thesis submitted in partial fulfilment of the
requirements of the University of Greenwich
for the Degree of Doctor of Philosophy
in Computing and Mathematical Sciences

August 2018

DECLARATION

"I certify that the work contained in this thesis, or any part of it, has not been accepted in substance for any previous degree awarded to me, and is not concurrently being submitted for any degree other than that of Doctor of Philosophy being studied at the University of Greenwich. I also declare that this work is the result of my own investigations, except where otherwise identified by references and that the contents are not the outcome of any form of research misconduct."

11/12/2019	[Signature redacted]
Date	Simo Haasanen (Student)
11/12/2019	[Signature redacted]
Date	Dr Peter Lawrence (First Supervisor)
11/12/2019	[Signature redacted]
Date	Professor Edwin Galea (Second Supervisor)
13/12/2019	[Signature redacted]
Date	Dr Steven Gwynne (External Supervisor)

ACKNOWLEDGEMENTS

I'd like to thank:

My God, my Saviour and Redeemer, Lord Jesus Christ, for everything!

My wife, Chido Noreen Haasanen, for your love and support, and proof reading.

First supervisor and colleague Dr Peter Lawrence who expertly took over the main supervisor role from Dr Steven Gwynne. Having worked with Dr Peter Lawrence over 10 years, he excels in supervisory and management skills to meet deadlines. Somehow, he always finds time for his students even when he is drowning in other work. The continued and detailed feedback from Dr Peter Lawrence has made this work so much better.

Dr Steven Gwynne. My primary supervisor for the first half of this research, until his departure across the Atlantic. He really helped to steer the research in the right direction, which led to a successful EU-Project, AUGGMED. Fortunately, Dr Steven Gwynne agreed to stay as my external supervisor as this research would be so much weaker without him.

Second supervisor and line manager Professor Edwin Galea for getting me into this research and getting it funded.

My colleagues from Fire Safety Engineering Group (FSEG) at the University of Greenwich for their expertise and knowledge of the field. With a special mention to Dr John Ewer. He provided the SMARTFIRE expertise and CFD simulations for this thesis, and those late brainstorming sessions after 9pm. Also, Darren Blackshields for his EXODUS expertise and building the main circulation for the airport geometry and being the first test candidate for the IRIS system. Steven Deere, PhD, helped with printing and binding of this thesis. And other colleagues from University of Greenwich who voluntarily took part in the case studies.

All the friends at STwSS and my parents, who have prayed for me.

ABSTRACT

Crowd management is a key issue in large public assemblies in both emergency and non-emergency scenarios. It is important to be able to train for crowd management and test crowd management procedures for both emergency and non-emergency scenarios, however, full-scale exercises are costly, difficult and time consuming, and prevent exposure to immersive fire, smoke and toxic products. Interactive virtual training environments (VTE) with validated pedestrian dynamics are non-existent in firefighting and law enforcement training. Human behaviour is often made to appear real in VTEs rather than qualitatively correct and direct interaction with simulated agents is either not possible or limited to one-to-one interaction.

An Immersive Real-time Interactive Simulation (IRIS) system is proposed to link a validated pedestrian behaviour computer model with an interactive virtual reality (VR) environment and Computational Fluid Dynamic Fire data produced by a CFD Engine. This 3-way link enables new use cases for using pedestrian behaviour models in new fields, for example real-time immersive and interactive first responder training with reactive crowd behaviour in fire scenarios. The linking of the software components also enables rapid prototyping, development, testing and verification of new behaviours and capabilities that are not currently present in the linked computer model. The IRIS system allows the model to be used for testing and prototyping of crowd procedures and theories in an interactive multi-user VR environment.

To improve immersion in interactive VR fire scenarios, it is important to visualise smoke accurately and to represent the impact of heat and toxic products of the fire on the simulated population and user-controlled agents. A novel, real-time, accurate fire smoke visibility visualisation system is presented. Previously, accurate fire smoke visibility visualisation has been limited to expensive ray tracing algorithms as a post-processing tool, not allowing real-time virtual environments to benefit from accurate fire smoke visibility representation.

CONTENTS

DECLARATION ii

ACKNOWLEDGEMENTS iii

ABSTRACT iv

TABLES xiii

FIGURES xvi

1 INTRODUCTION 1

1.1 Motivation 1

1.2 Research questions and objectives 3

1.3 Contributions to knowledge 5

1.3.1 Integration of three elements (crowd simulation, fire simulation and VR) that can be used and added to flexibly 6

1.3.2 Efficient and realistic smoke rendering 7

1.3.3 Interaction of users with pedestrian simulation via commands 7

1.4 IRIS use cases for different disasters 7

1.4.1 Buildings 8

1.4.2 Cities 8

1.4.3 Maritime 9

1.4.4 Aircraft 11

1.4.5 Trains 11

1.5 Thesis structure 12

1.6 AUGGMED EU-Project 14

2 LITERATURE REVIEW **16**

2.1 Evacuation and pedestrian dynamics 17

2.1.1 Ingress, circulation and egress (ICE) 18

2.1.2 The understanding of crowd behaviour 18

2.1.3 Effects of Authority 24

2.1.4 Evacuation and pedestrian dynamics summary 25

2.2 Human behaviour in fire smoke 25

2.2.1 Visibility in fire smoke 26

2.2.2 Normalised soot density as an indication of visibility in smoke 30

2.2.3 Human behaviour in fire smoke summary 30

2.3 Virtual training environments (VTE) 31

2.3.1 Full-scale training 32

2.3.2 Serious games (SG) 32

2.3.3 VTEs in law enforcement 34

2.3.4 VTEs for learning spatial knowledge 35
2.3.5 VTEs for firefighting 37
2.3.6 VR in pedestrian dynamics 43
2.3.7 Interaction interface with virtual characters 44
2.3.8 Natural graphics 46
2.3.9 Virtual training environments summary 47
2.4 Computer models and simulations 48
2.5 Summary 50

3 THE SYSTEM **53**

3.1 System architecture 55
3.1.1 CPU thread model 56
3.2 Network design 60
3.2.1 Server network library implementation design 61
3.2.2 Client network library implementation design 62
3.3 Network protocol implementation details 63
3.3.1 Network protocol message container 65
3.3.2 Load scenario message 66
3.3.3 Load fire message 67
3.3.4 Set flow rate message 67
3.3.5 Set agent count message 68

3.3.6 Time update message 69
3.3.7 Start alarm message 70
3.3.8 Start fire message 70
3.3.9 Add agent message 72
3.3.10 Move agent message 73
3.3.11 Remove agent message 74
3.3.12 Kill agent message 74
3.3.13 Add client agent message 75
3.3.14 Confirm the addition of client agent message 75
3.3.15 Gesture and voice command message 76
3.3.16 Update agent facing direction message 78
3.3.17 Network message optimisations considerations 78
3.4 Client-Server architecture development phases 80
3.5 Summary 83

4 GRAPHICS FIDELITY **85**

4.1 Upgrade from Direct3D 9 to OpenGL 86
4.2 New 3D models 88
4.3 Translucent blending equations for smoke rendering 91
4.4 Multi-pass smoke rendering phases 94
4.5 Smoke particle texture animation 98

4.6 Calibrating the smoke rendering 101

4.7 Smoke validation 107

4.8 Aircraft fire 117

4.9 Smoke rendering performance tests 131

4.10 Summary 132

5 USER INTERACTION **134**

5.1 User interaction capabilities 135

5.1.1 Control areas 139

5.1.2 Likelihood of agents obeying a command 142

5.2 Verification of catchment area and line of sight 146

5.3 Base case for hand gesture and voice command verifications 149

5.4 Evacuate in direction command verification 151

5.5 Stop and Go commands verification 154

5.6 Get down and Get up commands verification 161

5.7 Start evacuating command verification 169

5.8 Get out of the way command verification 172

5.9 Move over there command verification 174

5.10 Summary 180

6 CASE STUDIES **182**

6.1 Geometry description 183

6.2 Population circulation 185

6.3 Performance measurements 188

6.3.1 Preliminary software performance tests 188

6.3.2 Base case performances 190

6.4 Participants 193

6.5 Scenario: Assisted Airport Evacuation 195

6.5.1 Scenario Description 195

6.5.2 Subject performance 200

6.6 Scenario: Secure an area 206

6.6.1 Scenario description 206

6.6.2 Subject performance 211

6.7 General scenario participant feedback 215

6.8 The validity of the pedestrian simulation software 217

6.8.1 Validity of first-person view 218

6.8.2 User interaction effect on validation 218

6.9 Summary 220

7 CONCLUSIONS AND FUTURE WORK 222

7.1 Key research questions revisited 224

7.2 Research related future work 228

7.2.1 Calibration of hand gestures and voice commands 228

7.2.2 Validation of hand gestures and voice commands 229

7.2.3 More comprehensive case studies with more participants 230

7.2.4 Comparison of the IRIS system to real-world training 230

7.2.5 Compare the stochastic model with a deterministic model for further case studies 231

7.2.6 Sensitivity analysis of the gestures and commands 233

7.3 Implementation related future work 233

7.3.1 Graphical Fidelity 233

7.3.2 Hardware and equipment 237

7.3.3 Audio feedback 237

7.3.4 Optimisation 238

7.3.5 Commercialisation 239

7.3.6 Additional hand gestures and voice commands 239

7.3.7 Network connection security 240

REFERENCES **242**

APPENDICES **265**

A AIRPORT SMARTFIRE SIMULATIONS **266**

B UNIVERSITY OF GREENWICH RESEARCH ETHICS COMMITTEE ACCEPTANCE LETTER **272**

C QUESTIONNAIRE FOR USE CASE STUDIES **274**

D USE CASE PERFORMANCE DATA FROM PARTICIPANTS 287

D.1 Assisted airport evacuation graphs 288

D.1.1 Assisted evacuation without smoke 288

D.1.2 Assisted evacuation in smoke 291

D.2 Secure an area 294

D.2.1 Group 1 footfall 294

D.2.2 Group 2 footfall 296

D.2.3 Group 3 footfall 297

D.2.4 Group 4 footfall 299

D.2.5 Group 5 footfall 300

E SMOKE FIGURES USED IN THE QUESTIONNAIRE 302

F VREXODUS DESCRIPTION 315

F.1 History of vrEXODUS 315

F.1.1 Current state and the limitations of vrEXODUS 318

TABLES

1.1 Maritime vessel incidents 10

2.1 Minimum refinement levels for movement in evacuation models to be suitable for immersion and interaction 49

3.1 Network protocol message container 65

3.2 Load scenario message body 66

3.3 Load fire message body 67

3.4 Set flow rate message body 67

3.5 Set agent count message body 68

3.6 Set current simulation time message body 69

3.7 Alarm started message body 70

3.8 Fire started message body 71

3.9 Add agent message body 72

3.10 Move agent message body 73

3.11 Remove agent message body 74

3.12 Kill agent message body 74

3.13 Add client agent message 75

3.14 Confirm the addition of client agent message 75

3.15 Gesture and voice command message 77

3.16 Update agent facing direction message 78

4.1 Smoke validation: Observed visibility variations in the number of particles per metre 116

4.2 Smoke rendering performance results on 2 different PCs 132

5.1 Crowd control commands 136

5.2 Command attribute definitions 145

5.3 Shared command attributes 151

6.1 Airport performance test 189

6.2 Summary statistics for comparing total evacuation time (TET) between the base case with smoke (smoke column) and without smoke (clear column) based on the 1000 randomised simulations. 191

6.3 Assisted evacuation without smoke: total evacuation time (TET) results in seconds 200

6.4 Assisted evacuation in smoke: total evacuation time (TET) results in seconds 202

6.5 Assisted evacuation use case participant questions 205

6.6 Secure an area use case participant questions 213

6.7 General participant questions 215

D.1 Assisted evacuation without smoke: total evacuation time (TET) results in seconds 291

D.2 Assisted evacuation in smoke: total evacuation time (TET) results in seconds 294

FIGURES

1.1 System overview with 3-way link. 6

3.1 System overview to chapter and section mapping. New components introduced in this thesis are coloured in blue. The existing components, green, red and orange coloured boxes are as described previously in Chapter 1 Section 1.3.1. 54

3.2 Multithreaded process model of vrEXODUS 56

3.3 Server-client architecture 60

3.4 Server-side network design 61

3.5 Client-side network design 62

3.6 The first version of the network model implementation 81

3.7 vrEXODUS connected to real EXODUS server 82

4.1 Graphical improvement steps 88

4.2 New 3D agent models 89

4.3 Animation flipbook compiler 89

4.4 Parallel visualisation and updating of graphics data from multiple threads with a shared OpenGL context. 90

4.5 Multi-pass smoke rendering 95

4.6 Combining the visualisation of heat and toxic products on agents with the multi-pass smoke visibility visualisation. 97

4.7 Smoke CFD and rendering in Blender 98

4.8 The inverted texture of smoke particle at frame 300 of the animation. The darkest area is roughly about a 3^{rd} in width and in height of the total width and height. 99

4.9 The texture histogram. Shows the number of pixels representing each colour value. Almost 10% of the colour (below 25) is missing (i.e. not utilised). The white colour count, values 254 and 255, are 146,985 and 146,726 respectively and off the chart to keep the scale of the chart usable. 100

4.10 Simplex noise colour adjustment 105

4.11 Smoke particle billboard system. The green billboards are visible to the user. 106

4.12 Visibility of unlit sign to light extinction coefficient. 108

4.13 Smoke test geometry in EXODUS 109

4.14 5cm long test image 110

4.15 Smoke validation: 2 particles per metre with varying light extinction coefficients 112

4.16 Figure 4.15a zoomed in. 113

4.17 Smoke validation: 0.2 light extinction coefficient with varying number of particles per metre 114

4.18 Smoke validation: Black smoke vs. coloured smoke. 1 particle per metre with 0.2 light extinction coefficient. 115

4.19 Smoke validation: Visibility of 3D cow model by participants in comparison to Jin's experiment. 115

4.20 Observed visibility of 3D agent model. 117

4.21 NACRE Blended Wing Body aircraft CAD drawing 118

4.22 CAD drawing in EXODUS, including seats 119

4.23 CAD drawing in EXODUS with movable space in green 119

4.24 3D view: Outside view of the NACRE aircraft in vrEXODUS 121

4.25 Close up outside view of the NACRE aircraft in vrEXODUS 121

4.26 Internal NACRE aircraft smoke visualisation in vrEXODUS with 2 layer smoke representation 122

4.27 NACRE aircraft: Internal smoke visualisation in vrEXODUS with 10-layer smoke representation 123

4.29 NACRE aircraft: Internal smoke visualisation in vrEXODUS in key locations 124

4.28 Difference between 2 (on the left) and 10 (on the right) vertical layers of smoke.125

4.30 NACRE aircraft: Side angle smoke progression visualisation in vrEXODUS . 126

4.31 NACRE aircraft: Side angle smoke progression visualisation in vrEXODUS . 127

4.32 NACRE aircraft: Top down angle smoke progression visualisation in vrEXODUS128

4.33 NACRE aircraft: Top down angle smoke progression visualisation in vrEXODUS129

4.34 NACRE aircraft: Side angle smoke visualisation in vrEXODUS as empty and full 130

4.35 NACRE aircraft: Top down angle smoke visualisation in vrEXODUS as empty and full 130

5.1 Stop, Go, Get down and Get up commands shown in vrEXODUS 137

5.2 Start evacuating, Evacuate that way, Get out of the way and Move over there commands shown in vrEXODUS 138

5.3 Hand gesture (blue) and voice command (red) influence areas 139

5.4 Starting point, where agents A and B are preparing to move towards exit Door_1 146

5.5 EXODUS view: line of sight verification case. Agent B has stopped in the gesture influence area. The yellow area shows the combined area of both the hand gesture and voice command. 147

5.6 3D view: User can see the agent in corridor B, but cannot see the agent in corridor A. 148

5.7 The default behaviour of the base case. 149

5.8 User standing in front of Door_1 redirected the crowd to Door_2. 152

5.9 3D view: Whole crowd is redirected to different exit. 152

5.10 3D view: Single user is selected and issued a command. 153

5.11 Single user is redirected to Door_2. 154

5.12 3D view: User issued a Stop command to all agents in the catchment area. . 155

5.13 3D view: User issued a Stop command to all agents in the catchment area. . 155

5.14 EXODUS view: User issued a Stop command to all agents in the catchment area. 156

5.15 EXODUS view: User issued a Go command to all agents in the catchment area. 157

5.16 EXODUS view: User issued a Go command to all agents in the catchment area. 157

5.17 3D view: User is selecting an agent before issuing a command. 158

5.18 3D view: User issued a Stop command to an individual agent. 159

5.19 3D view: User issued a Stop command to an individual agent. 159

5.20 3D view: User issued a Stop command to an individual agent. 160

5.21 EXODUS view: User issued a Stop command to an individual agent. 161

5.22 3D view: User issued a Get down command to all agents in the catchment area. 162

5.23 3D view: User issued a Get down command to all agents in the catchment area. 163

5.24 3D view: User issued a Get down command to all agents in the catchment area. 163

5.25 3D view: User issued a Get down command to all agents in the catchment area. 164

5.26 3D view: User issued a Get up command to all agents in the catchment area. 164

5.27 3D view: User issued a Get up command to all agents in the catchment area. 165

5.28 3D view: User is selecting an agent before issuing a command. 166

5.29 3D view: User issued a Get down command to an individual agent. 166

5.30 3D view: User issued a Get down command to an individual agent. 167

5.31 3D view: User issued a Get down command to an individual agent. 168

5.32 EXODUS view: User issued a Get down command to an individual agent. . . 168

5.33 3D view: User issued a Get up command to an individual agent. 169

5.34 EXODUS view: Evacuation simulation at 27.8 seconds with a normal response and without user influence. 170

5.35 EXODUS view: User issued a Start evacuating command to all agents in the catchment area at 5.3 seconds into the simulation. 170

5.36 EXODUS view: Evacuation at 27.7 seconds when the user started issuing a Start evacuating command to all agents in the catchment area from 5.3 seconds into the simulation. 171

5.37 3D view: User issued a Start evacuating command to all agents in the catchment area. 171

5.38 3D view: User issued a Start evacuating command to all agents in the catchment area. 172

5.39 EXODUS view: User issued a Get out of the way command to all agents in the catchment area. (Without visible influence areas) 173

5.40 EXODUS view: User issued a Get out of the way command to all agents in the catchment area. (With visible influence areas) 173

5.41 3D view: User issued a Get out of the way command to all agents in the catchment area. 174

5.42 EXODUS view: User issued a move over there command to all agents in the catchment area. 175

5.43 3D view: User issued a Move over there command to all agents in the catchment area. 176

5.44 EXODUS view: User issued a move over there command to all agents in the catchment area to move to a new location. 177

5.45 3D view: User issued a Move over there command to all agents in the catchment area to move to a new location. 178

5.46 3D view: User issued a Move over there command repeatedly to an individual computer-controlled agent. 179

5.47 EXODUS view: User issued a Move over there command repeatedly to an individual computer-controlled agent. 180

6.1 Aerial view of Pirkkala airport in December 2014, from Finavia website[1] . . 184

6.2 The airport geometry in EXODUS with partitioned areas 185

6.3 The airport geometry in EXODUS with population 187

6.4 Footfall heat map of most used areas at 23 minutes into the simulation . . . 187

6.5 The mean evacuation time for each agent in the simulation from 1000 runs with and without smoke. 191

6.6 Histogram of total evacuation times (TET) of the airport geometry without smoke based on 1000 simulations. 192

6.7 Histogram of total evacuation times (TET) of the airport geometry with smoke based on 1000 simulations. 193

6.8 Example routes adopted by two users during the evacuation 195

6.9 3D view of the example users issuing Start Evacuating command. Red area indicates the voice command influence zone. Blue area indicates the hand gesture influence zone. The hand gesture and voice command areas overlap in front of the user. 197

6.10 3D view of the example user 2 issuing Start Evacuating command. Red circle-area indicates the voice command influence zone. Blue semicircle-area indicates the hand gesture influence zone. The hand gesture and voice command areas overlap in front of the user. 198

6.11 Seating positions for two participants 199

6.12 3D view of a user in smoke with 8 metres of visibility (i.e. the light extinction coefficient is 0.375) 199

6.13 Assisted evacuation without smoke: Group 1 performance 201

6.14 Assisted evacuation without smoke: Group 2 performance 202

6.15 Assisted evacuation in smoke: Group 1 performance 203

6.16 Assisted evacuation use case participant responses to questions in Table 6.5 . 205

6.17 The location of a suspicious package (red circle above exit E7) and example locations of two user-controlled agents directing other agents away from it . . 206

6.18 The suspicious (red) package in the 3D view highlighted by a yellow circle . 207

6.19 3D view of the example user 2 issuing Evacuate That Way command. Red area indicates the voice command influence zone. Blue area indicates the hand gesture influence zone. The hand gesture and voice command areas overlap in front of the user. 208

6.20 3D view of the example users in bird's-eye view. Red circle-area indicates the voice command influence zone. Blue semicircle-area indicates the hand gesture influence zone. The hand gesture and voice command areas overlap in front of the user. 208

6.21 Footfall contours for the Base Case, and User Case. The location of the suspicious package is indicated by a red circle. 210

6.22 Secure an area: Group 4, Run 1, footfall contour 212

6.23 Secure an area: Group 4, Run 2, footfall contour 212

6.24 Secure an area: Group 4, Run 3, footfall contour 213

6.25 Secure an area use case participant responses to questions in Table 6.6 . . . 214

6.26 General participant responses to questions in Table 6.7 216

6.27 EXODUS sub-model user interaction 219

A.1 The airport geometry provided by AUGGMED project. 267

A.2 Airport mass fraction of smoke graph in SMARTFIRE 268

A.3 Airport temperature graph in SMARTFIRE 268

A.4 vrEXODUS smoke visualisation progress of SMARTFIRE fire simulation in the airport geometry. Part 1. 269

A.5 vrEXODUS smoke visualisation progress of SMARTFIRE fire simulation in the airport geometry. Part 2. 270

A.6 vrEXODUS smoke visualisation progress of SMARTFIRE fire simulation in the airport geometry. Part 3. 271

A.7 vrEXODUS smoke visualisation progress of SMARTFIRE fire simulation in the airport geometry. Part 4. 271

D.1 Assisted evacuation without smoke: Group 1 performance 288

D.2 Assisted evacuation without smoke: Group 2 performance 289

D.3 Assisted evacuation without smoke: Group 3 performance 289

D.4 Assisted evacuation without smoke: Group 4 performance 290

D.5 Assisted evacuation without smoke: Group 5 performance 290

D.6 Assisted evacuation in smoke: Group 1 performance 291

D.7 Assisted evacuation in smoke: Group 2 performance 292

D.8 Assisted evacuation in smoke: Group 3 performance 292

D.9 Assisted evacuation in smoke: Group 4 performance 293

D.10 Assisted evacuation in smoke: Group 5 performance 293

D.11 Secure an area: Group 1, Run 1, footfall contour 294

D.12 Secure an area: Group 1, Run 2, footfall contour 295

D.13 Secure an area: Group 1, Run 3, footfall contour 295

D.14 Secure an area: Group 2, Run 1, footfall contour 296

D.15 Secure an area: Group 2, Run 2, footfall contour 296

D.16 Secure an area: Group 2, Run 3, footfall contour 297

D.17 Secure an area: Group 3, Run 1, footfall contour 297

D.18 Secure an area: Group 3, Run 2, footfall contour 298

D.19 Secure an area: Group 3, Run 3, footfall contour 298

D.20 Secure an area: Group 4, Run 1, footfall contour 299

D.21 Secure an area: Group 4, Run 2, footfall contour 299

D.22 Secure an area: Group 4, Run 3, footfall contour 300

D.23 Secure an area: Group 5, Run 1, footfall contour 300

D.24 Secure an area: Group 5, Run 2, footfall contour 301

D.25 Secure an area: Group 5, Run 3, footfall contour 301

E.1 Relationship between terms in Equation E.2 303

E.2 5cm cow figure, 5m distance, 2 particles per metre 304

E.3 10cm cow figure, 10m distance, 2 particles per metre 305

E.4 15cm cow figure, 15m distance, 2 particles per metre 306

E.5 Agent figure, 15m distance, 2 particles per metre 307

E.6 Agent figure, 10m distance, 2 particles per metre 308

E.7 Agent figure, 2m distance, 2 particles per metre 309

E.8 Agent figure, 7.5m distance, 2 particles per metre 310

E.9 Agent figure, 5m distance, 2 particles per metre 311

E.10 Agent figure, 1m distance, 2 particles per metre 312

E.11 5cm cow figure and person shape agent figure at 5m distance, 0.40 light extinction coefficient, varying number of particles per metre. Left hand images have 1, 3, 5, 7 and 9 particles per metre respectively from top to bottom. Right hand images have 2, 4, 6, 8 and 10 particles per metre respectively from top to bottom. 313

F.1 The last known copies of the first 2 vrEXODUS versions written in World-ToolKit for Windows version 2.0 316

Chapter 1

INTRODUCTION

1.1 Motivation

First responders (such as firefighters and law enforcement officers) are being trained with virtual training environments (VTE) without validated, accurate and realistic representation of human behaviour [1–10]. However, human life is and should be a key factor in any serious life-threatening incident or disaster training. It is thus paramount that the VTEs represent actual human behaviour as accurately as possible to enable training for realistic scenarios where human life is at risk.

It is important that safety procedures involving humans can be tested, evaluated and improved well in advance before it is necessary to utilise them in real life-threatening scenarios. It is also important that both training and testing of procedures can be practised with realistic human behaviour in a virtual environment where the subjects can be exposed to realistic and life-threatening situations such as fire and the products of fire (i.e. smoke, heat and toxic products) [1, 7]. For the training and the testing of procedures to be effective and realistic in fire emergencies, it is crucial that the fire and its effects are credibly represented and are realistic in the virtual environment to enable the subjects to feel fully immersed and hence behave realistically and make natural real-life decisions [1, 7, 11, 12].

Better testing and training could have avoided the failure of crowd management in large public assemblies, such as the Love Parade in Germany in 2010 [13] and the Hillsborough disaster in Britain in 1989 [14]. The failures resulted in fatalities and severe injuries from

crushing crowds. It is important to be able to train, test and evaluate procedures for crowd management to help to avoid repeating these disasters. Training and testing of procedures with large real-life crowds is difficult, time consuming and expensive, and, most importantly, participants behave differently in drills than in real fire emergency situations (i.e. the heightened physiological and psychological stress impacts on decision making) [15]. Some participants have also died or been severely injured in real-life trials [16]. Therefore, it is desirable to train and test procedures for disaster management involving crowds of people in a virtual environment when the simulated crowd is validated against real-life behaviour and will provide realistic results.

Furthermore, the response to real-life incidents and disasters (i.e. fire, terrorism, natural disasters) often involves multiple agencies with numerous personnel working together [17]. Therefore, it is important that the virtual training exercise or the testing of procedures can be done collaboratively and simultaneously by multiple actors in the shared environment working together in the same realistic virtual scenario.

To improve immersion in interactive VR environments in fire emergency scenarios, it is important to visualise smoke accurately and to represent the impact of the toxic products of the fire on the population as well as user-controlled agents to generate representative external conditions. The impact of the hazard on the population and users of the system is achieved by the coupling of the toxicity model with the pedestrian behaviour model.

A novel, real-time, fire smoke visibility visualisation system is presented in Chapter 4 to visualise accurately the light extinction coefficient [18–21] of smoke data as calculated by CFD fire simulation engines [22, 23]. Previously, accurate fire smoke visibility visualisation has been limited to computationally expensive ray tracing algorithms as a post-processing tool [24, 25], not allowing real-time virtual environments to benefit from accurate fire smoke visibility representation. An efficient real-time hardware accelerated fire smoke visibility rendering solution using order-independent rasterised translucent billboard particles with the accurate representation of smoke visibility is proposed and validated with a proposed test case to enable real-time visualisation of fire smoke visibility in VR environments.

To address the highlighted limitations in emergency training and evaluation of proce-

dures in virtual environments, this thesis proposes the **I**mmersive **R**eal-time **I**nteractive **S**imulation (IRIS) to link a validated pedestrian behaviour computer model [26] with an interactive virtual reality (VR) environment, together with Computational Fluid Dynamics fire data produced by a validated CFD fire simulation engine. This 3-way link enables new use cases for using pedestrian behaviour models in new fields, for example, real-time immersive and interactive first responder (i.e. firefighter) training with reactive individual and crowd behaviour in fire scenarios. The linking of the software components also enables rapid prototyping, development, testing and verification of new behaviours and capabilities that are not currently present in the linked computer model, for example enabling user-controlled agents to interact with computer-controlled agents using hand gestures and voice commands. The IRIS system allows the linked model to be used for testing and prototyping of crowd procedures and theories in an interactive multi-user VR environment.

Using the IRIS system, the coordinator will have a clearer overview of the scenario since they can effectively adopt a bird's-eye-view approach, fly around and get in close to what the participants are doing without getting in the way of the participant. Using the data output options of the linked behaviour simulation software can aid the coordinator by providing them with qualitative and quantitative data after the trail. The qualitative and quantitative data from the linked behaviour simulation software can aid the coordinator in their assessment of the participant i.e. in the manner used in the test cases.

The IRIS system and the real-time accurate fire smoke visibility visualisation have wide reaching implications, not only providing a more realistic virtual environment where certain actual emergency procedures and training can be evaluated and performed in a holistic way, but also impacting on more general applications, such as the $100bn video game entertainment industry [27], where more immersive and realistic environments are in demand [28] (i.e. enabling more accurate representation of smoke in real-time).

1.2 Research questions and objectives

The main goal of this research is to identify: ***How can virtual reality training and the evaluation of procedures be improved through the development and use of an***

immersive, real-time interactive simulation (IRIS)?

The objective of this research is to address the current limitations of virtual training environments (namely the lack of validated human behaviour and the lack of accurate visualisation of fire and its products, i.e. smoke, heat and toxic substances) so that the outcome in training, testing and evaluation of procedures in emergency scenarios, such as fire, terrorism, crushing and large scale disasters, is more representative of real world scenarios.

The main objective immediately raises the following sub-questions:

What technologies are available to support the development of an immersive, real-time, interactive simulation (IRIS)?

A solution is proposed in this thesis to utilise existing engineering tools such as validated computational pedestrian behaviour models, hazard models and CFD fire simulations linked together in a novel interactive and reactive VR environment. The linking of the existing software tools as software components provides its own challenges, such as compatibility, suitability and availability, as will be discussed in Chapters 2, 3, 4 and 5.

What types of interaction behaviours are necessary to facilitate crowd interaction?

To achieve realistic interactivity between user-controlled agents and computer-controlled agents in a virtual environment, the interaction should be as representative as possible between humans. However, there are several factors that affect the interactivity, such as gesture or voice, perceived authority or role, comprehension of the interaction, and the reaction of other nearby individuals.

How can the immersion and the realism of the real-time virtual reality in emergency scenarios be improved?

Presenting realistic scenarios in a virtual environment relies on the validation of the elements that make up the scenario, such as people behaviour and accurate representation of hazards such as fire and its products (i.e. smoke, heat and toxic substances) including the impact of these products on the modelled population.

What physical characteristics of a fire hazard can practically be included in virtual reality when applying emergency procedures?

This thesis seeks to establish a new standard in representing the effects (i.e. smoke, heat and toxic substances) of realistic fire hazards on user-controlled and computer-controlled agents in the same VR environment.

1.3 Contributions to knowledge

An immersive real-time interactive simulation (IRIS) system is proposed to link a validated pedestrian movement computer model with an interactive virtual reality (VR) environment. This link enables new use cases for using pedestrian models in new fields, such as virtual training environments, such as in firefighting and law enforcement training. The linking of the software components also enables rapid prototyping, development and testing of new behaviours and capabilities that are not currently present in the computer model; for example, enabling user-controlled agents to interact with computer-controlled agents using hand gestures and voice commands. The IRIS system allows the model to be used for testing and prototyping of crowd procedures and theories in an interactive multi-user virtual environment as the used pedestrian model has been validated.

The novel contributions of this thesis are as follows:

1.3.1 Integration of three elements (crowd simulation, fire simulation and VR) that can be used and added to flexibly

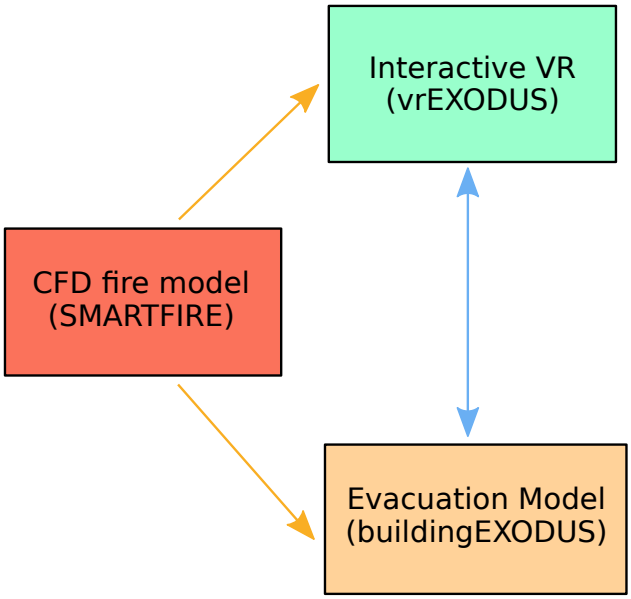


Figure 1.1: System overview with 3-way link.

Figure 1.1 shows how the three elements are linked together as components in a new larger system. The blue two-way arrow indicates the real-time data connection between the evacuation model and the VR environment. The yellow one-way arrow indicates the non-real-time data connection from the CFD fire model to both the evacuation model and the VR environment. The linking allows more realistic and simultaneous fire and agent representation and interactivity in virtual environments, such as in virtual training environments (VTEs), than what has been previously achieved. This allows VR environments to utilise the sophisticated agent behaviour, such as navigation and response behaviour, found in evacuation models. Previous VR environments have provided limited or no interaction with simulated agents. The behaviours of agents have been very primitive and not utilising the current understanding found in literature, which is often built into computer models. Any of the linked elements can be replaced with a different software tool. This has been successfully demonstrated in the EU-Project AUGGMED (see Section 1.6 later in this Chapter) by replacing the vrEXODUS software tool with a Unity3D game engine. Further information on the system architecture is provided in Chapter 3.

1.3.2 Efficient and realistic smoke rendering

In Chapter 4, a new smoke visibility rendering method is proposed utilising a billboard particle system to visualise CFD data in real-time. Typically CFD data is rendered with a slow ray-traced pipeline instead of rasterising pipeline (see Chapter 2). In this research a ray-traced algorithm is adapted to rasterisation to introduce accurate real-time smoke visibility visualisation. This allows an accurate representation of actual smoke visibility in real-time which has not been previously achieved. The visual accuracy is validated through human participants (in Section 4.7, Chapter 4) and demonstrated in a larger case study in Chapter 6. Computational performance impact of the smoke is also recorded in Chapter 4, Section 4.9.

1.3.3 Interaction of users with pedestrian simulation via commands

This thesis proposes an interface to interact with computer-controlled agents through simulated gesture and voice commands. The set of commands and the ability to interact with pedestrian model commands are defined and verified in Chapter 5. The commands were identified from interviews with armed police officers and allow users a more realistic way to interact with an existing evacuation simulation for the testing of procedures and training. The interactivity is demonstrated with multiple human participants in a case study in Chapter 6. How the interaction work affects the validity of the pedestrian simulation software is discussed at the end of Chapter 6 in Section 6.8.

1.4 IRIS use cases for different disasters

The IRIS system has a number of potential use cases and this section identifies some of the example areas of application.

1.4.1 Buildings

The building environment is a key modelling application - most people live in urban environments and spend the majority of their time within buildings. The built environment also represents the greatest variety of challenges - given the scale, complexity and purpose of the structures. These may range from single occupancy dwellings to high-rise structures, from shopping malls to hospitals [29]. Occupants in buildings vary from families to individual dwellers to elderly to disabled people. In public buildings the occupants include members of staff who have different roles, such as security, fire warden, temporary contractors, janitors, commercial staff, with varying levels of training, expertise and familiarity with the structure. The social dynamics, roles, training and familiarity of the building influence the occupant's performance during an incident [30] (see Section 2.1.2). The structural design of a building plays a role in the outcome of an incident (i.e. how the fire, smoke and toxic products travel within the structure) such as the amount of flammable materials available to a fire, the use of sprinkler systems, the choke points on exit routes and the level of occupancy in the building, as was seen in the Rhode Island Nightclub Fire in 2003 [31]. The World Trade Center incident in 2001 [32] had occupants evacuate from as high as the 78^{th} floor in a hot, dark staircase without the assistance available from lifts, which highlights the importance of working emergency lights and the use of lifts in emergency situations. Having to use a dark stair case for 78 floors worth of stairs rules out a number of occupants from being able to evacuate. Kinsey concluded in his PhD thesis in 2011 that including lifts in a high-rise building would reduce the total evacuation time between 33% and 50% [33]. The IRIS system is an ideal virtual training environment for building related incidents, enabling the accurate representation for fire emergency scenarios and immersive simultaneous multi-user roles with individual and crowd interactions.

1.4.2 Cities

City-wide disasters (natural or otherwise) are large-scale events that might include multiple locations and conditions, spreading beyond the initial cause. These include floods, hurricanes, earthquakes, wildfires, volcanic eruptions, avalanches, medical emergencies, and technological

incidents. Recent examples of disaster emergencies include:

- Hurricane Katrina in 2005 affected 90,000 square miles of the United States and killed nearly 2,000 people [34]
- Haiti Earthquake in 2010 and killed almost 160,000 people [35]
- Indian Ocean Tsunami in 2004 killed over 227,000 people in 14 countries around the Indian Ocean with damages up to 3km inland [36].

In the scale and complexity of these city-wide disasters the procedural response will typically involve numerous agencies operating on different levels while providing a range of functions, including responding to the incident, a protection strategy, or addressing the long-term effects of the incident. Each incident allows for its own degree of prior warning, requiring procedural responses (i.e. evacuation vs defending in place), and communication strategies. All of these factors make the organizational response more complicated and reliant upon trained and expert staff [17].

The IRIS system allows the simulation of crowds and potentially even tens or hundreds of thousands of computer-controlled agents (depending on the used behaviour model and the scenario). The IRIS system enables multiple first responders to work simultaneously in tandem with managing the potentially large crowds in disaster events, either for training or for testing procedures.

1.4.3 Maritime

Maritime incidents are complex problems and differ greatly from building or aircraft incidents. This is prevalent when looking at some of the high-profile incidents that resulted in fatalities (see Table 1.1 below).

Table 1.1: Maritime vessel incidents

Year	**Vessel**	**Type of incident**
2014	MV Sewol	Capsize [37]
2012	Costa Concordia	Grounding [38]
1998	Carnival Ecstasy	Fire [39]
1995	Saint Malo Ferry	Grounding [40]
1994	MV Estonia	Equipment failure [41]
1987	MS Herald of Free Enterprise	Human error [42]
1912	RMS Titanic	Collision with iceberg [43]

Ship evacuations are different from building evacuations in a number of ways. These include physical, procedural and experiential issues. For example, vessels are dynamic and can list (either heel or trim) during an incident. This can influence movement rates or the availability of egress routes [44]. Unlike most buildings, maritime evacuations can involve ascending. This would be counter-intuitive in building evacuations. There is also no obvious or simple movement to a place of safety. Maritime procedures dictate that evacuation from the ship is a last resort [45,46], requiring a staged safety procedure [44]. In addition, unfamiliar technology will likely be encountered that can complicate or hinder evacuee performance (such as life-vests, chutes and life rafts). Of particular relevance here is the reliance of the evacuating population upon the ship's crew [47]. The crew will be responsible for managing the evacuation. Although passengers may be subjected to mustering drills, they will likely be unfamiliar with the conditions and procedural details that will be present during a real incident. Therefore, crew expertise is critical as is the ability to manage and deploy them effectively during an incident.

The IRIS system enables for the users to take different roles in the simulation, such as an authoritative role or an individual as part of a crowd. The IRIS system can be used to test procedures or train for different roles in a maritime evacuation.

1.4.4 Aircraft

An aircraft evacuation is highly procedural by design to meet a 90-second safe evacuation certification test [48]. Togher explains: *"that crew procedures developed from the analysis of certification trials where no social bonds exist may become irrelevant during a real emergency where social bonds do become relevant."* [48]. Assistance and instructions from cabin crew in aircraft evacuation improves evacuation time from aircraft [48]. Aircrafts have highly drilled evacuation procedures. Often the crews are highly trained and therefore the whole evacuation process is controlled by the crew.

The IRIS system can be used to train cabin crew and test for procedures in an interactive virtual environment for aircraft evacuation in emergencies. Additionally, the IRIS system enables the use of realistic fire data and the realistic visualisation of the impact of the fire (and its products) on the virtual agents in aircraft emergencies potentially allowing for more realistic evacuation certification tests.

1.4.5 Trains

The railway egress environment is different from the building egress environment. Passengers can be thrown around within the train in an incident [49], which will make their evacuation difficult. They can potentially hinder the evacuation of others. People with disabilities may also hinder the evacuation of others [50]. Train cars may overturn in an incident, making evacuation difficult. When the train overturns, luggage may fall on passengers. The luggage may also hinder the evacuation by potentially blocking an exit route [50–52].

As a result of the crash, the rail car may get damaged, which may make some egress routes no longer possible or stop passenger travel between cars [51, 53, 54]. The environment where the accident happens can have an effect on the evacuation, e.g. a bridge or a tunnel [50, 55]. Passenger car windows may provide additional exit routes. Some trains provide tools to break windows. Windows may become the only exit routes in an overturned car. However, egress through a window may not be easy in an overturned car [50, 54, 56–58].

The IRIS system can be used in a multi-user configuration to train for train evacuation or the testing of procedures, enabling the assuming of different roles and interaction with computer-controlled agents. The IRIS system can also provide a realistic representation of a fire emergency scenario with the use of realistic CFD fire simulation data.

1.5 Thesis structure

This section will give guidance on how to approach this thesis and what to expect to find from each of the following chapters. Each Chapter is summarised briefly, highlighting the purpose and novelty of the research.

Literature review chapter

The Literature Review conducted in Chapter 2 gives an in-depth and critical review of the background to the current understanding and state of the art of people behaviour in emergencies and crowds and the current published theories. Chapter 2 also highlights from the published literature the strengths and weaknesses in current VR environments used for training and testing of procedures of emergency scenarios. It then identifies a set of computer models (i.e. evacuation models, toxicity models, and fire simulation models) that are suitable to enhance current real-time virtual training environments (VTEs).

The system chapter

Chapter 3 provides a system overview by detailing how the different software components are linked together to form a complete IRIS system.

The details include:

- The efficient and extensible real-time asynchronous network protocol to enable simultaneous multi-user interactivity of the IRIS system.

- Multi-threaded system architecture design and implementation that enables the responsive real-time interactivity part of the IRIS system.

Graphics fidelity / visualising smoke visibility chapter

Chapter 4 summarises the key graphical implementation enhancements to the EXODUS software suite. The enhancements were necessary to achieve a working IRIS system, together with the enhancements required to incorporate developments necessary so that human factors/hazards can be considered in VTE (Virtual Training Environment).

Some of the key aspects include:

- Efficient and accurate real-time hardware accelerated smoke rendering as billboard particles with order-independent translucency.
- A smoke visibility validation case is proposed and tested based on the existing published smoke visibility test research.
- Performance results from multiple external participants taking the virtual visibility validation tests are analysed and presented.

User interaction / commands and gestures chapter

Chapter 5 details the user interactivity capabilities and controls to enable crowd and individual agent interactivity through a set of simulated hand gestures and voice commands, which provide the interactivity part of the IRIS system. Each hand gesture and voice command is verified in terms of functionality and completeness.

Case studies chapter

The case studies in Chapter 6 provide a few example use cases for the IRIS system in training and testing of procedures and highlight the benefits of using the IRIS system in more complicated scenarios. The potential use cases presented in Chapter 6 demonstrate the

new capabilities of the IRIS system that address the shortcomings in current state-of-the-art VTEs in published research, which are identified in the literature review in Chapter 2. Summarised results of the data collected from external participants is analysed and presented to support the arguments in this thesis.

Conclusions and future work chapter

Chapter 7 revisits the research questions from Chapter 1 and provides a summary of the research detailed in this thesis. Chapter 7 also provides suggestions for future work that has been identified during this research.

1.6 AUGGMED EU-Project

During the lifetime of this research the AUGGMED EU-Project (Project ID: 653590) was conducted between 1^{st} of June 2015 and 31^{st} of May 2018 [59, 60].

The aim of the AUGGMED was to develop a serious game VR and MR platform that can be used for single and cooperative training in different settings offering different levels of realism and engagement. The platform enables police, security forces and counter-terrorist units as well as first responders to train their staff in different VR environments with different scenarios and apply this training in the real infrastructure environment using mixed reality techniques. It also featured guns, explosions, fire and smoke with various mixed reality and VR headsets and a haptic feedback vest. The main VR client for the project was implemented in Unity3D [59, 60].

Some of the ideas and research presented in this thesis have contributed to the AUGGMED project, such as the network communication protocol and design, interactivity and agent control with EXODUS, and fire data integration from SMARTFIRE. However, no part of the AUGGMED project used the smoke rendering techniques or the algorithm presented in this thesis. The research detailed in this thesis started on 1^{st} of September 2011 and was run until 31^{st} of August 2018.

The AUGGMED project successfully utilised the IRIS system proposed in this thesis. The modularity of the IRIS system let project partners replace the 3D VR environment described in this thesis (namely vrEXODUS) with a Unity3D based custom VR environment. VR and MR (Mixed Reality) hardware, such as headsets, controllers and an impact/feedback vest, were used in the AUGGMED project.

The hand gestures and the voice commands were identified during the AUGGMED project by interviewing armed police officers from the West Yorkshire Police. The necessary gestures and commands for crowd control as identified by officers with real-world experience are: *Stop*, *Go Get down*, *Get up*, *Start evacuating*, *Evacuate in direction*, *Get out of the way*, and *Move over there*. The gestures and commands are further detailed in Chapter 5.

Chapter 2

LITERATURE REVIEW

In the previous chapter the IRIS system was proposed (see Section1.3) with potential use cases (see Section1.4). This chapter provides a critical review of the state of the art in computational evacuation modelling, crowd behaviour and control, serious games and virtual training environments, and fire and smoke simulation and rendering.

This chapter will also seek to identify the limitations of current virtual environments that are being used to train, for example, spatial navigation, firefighting, law enforcement practises, performance-based assessment, re-enactment, research and forensic analysis, and how the IRIS system can be used to improve on these limitations. To identify the features necessary for the IRIS system, it is important to look at how humans are currently interfaced with virtual worlds, including the interaction with virtual characters and crowds, and the immersive visual representation requirements of virtual worlds in emergency scenarios.

Due to the wide multi-disciplinary focus of this research, this Literature Review will cover topics in psychology, sociology, mathematics, physics and computer science. Psychology and sociology topics covered in this chapter are the effects of fire smoke on humans, crowd dynamics in evacuation and emergency settings, and the interaction and impact on crowds through different roles. Mathematics and physics topics covered are the human behaviour models, and computational fluid and fire dynamics. Computer science topics are the computer model simulations, virtual training environments (including serious games), virtual visualisation tools of pedestrian behaviour models, game engines, computer network engineering and hardware accelerated graphics engineering.

To determine the benefits of the IRIS system (as introduced in Chapter 1, Section1.3), it is important to establish the current state of the art related to human behaviour understanding, which is the foundation for any sociological, psychological and mathematical model based on human behaviour research. It provides insight into how people are expected to respond in a fire scenario and therefore what modelling functionality is required to represent this response. To establish the human behavioural requirements of the IRIS system it is important to understand pedestrian decision-making in emergency situations or those that provide insights into emergency situations.

2.1 Evacuation and pedestrian dynamics

Evacuation and pedestrian dynamics are complex, as will be discussed in this section. Previous human behaviour theories for individuals as well as crowd behaviour are reviewed from the literature. It is important to note that the need for evacuation does not necessarily mean a presence of fire and/or smoke. The human behaviour in fire is more specifically reviewed in Section 2.2. This section will also highlight how different scenarios affect the complexity of the evacuation and pedestrian dynamics in emergency situations, and the roles and impacts of authority in an emergency in terms of crowd management.

This section seeks to answer the following questions:

- What is the current understanding of pedestrian and evacuation dynamics?
- How are individual roles presented in a current state-of-the-art social psychological understanding of evacuation and pedestrian dynamics?
- How do disasters pose new difficulties in different environmental settings in terms of emergency evacuation?

2.1.1 Ingress, circulation and egress (ICE)

To realistically simulate people's interaction and the impact of emergency procedures, it is necessary to model the ingress, circulation and egress phases of the building as identified by Boswell and Gwynne [61]. Boswell and Gwynne [61] proposed that in terms of people movement the life cycle of a building can be described with a system denoted as ICE, which stands for **I**ngress-phase, **C**irculation-phase and **E**gress-phase of people's movement within a structure. ICE essentially defines that people who enter a building will eventually have to leave the building, but while inside the building they will circulate around the building while performing various tasks.

It is important to establish that the people's movement and behaviour in an emergency within the building is impacted by an occupant's historical knowledge and memory of the past events and activities within the building, such as the occupant's route selection and response to emergency procedures. During an egress phase the occupant therefore utilises the historical knowledge gathered during the ingress and circulation phases and combines this with information from past events and the current new events (i.e. procedures, security restrictions, and other current event conditions) [62].

The ICE system will be used extensively in Chapter 6 with an airport case study to simulate the lifecycle of the airport building, where agents enter, circulate and leave the structure.

2.1.2 The understanding of crowd behaviour

To realistically interact as an authority with virtual computer-controlled and simulated agents in a virtual environment, it is necessary to understand the importance of crowd management as is identified by Still in his research (as will be discussed in this section). In his thesis Still [14] highlights the struggles of modelling crowds in a physical science approach. This section will also look into the current understanding on the effects of roles and authority, as well as the complex dynamics of representing these in crowd dynamics.

Still [14] explores in his PhD thesis crowd behaviour in Wembley Stadium and the surrounding areas during organised events (public assemblies) at the stadium. He makes several observations, such as how crowd behaviour in confined spaces, such as in corridors, stairways, train platforms, at ticket gates/turnstiles and in emergency egress, differs from the observations in Fruin's research [63] in a pedestrian street environment.

Still [14] concludes that generally crowds will utilise any available space on a *least effort* basis when attempting to get to a desired location. Depending on the motivation and the drive, the crowds will pack into tighter spaces, which eventually if unmanaged by an authority will end up crushing by severely injuring or killing individuals. This behaviour has been seen in a few different past events including the Love Parade in Germany in 2010 [13] and the Hillsborough disaster in Britain in 1989 [14]. Fruin proposed a Level of Service (LoS) [63] where a person occupies a certain space and the available space around the person defines his comfort level and affects his mobility in such a way that a person needs certain amount of space around him to move. Still contradicts Fruin by observing how crowds can still move even when heavily congested in high densities (i.e less than 0.46 square metres of available space around a single pedestrian) [14].

Still [14] summarises his study of crowd behaviour around Wembley stadium into the following 4 interactive rules:

- Objective - Try to move to a desired or intended end point.
- Motility - Try to maintain your optimum velocity.
- Constraint - Try to maintain a minimum distance between yourself and the other objects in the environment.
- Assimilation - Delay time taken to read and react to the environment.

Interestingly, Still [14] concludes that it was necessary to include noise (variation) into the above 4 rules to make the crowd appear realistic, whereas without the noise the crowd behaviour using the 4 rules was too regular compared to reality.

Still [14] also writes: *"In the Hillsborough example we can see that there was no commu-*

nication between entities, in that the people who were pushing into the tunnel were unaware of those people who were being crushed at the perimeter fence. It is important to note that the collective intelligence of a crowd is, in this way, similar to that of an ant colony. The stadium staff, police and, via the television, millions of people watching the match, were aware of the event, but the people at the back were not.". This highlights the importance of crowd management by the officials to combat the *"ant colony"* behaviour.

Still [14] discusses in his PhD thesis the importance of police in the crowd management in various scenarios: *"The crowd is held clear off the road by police horses. When the coaches pass into the tunnel the horses move aside and the two crowds, to the left and right of the tunnel, flow past by each other."*, *"The narrow pavement and lack of crowd management measures, temporary barriers, or police monitoring, mean that queuing supporters spill over the pavement and onto the busy road."*, *"In an attempt to relieve the pressure at the turnstiles, the police opened the exit gates allowing 2,000 additional fans into the ground and towards pens 3 and 4 which were already full."* [14]. Still also demonstrates[1] through examples the importance and the effect of crowd management by modelling pre-programmed crowd redirection behaviour using an authority figure (i.e. police) issuing *Evacuate in direction* instructions to the crowd with 100% crowd compliance factor.

Reviewing Still's research has shown the importance of crowd management, essentially when used correctly (i.e. by security or police) preventing large crowds from crushing people to death or to a serious injury.

While Still focused his research on crowd behaviour in tight spaces by modelling it as if it was an *"ant colony"* (i.e. the physical science approach [64]), Chu expands on the complexity of crowd behaviour in his PhD thesis in 2015 [65] by looking into social theories that are developed through empirical studies of past accidents. Chu summarises an individual's decision-making process in egress situations into three theories: *bounded rationality*, *reward-maximising* and *place scripts*.

The *place scripts* theory compliments the ICE behaviour theory in the Section 2.1.1, based on Tong and Canter's research [64]. Tong and Canter [64] highlight the importance of roles, prior training and previous experience. The previous experience comes from the expe-

[1]The demonstration can be found on page 241 in Still's PhD thesis [14].

rienced past incidents and events, as well as knowledge of the building/structure. Tong and Canter [64] explain that prior training and previous experience work against an occupant's motivation to evacuate. Such as if the occupant is aware of an available exit route and has prior training dealing with the currently imposed situation, while the situation is perceived attainable (i.e. fighting a fire), the occupant feels less threatened by the situation and is less likely to start evacuating. Tong and Canter [64] found that in the Beverly Hills Supper Club Fire in 1977, the waitresses who served the guests used the previously attained relationship to guide guests to the exits, indicating the role the waitresses had over the guests prior to egress retained.

Chu summarises [65] *bounded rationality* as behaviour where people find a *good-enough* solution from a set of alternatives to a problem limited by perceived information and heuristics and then tend not to deviate as long the solution stays viable. Chu summarises [65] *reward-maximising* as behaviour where people are motivated and shaped by selfishly maximising their own individual reward by calculating the surrounding elements, such as behaviour and reaction of a crowd. Out of the three *individual behaviour* theories that Chu explored in his PhD thesis [65] the *place scripts* theory gives the most insight into roles and role-based behaviour that individuals assume or follow.

In his thesis Chu also looks into actual crowd behaviour theories: Le Bon's *Group mind theory* [66], McPhail's *Predispositionist theory* [67], Aguirre's et al. *Normative theory* [68], Sime's *Affiliative theory* [69], Tafjel and Turner's *Social identity* [70].

Le Bon's *Group mind theory* [66] is from his book from 1895, where the English version became available the next year in 1896. Le Bon focused his analysis on the nature of crowds in political movements. His book does not give insight into crowd movement behaviours, but Chu draws the following conclusions from the book: people lose all sense of self-responsibility by becoming part of the crowd with anonymity and share the group's emotions, which makes individuals feel, think and act differently than when they are in isolation. This shows that crowd behaviour theories are an old research topic and highlights how the modern understanding of crowds through continued research and studies have superseded this early way of thinking [71].

McPhail's *Predispositionist theory* [67] is from his article in Sociological Quarterly in 1989. McPhail raised a question: *"If conscious and rational individuals are in control of their own behaviour most of the time, pursuing their diverse interests and engaging in a diversity of behaviours, when and how and why do they begin to act in common or in concert?"*[2]. McPhail criticises Le Bon's (and Park's and Blumer's) previous work on the *transformation hypothesis*, which Chu calls *Group mind theory*, claiming in 1989 that it is without logical or empirical foundation. McPhail then explains with examples how individuals within crowds understand that they are part of a crowd by making a conscious assessment of their anonymity and *"safety in numbers"* [67]. McPhail expands on his theory and suggests that instead of *"collective hypnosis"* of the crowd, individuals become part of a crowd through communication with other crowd members and control their behaviour by means of self-instruction for behavioural adjustment in relation to the goals or objectives they are pursuing (i.e. more through social alignment than contagion). McPhail suggests that the goals or objectives are communicated and suggested by others [67], but individuals remain in full control of their own behaviour.

Aguirre's et al. *Normative theory* [68] is from their study on the 2003 fire at the Station nightclub in Rhode Island to retest the *normative model* associated with Turner and Killian [72] and Feinberg and Johnson [73]. Aguirre et al. [68] discuss Feinberg and Johnson's previous study on the Beverly Hills Supper Club fire in 1977, where Feinberg and Johnson discovered that people have social norms, values and affiliations which made people return into burning buildings to search for dear ones, and staff often performed heroic and selfless acts trying to save their clients (see *place scripts* theory from Tong and Canter [64] earlier in this section). Aguirre et al. propose Turner and Killian came into similar conclusions and points out: *"Self-categorization, like all normative theories, relies on the assumption of "thinking subjects" who are members of social groups, who feel collectively threatened or gratified by some incident, and act on the basis of this shared categorization."* [68]. Aguirre et al. conclude that the study on the Station's fire shows the individuals who died were not competing with each other for egress from the building, but instead they were helping each other until the very end. Aguirre et al. point out that the survival of the survivors relied on the survivors helping each other, highlighting the support for the normative model.

[2]The quotation is from page 412 from McPhail's article [67].

Sime's *Affiliative theory* [69] is from his case study in 1985 into people's affiliation with each other and their current structure (or place) and the impact of these affiliations into the cognitive decision of the escape route during egress. Sime aimed to perform this experiment without physical presence of smoke or flames, but only the assumption of threat, while recording people's decision to move towards or away from the threat based on their familiarity with each other and the building. Sime [69] suggests that in an emergency people are drawn more towards familiar places, routes and people than under normal circumstances (i.e. the *affiliative theory*). Interestingly Sime also included members of staff (an authority) in the case study. Sime discovered [69] that when staff guided the people towards a fire exit, the people who were close to the staff were more likely to choose the instructed, but unfamiliar fire exit route, whereas those who were further away and unlikely to hear or notice the instructions from the staff were more likely to choose a familiar (known) exit. Sime also discovered [69] that the staff who did not instruct the occupants or stopped interacting with the occupants lost (or never had) any influence over the occupants' behaviour.

Tafjel and Turner's *Social identity* [70] theory is from their article in the *Psychology of intergroup relations* journal in 1986. In the article Tafjel and Turner discuss intergroup and in-group behaviours mainly in terms of competition and conflict. Tafjel and Turner suggest that for individuals to associate themselves as a group, they must internalise their group membership individually as an aspect of self-concept and it is not enough for others to define them as a group.

The research by Chu identified in his PhD thesis [65] reviewed known and proposed individual and group behaviour theories. It is clear from the above reviews that there does not exist a single unifying crowd behaviour theory. It is interesting to point out the discovered influence of authority (and role) in many of the researches, such as Sime [69], Aguirre et al. [68], Tong and Canter [64]: a person with an authoritative role (i.e. a member of staff or police) has high influence over a group's overall behaviour in following instructions in emergency situations near the vicinity of the person, but those who could not hear or see him were likely to choose the previously known (familiar) route. Uniformally across the above researched group behaviour theories it is clear that group members copy behaviour from fellow in-group members progressively either through communication or observation.

The research reviewed in this section highlighted the role and the influence of an authority on a crowd of people, and how affecting an individual from the crowd can have a propagating effect on the rest of the crowd, as other members of the crowd will be observing and reacting to the behaviour of the affected individual. This is important in the implementation of the interaction capabilities between the user-controlled agents and the computer-controlled agents for the IRIS system.

2.1.3 Effects of Authority

When looking at crowd interaction, the important factor of this is looking at the authority figures and their impact on crowds. In the London Bridge stabbing incident on 4^{th} of June 2017 [74, 75], the effects of the authoritative role of police officers influencing the crowd behaviour to *"get down"* can be seen from the recorded video footage. The police officers repeatedly shouted, *"Get Down!"* to the crowd while performing a hand gesture. Some of the crowd reacted immediately, whereas the rest of the crowd followed eventually. The police kept re-instructing the same commands repeatedly to reach full compliance from the crowd.

In an article in July 2018 from the Fire Technology journal, Chu and Law [76] revise the currently known understanding of group behaviour. In the article, Chu and Law suggest there are three individual behaviour levels that a person considers when making cognitive decisions about evacuation behaviour: *Individual Level*, *Group Level* and *Crowd Level*.

The *Individual Level* considers the following factors: physical profile, *place script* (see Section 2.1.2 above) and cue awareness factors. The *Group Level* considers the following factors: social group identification (see Section 2.1.2 above), group compliance, group influence and group separation tolerance. The *Crowd Level* considers the following factors: crowd compliance, social order and crowd following time lag. From the three distinctively identified levels, some interesting attributes stand out, such as *crowd following time lag*, which Chu and Law [76] describe as an "example of influence" to indicate the time needed for the person to follow decisions of the surrounding crowd.

Even though in the majority of the cases people eventually follow instructions from an authority, there are exceptions, such as a group of 4 people in the London Underground

fire in King's Cross St. Pancras tube station in November 1987 as identified by Donald and Canter [77]. The reasons for this group disobeying the instructions are unknown, but Donald and Canter [77] speculate that this was due to the group either not noticing the instructions or they may have decided to take the most direct (straight) route to the exit from the escalator as they were almost already outside. Nevertheless, Donald and Canter concluded: *"This group lacked identifiable personal reasons for choosing any particular exit."*.

2.1.4 Evacuation and pedestrian dynamics summary

This section has highlighted some of the complexities of modelling human behaviour in emergency evacuation scenarios and the necessity of modelling these in virtual environments. The highlights have shown that crowds have complex behaviours and potentially devastating results if unmanaged. The crowd dynamics depend on many factors, such as the assumed role of the authority, proximity to the authority, previous interactions with the authority and the influence from the surrounding individuals. The influence from peers within a crowd increases the likelihood of the individual following a command once one of the peers begins to follow the said command from an authority. Crowd of people are not likely to obey immediately a given command from an authority, but will do so incrementally, until the whole crowd is eventually following the command. Issuing repeated commands as an authority is often necessary to achieve full compliance. It is necessary to represent these complex crowd behaviours when a user-controlled agent interacts with a crowd and how people respond to commands given by the user-controlled agent representing an authority.

Kramer made an interesting statement in his book: *"If the human brain were simple enough for us to understand, we would be too simple to understand it."* [78].

2.2 Human behaviour in fire smoke

To give the user of the IRIS system an immersive experience in fire emergency scenarios, it is important to accurately subject the virtual user-controlled agents to the effects of fire (i.e. smoke, heat and toxic products). Some of these effects are invisible to the naked eye

and therefore cannot be naturally visualised on their own. Nevertheless, the effects of the invisible products can be visualised and simulated on the virtual agents (i.e. the user may notice that a computer-controlled agent may suddenly collapse or start crawling within the visibility range of the user-controlled agent). Proulx and Fahy described in 2008 that even a tiny dosage of toxic products, such as a couple of breaths can kill [30]. Therefore, it may not be necessary to subject the virtual agents to toxic products for prolonged periods and the incapacitation of the user-controlled agent may come suddenly to the user. Fortunately, this will happen only in the virtual world and the user will learn a valuable and potentially lifesaving lesson.

One of the linked software components used in the research presented in this thesis, namely buildingEXODUS, comes with a hazard and toxicity model [79], which enables the simulation of the effects of fire and the fire produced products on the virtual population including the user-controlled agents. The toxicity model as described by Purser [80] in SFPE Handbook of Fire Protection Engineering in 2016 uses Fractional Effective Concentration or Dose (FEC or FED) for toxic hazard calculations. The FED models are typically limited to consideration of small key toxic products (such as CO, HCN and acid gases) even though fire effluents are known to contain many hundreds of toxic substances [80]. The production of toxic substances and other important effects of fire and smoke can be simulated by a CFD fire simulation software program, such as SMARTFIRE [81, 82] or FDS [83, 84].

Visibility in fire smoke can have a huge impact on the evacuation velocity and therefore on the total evacuation time by negatively affecting wayfinding [85]. It is therefore important to correctly visualise the loss of visibility in smoke on the user-controlled agent in an immersive fire emergency scenario. This is discussed in more detail in the Section 2.2.1.

Kuligowski [71] summarises: *"Human behavior in fire is at the core of all life safety projects completed by fire safety or fire protection engineers."*.

2.2.1 Visibility in fire smoke

Getting lost in smoke in a real fire emergency can have devastating results, not only to the individual, but to the families around the person as was seen in the Grenfell Tower fire in

2017 [86]. Therefore, it is paramount to be able to accurately represent visibility in smoke in virtual environments in order to enable training and testing of fire scenarios and procedures before they happen. To enable the IRIS system to represent the visibility correctly in real-time, it is necessary to establish what is currently understood on the topic of visibility in fire smoke, in published research.

Current understanding of visibility and human behaviour in fire smoke is summarised in the SFPE handbook of Fire Protection Engineering [21]. The visibility work is based on Jin's research and experiments in Japan in 1970 [18]. Jin was the author of the chapter *"Visibility and Human Behaviour in Fire Smoke"* in the SFPE handbook in edition 3 (in 2002) and edition 4 (in 2008), but the chapter received new authors: Yamada and Akizuki in the latest edition (edition 5) in 2016. Nonetheless, the smoke visibility research in the 2016 edition (the latest edition at the point of writing) is still based on the same research by Jin in 1970. The *"Visibility and Human Behaviour in Fire Smoke"* chapter in the SFPE handbook gives insight for CFD engines and human behaviour models on how to correctly model the visibility and human behaviour in fire smoke, of which the visibility side is of particular interest to the research presented in this thesis.

The visibility calculations proposed in SFPE handbook [21], based on Jin's research [18, 19, 87], are reduced to a light extinction coefficient variable. The visibility of an object is dependent on the relationship of the light extinction coefficient of the smoke and the brightness of the object [21]. There are other factors that affect whether a person can see the object, such as visual acuity which depends on the person's eye sight, age, and irritation levels of the smoke (i.e. a correlation of eye blink rate to visual acuity) [21, 87].

In 2005 [25] and 2007 [24] Kang published research into visualising the smoke visibility in VR. Kang proposed an expensive ray tracing algorithm that calculates pixel by pixel visibility of smoke from a CFD simulated smoke domain. Kang used a Fire Dynamics Simulator (FDS) program to model the spread of smoke and fire in an underground mass transit station [24]. In the visualisation of the smoke visibility Kang used perfect-white coloured surfaces in the virtual environment with wire frames guiding the geometric shapes. The light extinction coefficient was modelled to produce black coloured smoke, covering the wireframes and the perfect-white surfaces in Kang's demonstration. Kang also explained that to render

his expensive ray-tracing algorithm in real-time, the smoke visibility data must be pre-calculated into predefined bounding volumes for a fixed camera location and orientation and only then can the ray-tracing algorithm be sufficiently fast to compute for the single fixed camera location and orientation [24]. This limitation is not different from producing a video file by pre-calculating the playback of smoke visibility data. In the video file the raster would be pre-calculated and encoded frame by frame from also a fixed camera location and orientation, similarly to Kang's proposed real-time rendering [24]. Video files can also be rendered in real-time and it imposes the same limitations and requirements to Kang's ray tracing algorithm. It is safe to say the ray-tracing algorithm is unsuitable for real-time visualisation in an interactive virtual environment.

In a recently (May 2018) published article [88] in the International Fire Professional journal, Conway reviewed recent developments in smoke visibility research. The current understanding of visibility in smoke is based on Jin's research [88]. Conway concludes [88] that smoke particles can affect the measurement of the light extinction coefficient, such as the size and shape of smoke particles affect light scattering, while some smoke particles are smaller than the visible light wavelength. Conway also notes [88] that smoke is known to age which happens by the merging of smaller smoke particles resulting into "thicker" smoke. Conway [88] suggests that light wavelength may have an importance in the calculation of the light extinction coefficient in two ways: the way humans perceive it (i.e. human eye has unequal sensitivity to colour) and how well the light may travel or scatter through smoke.

Vigne and Węgrzyński [89] published in 2017, results of a full-scale experiment of burning various materials (methyl alcohol, propane alcohol, heptane and toluene) in a controlled test environment measuring the smoke visibility relationship to soot yield. Vigne and Węgrzyński [89] also ran computer simulations to compare the live experiment results to the results of a CFD model, Fire Dynamics Simulator (FDS). Vigne and Węgrzyński [89] suggest that there is a close correlation between soot yield and the light extinction coefficient in Jin's research, further validating Jin's experiments from 1970.

In 2014 Xu et al. [6] published research into real-time fire smoke rendering in a VR training environment. They proposed a volumetric voxel rendering technique of simplified and pre-processed FDS CFD data, based on soot density, a similar technique was already

used by Cha, Han, Lee and Choi in 2012 [7]. The opacity of the smoke was proposed as normalised soot density values based on the maximum soot density found on the current time step of the simulation[3]. See the Section 2.2.2 for discussion on the use this kind of normalised soot density to represent smoke visibility. Additionally, Xu et al. [6] used a voxed based volumetric rendering technique of a pre-processed 3D grid of voxels. While it is more efficient than Kang's ray tracing algorithm [24] (see the review earlier in this section), the rendering of the volumetric voxel smoke was limited to around 10,000 voxel cells rendered in real-time (60 frames per second on the tested hardware). Xu et al. [6] proposed a number of optimisations for the volumetric voxel rendering technique to reach the stated performance level, such as simplifying and reducing the accuracy of the FDS CFD data as well as removing transparent voxels during the pre-processing step. However, the voxel based volumetric rendering is inefficient as a voxel has 6 faces (quads) in comparison to a billboard render of only a single face (quad) that is proposed in Chapter 4. Improved, more efficient, and actually accurate billboard-based smoke visibility rendering system is proposed in Chapter 4.

In 2014 Meng and Zhang [15] experimented with wayfinding in fire smoke scenarios in a virtual environment, but by using a real smoke generator outside of the virtual world. The aim of the study was to induce stress on the participants through immersion and measure how close the participants perform in relationship to real fire scenarios. While the virtual environment produced by Meng and Zhang was not able to accurately represent visibility in smoke, the research concluded that the fidelity of the virtual environment plays a key factor in stressing and immersing the user, which in turn makes the user's behaviour match more closely to the behaviour observed in real scenarios [15].

This section has provided the basis for the implementation of accurate real-time smoke visibility in a virtual environment, which is covered in detail with a proposed validation case and a demonstration in in Chapter 4. The smoke is further used and demonstrated in a case study with external participants in Chapter 6.

[3]The description can be found on Section 2 on page 2 and Fig. 1 on page 3 [6].

2.2.2 Normalised soot density as an indication of visibility in smoke

During the review of literature on current state of the art modelling of visibility in smoke, a few research articles in visualising FDS based CFD simulations stated the following assumption: normalised soot density equals smoke visibility [1,4,6,7]. St Julien and Shaw [4] claimed accurate smoke visibility representation by normalising soot density to a ramping function to map smoke visibility from 10% opacity to up to 75% opacity. St Julien and Shaw [4] provide no further explanation to the accuracy of the ramping function than it is *accurate* and *appropriate*. Xu et al. [6], and Cha et al. [7] claimed that normalised soot density against the highest soot density value in the current simulation is an accurate representation of visibility in smoke. The inherent problem with normalising the values in a CFD simulation to the maximum value produced in the current simulation run is that if e.g. a CFD simulation was done on a burning matchstick and another CFD simulation was done on a burning building, the normalised soot density of the matchstick fire smoke and the house fire smoke would produce equal smoke visibilities at the highest soot density.

Rubini et al. [90,91] proposed how soot density can be used to visualise smoke visibility, by calculating a light extinction coefficient from the mass specific extinction coefficient in relation to the mass of soot in a unit volume of smoke. Rubini's et al. [90,91] proposed solution assumes a spherical shape of soot and treats them as *"optical lenses"* with dimensionless extinction efficiency (i.e. scattering). The proposed solution by Rubini et al. [90, 91] does not take into account how the smoke particles change over time due to *"aging"* as was highlighted by Conway [88] (see Section 2.2.1). Weingartner et al. [92] published research results of measuring light extinction coefficient of soot particles. Therefore, using relevant research [88, 90–92], it is possible to extract light extinction coefficient from soot density values produced by FDS or other CFD model. The light extinction coefficient can then be used with the proposed solution in Chapter 4 to visualise smoke visibility.

2.2.3 Human behaviour in fire smoke summary

This section highlighted the impacts of the effects of fire (i.e. heat, smoke, toxic products) on people and how it is necessary for the IRIS system to simulate and visualise these effects

on virtual agents. Meng and Zhang's [15] research showed that the higher fidelity of virtual environments is necessary to reach more realistic behaviour from the user in emergency scenarios, while pointing out that the user behaves differently in non-emergency scenarios than he does in emergency scenarios. Furthermore, the visibility in smoke has not been accurately visualised in real-time virtual environments even though this is a crucial part of the immersion in VR to produce realistic behavioural results from the user. This thesis proposes a solution in Chapter 4 to this.

2.3 Virtual training environments (VTE)

In comparison to *evacuation and pedestrian dynamics* and *human behaviour in fire* research (since early 1900s as identified by Bryan [85]), research into virtual training environments (VTE) is a relatively new field. In 1993 Kozak et al. [93] set an experiment to test the suitability of VR hardware and software as a training tool. Kozak et al. came to a conclusion that VR environments were not yet suitable for training as their experiment showed no improvement of the trainees who trained using VR environment over those who did not have any prior training, while those who trained without the VR showed significant improvement [93]. In the same paper Kozak et al. also pointed out that this was mainly due to the current state of the VR hardware and software, in terms of computer graphics quality and performance, and the motion tracking quality and performance of the hardware, suggesting the need for substantial improvements in the VR environments before they are suitable for training [93].

This section seeks answers for the following questions:

- What virtual training environments exist and are currently used or proposed for training emergency first responders such as law enforcement officers and firefighters?
- What are the current state of the art interaction models with virtual agents in a simulated virtual training environment
- What current limitations exist in the virtual training environments for first responders that the IRIS system can address?

2.3.1 Full-scale training

Full-scale training is expensive, time consuming and requires substantial effort to organise. To demonstrate the cost of full-scale training and make the case for the need for Virtual Training Environments for the financial purposes alone: the 11 days long Arrow 17 International Mechanised Military Exercise organised by Finnish Defence Forces in 2017 [94–96] can serve as an example.

Arrow 17 involved thousands of military troops, hundreds of vehicles, including tanks, artilleries, armoured infantry vehicles and support trucks, as well as aircraft ranging from helicopters to fighter jets. The large-scale exercise required countries such as US and Norway to bring their military vehicles, troops and equipment to the exercise in Finland. Arrow 17 involved the consumption of expensive ammunition, explosives and fuel as well as the maintenance costs of personnel and equipment.

While full scale training exercises can teach invaluable lessons, including subjecting the trainee to elements that are not possible through a virtual training environment, the exercise itself can be so costly that it is difficult to organise the exercise frequently or even repeat if the outcome was undesirable. While Finnish Defence Forces manages to organise the Arrow International Mechanised Military Exercise annually, it clearly requires a military level budget and organisational skills.

2.3.2 Serious games (SG)

Serious Games (SG) are a subset of Virtual Training Environments (VTE) in a way that they propose a VR training environment using computer games and computer game technology [1, 2, 9, 97, 98]. Williams-Bell et al. [1] describe serious games as *"With the ubiquity of video game play and advent of new consumer-level physical interfaces for video games (e.g., the Nintendo Wii Fit balance-board and the Microsoft Kinect), serious games (games whose primary purpose is education and training), are able to provide users with innovative interactive techniques that are highly engaging and immersive."*. Thus, serious games are treated as VTEs in this thesis.

BinSubaih et al. [2] suggest that playing video games can improve hand-eye coordination, visual acuity, learning procedures and recalling experiential details, analytical and spatial skills (such as visualisation), strategic thinking, insight and better reaction time.

BinSubaih et al. [2] describe challenges faced in adopting serious games as a suitable training tool. They write that the first problem is the presumption of games to not be suitable for learning, while arguing that games can provide a more interesting and easier approach to learning on topics that learners do not want or even resist to learn because it is boring or complicated. While other challenges being: to prove learning effectiveness, assessment challenges of abstract skills (i.e. teamwork and leadership), and the development challenges such as building the actual game (i.e. selecting a game engine, adding artwork, audio, and dialogue systems) [2]. BinSubaih et al. [2] describe that *edutainment* is badly received and suffers a lack of respect, which poses adoption problems to *serious games* as they are often linked together. Williams-Bell et al. [1] point out that serious games tend to lack means of testing and progress-tracking that are typically expected of education and training tools, adding that the assessment of the learner is not trivial and greater work is required.

BinSubaih, et al. [2] conclude that one of the important reasons for using serious games for learning is that they are fun, while highlighting that 87% of the most frequent computer and video game players play video games because they are fun. Smith and Ericson [99] found that teaching children fire-safety skills in *"game-based VR training systems did improve the children's satisfaction with VR fire-safety training and their overall motivation for fire-safety training"*.

Sacfung et al. [100] developed a SG for an evacuation exercise where participants needed to evacuate from a University building. Sacfung et al. concluded that *"there is very high correlation coefficient between the in-game evacuation time and the in-real evacuation time"*. The participants used keyboard and mouse to control the agents. However, the participants had to evacuate alone and there were no other agents present in the test to influence the participant's decisions or to interact with. Fire and smoke were also absent from the tests.

Feng et al. [101] reviewed multiple papers on immersive VR environments for learning

evacuation knowledge and behaviour assessment which are based on SGs. Feng et al. writes *"absorbing gaming environment is able to induce participants to react as close as to the reactions in real life"*. The review claims that immersive VR SGs are a promising tool to analyse the behaviour of participants for different purposes, such as analysing human behaviour during emergencies. The review also states immersive VR and SGs help participants to retain knowledge longer from the full engagement and higher emotional and physiological arousal than traditional training methods.

Heldal et al. [102] performed 33 interviews in Europe (Sweden, Estonia, the Netherlands, Norway, the United Kingdom and Denmark) and Singapore, asking researchers, developers, managers and instructors on the use of Simulations and Serious Games (SSG) for training. Heldal et al. discovered from the interviews some of the common hindrances in the adoption of SSGs (i.e. VTEs) for training, such as: *"a sceptical attitude towards the technology, a mismatch of expectations, or insufficient competence affecting the willingness to incorporate the new technology into the teaching practice."*. VTEs are also considered to be too complicated and too costly to set up and use. Few of the instructors had limited or no prior experience with VTEs and most instructors *"did not consider scenario development as part of their job, due to lack of skills and time."* However, Heldal et al. also discovered that some countries (Estonia, the Netherlands and UK) have experienced higher effectiveness for training fire fighters when using VTEs. The research concluded that instructors play a vital role in training even when using VTEs, and therefore, it is crucial that instructors are involved and trained from the beginning when introducing VTEs to an organisation.

2.3.3 VTEs in law enforcement

The main targeted application area for the IRIS system proposed in this thesis is first responders to emergency scenarios (including law enforcement officers). Therefore, enabling the IRIS system to be used for training and testing of procedures in a virtual environment with interactive crowd management capabilities and fire hazards.

BinSubaih et al. [2] wrote that computer games are used extensively for training in military domain but are lagging behind in police domain. Hormann [103] wrote on an FBI

Law Enforcement Bulletin about the need to include VR environments for law enforcement training, highlighting their common use in military exercises. Bos [104] also described the limited use of VR environments in law enforcement training, while military, medical industry, airline industry, education system, space program, architecture, ergonomic studies, assistance for the handicapped, and study and treatment of phobias enjoy VR technology training.

Bos [104] highlighted the desperate need for VTEs in law enforcement describing that law enforcement officers may go through an entire twenty-year career and never be exposed to certain critical situations of which they may not have chance to learn from. Bos [104] describes the lack VR training environments for law enforcement is regularly down to law enforcement budgets. BinSubaih et al. [2] also highlight the typical low budgets for law enforcement in comparison to military being a reason for the limited use of virtual training environments in the law enforcement domain. BinSubaih et al. [2] conclude that some of the only areas where virtual training environments are used in law enforcement are in training areas that cross over with military training, such as the simulation for training range shooting. BinSubaih et al. [2] proceeded to propose a virtual traffic accident training environment for law enforcement.

2.3.4 VTEs for learning spatial knowledge

VTEs can be used to learn spatial knowledge and test the impact of fire and smoke in the learning of spatial knowledge, wayfinding and navigation. As was discussed in Section 2.2, wayfinding can have a huge impact on the survival in fire emergency situations. VTEs can be used to learn spatial knowledge and practise procedures of public buildings (such as airports), which are difficult to shut down for training, and evaluate the most effective routes to deal with emergencies including redirecting of crowds. Looking into the past research studies can help identify what information is necessary to be presented to the user of the IRIS system to produce natural wayfinding experience and behaviour.

Waller et al. [105] performed a set of experiments to determine the suitability of VR environments in 1998 for learning spatial knowledge of a real-world environment. In the experiment the subjects had to navigate a maze and get to a set of numbered locations. The test subjects had different levels of exposure conditions: no exposure to the maze, real

exposure by allowing the subject to wander around the maze prior to the test, exposure by looking at a map prior to the test, or different levels of VR exposure. However, the VR exposure only presented naturally available information, such as it did not allow subjects to fly above the maze, see through walls or have any artificial assistance such as providing floating arrows in the air. Waller et al. [105] concluded that the VR training had proved to be as good or better than the real-world training of allowing the subject to navigate freely around the maze. Waller et al. [105] also pointed out the importance of allowing the subject to get immersed into the VR environment, such as when the subject was using the VR environment for only 2 minutes, this was comparable to the exposure through a map, but when the subject was using the VR environment for 5 minutes or longer, this was as good or better than the real world exposure. Waller et al. [105] highlighted the importance of a wide field of view of the virtual environment to the subject to allow better immersion.

Also, in 1998 Darken and Banker [106] did an experimental study into the suitability of using VR environments as training tools for learning spatial knowledge. However, the study by Darken and Banker [106] focuses on outdoor environments with different experience level test subjects: *beginner* level navigators understood the concept of orienteering but had very little experience in orienteering, *advanced* level navigators were experts in orienteering by participating in orienteering exercises monthly, where *intermediate* level navigators were everyone between the beginner and advanced level navigators. Darken and Banker [106] discovered that the VR training environment helped the intermediate level navigators the most as they were able to relate many of the navigational cues in the VR to real-world. The advanced navigators benefited the least from the VR environment, due to their highly developed expertise on terrain visualisation from maps, and VR environment did not provide anything new than what the advanced navigators had already mastered. The beginner navigators had difficulties relating the cues provided in the virtual environment to a real world, due to the limited capabilities of the VR environment and thus the unavoidable inaccuracies in representing the real world in the virtual world in this experiment in 1998. The research by Darken and Banker [106] demonstrates the importance of matching the virtual training environment to the target audience (i.e. in this case the intermediate navigators), but also highlighting that virtual training environments are not a magic bullet allowing further training those who have already mastered the training objectives (i.e. the advanced navigators in

this case). Darken and Banker [106] found that the virtual environment allowed the subjects to be trained faster than a real-world training would, due to the ability to move artificially fast with limited exhaustion in a vast training environment.

2.3.5 VTEs for firefighting

To establish use cases for the IRIS system in firefighter training, it is necessary to review the current firefighter VTEs with published research. This section will look for current state and the use of pedestrian dynamics and fire dynamics in firefighter VTEs.

Williams-Bell et al. [1] performed a review on 26 different video games, serious games and virtual simulations currently used or proposed as training tools in the fire service. 12 of the 26 reviewed tools were designed for only training, 7 were designed for education, 5 were designed for entertainment and 2 were designed for both education and training [1]. To clarify the difference between education and training in this context: education category means the tool is meant to educate the user on its intended topic, whereas training category refers the tool is meant to be used to learn new skills or improve on existing skills. Williams-Bell et al. [1] discovered that from the training tools only 5 out of 26 tools incorporate any fire dynamics and 10 out of 26 tools incorporate any people dynamics of which only 3 tools incorporate people dynamics beyond the representation of the user.

Bukowski and Séquin [3] proposed a virtual training environment for the study of fire hazards and their environmental effects in buildings. The VTE based their VR training tool fire dynamics on CFAST fire simulation. The purpose of the smoke visualisation in this training tool was to provide the user an indication of a pre-defined smoke layer height from the ground. Bukowski and Séquin's [3] proposed training tool does not include people dynamics.

St Julien and Shaw [4] developed a Firefighter Command Training Virtual Environment (FCTVE) training tool for firefighter command training. The fire dynamics of St Julien and Shaw's [4] training tool fire dynamics are based on FDS CFD simulations. However, the visualisation of the visibility of the smoke is based on an artistic representation of normalised soot density (see Section 2.2.2). The training tool features a pre-defined and pre-calculated

fire scenario based on a pre-defined made up house and the trainer is provided a pre-defined set of options to interact with the training tool and does not include any people dynamics.

Ren et al. [5] proposed a VR training tool in 2008 to evaluate emergency building evacuation. The research details a public train station with a visualised smoke hazard based on FDS CFD simulation, however, details such as how the visibility of the smoke is determined have been omitted. Ren, Chen, Luo [5] also demonstrate that the training tool can feature agents moving in preprogrammed paths defined by the trainer. Xu et al. [6] presented an updated version of the same training tool as Ren et al. [5]. The updated version demonstrates a smoke visibility visualisation based a normalised soot density against the highest soot density value of the current time step of the current simulation (see Section 2.2.2). The updated training tool by Xu et al. [6] demonstrates a primary school scenario with two pre-calculated evacuation paths which allows the trainer to show the trainee (i.e. firefighter) a safer path for rescue.

Williams-Bell et al. [1] reviewed Advanced Disaster Management Simulator (ADMS), a commercially available VR training solution for training emergency response commanders how to respond to terrorism, fire threats and other emergencies. Williams-Bell et al. [1] criticised it for the lack of immersion due to its simple, planar graphics technology. Cha et al. [7] wrote that ADMS does not provide fire experience for firefighters but is rather designed for learning command systems and firefighting plans, and therefore only features *"a simple particle system for fire simulation and graphic expression to ensure the visual recognition of fire accidents."* [7]. Cha et al. [7] proposed in 2012 an improved VR training tool over the ADMS for training firefighters. The improved system features a FDS CFD simulation for visualising smoke visibility, however, it is using a normalised soot density as visibility value for smoke (see Section 2.2.2) similarly to the proposed training tool by Xu et al. [6]. The improved training features a mission to approach a fire source while providing status information to the trainee of his current state, such as temperature, smoke and carbon monoxide levels. It allows the playback of pre-processed and simplified FDS CFD simulation data during training and records the trainee's exposure to temperature, smoke and carbon monoxide every second during the training scenario for later analysis allowing these to be summarised to the trainee at the end of the training scenario. The improved training tool does not include people dynamics [7].

Querrec et al. [8] developed a VR training tool for firefighters in 2004 called SecureVi (Security and Virtual Reality). It does not feature fire dynamics, but simulates gas and cloud spread (i.e. from a toxic gas leak) and has a collision model between gas and water wall. Nonetheless, it does feature a system called MASCARET (Multi Agents Systems to simulate Collaborative Adaptive and Realistic Environments for Training) as a plug-in framework for including a people dynamics model into SecureVi. The aim of MASCARET is to provide artificial intelligence to autonomous non-player agents. However, MASCARET is not proposed with a working human behaviour model but is provided as a plug-in system for the user to provide their own behaviour model that MASCARET will then utilise. The behaviour plug-in template for MASCARET is detailed in the paper by Querrec et al. [8]. The trainee will interact with MASCARET model and view this played back the SecureVi 3D virtual world environment built from VRML 3D models by the trainer. The interaction happens through a set of choices presented to the trainee by the non-player agents based on the programmed behaviour provided by the trainer. The real-time interaction with the system is only possible with a single non-player agent at a time [8].

Sharma et al. [107] published research on the development of a Unity3D game engine-based multi-user VR training environment for emergency scenarios with simulated crowds. However, the research presents several oversight and trivialisations, such as the computer-controlled agents trigger *"panic behaviour"* in the presence of smoke or fire and display *"panic body language"* (the body language is not explained). In 2008 Proulx and Fahy [30] explained that *panic* is a myth and is too rare to be of any importance in human behaviour in fire. Kuligowski reiterated this myth in SFPE Handbook of Fire Protection Engineering [71]. In 1982 Keating already wrote about *"The myth of panic"* [108]. The *"panic"* behaviour modelled in the research by Sharma et al. [107] is based on earlier research by Sharma et al. [109], where they write: *"A validation would require instrumenting and capturing data from real-life emergency situations where people actually panic."* [109] highlighting the lack of real-life *"panic"* and the inability to validate the *"panic"* behaviour. The training environment proposed by Sharma et al. [107] provides two types of interactivity: the user can control an agent to move around in the virtual world, and some computer-controlled agents are pre-programmed to work similar to magnets when the user moves his agent near them at which point they start following the user's agent unconditionally or without any interaction

from the user (the paper calls this: triggering a *"proximity sensor"* [107]). The user cannot interact with the simulated computer-controlled agents in any other way than to move in close proximity to trigger magnet-like behaviour. The VTE presents the computer-controlled magnetised *"proximity sensor"* [107] agents in uniformly bright blue colour to enable the user to easily distinguish them from the other uniformly brightly coloured agents, which the user cannot interact with. The training environment includes the representation of fire, nonetheless, this is purely for artistic purposes as it has no other effect in the simulation than to trigger the *"panic behaviour"* and is unable to harm anyone.

Rahouti et al. [9] also published research on a virtual training environment based on Unity3D game engine to train firefighting and evacuation. While the user can firefight with a fire extinguisher, the fire or smoke itself is not simulated or based on a simulation. Nonetheless, as part of scripted events, the user can get virtually hurt by the fire or smoke and toxic gas. The training environment itself comes with a set of three available pre-scripted levels where the user must make choices on what to do. The user is presented with a health bar for his own health as well as a progress bar for the fire and the extinguisher, which the user must keep an eye on to avoid death. At the end of the training event the user is presented a total *play time* (which is present while playing) and each missed goal during training adds penalty time. Rahouti et al. [9] explain that including CFD fire simulation would make the game closer to a real fire emergency as well as including a multi-player version would allow collaborative training.

Shen et al. [10] published a Unity3D game engine based prototype interactive virtual training environment for fire emergency scenarios. The game engine features a mobile-phone based VR headset for immersion. It allows multiple users to connect to the simulation at the same time and move user-controlled agents. The training environment features a virtual single platform underground station. The computer-controlled agents are simulated by distributing the agents evenly throughout the structure at the beginning of the simulation and then updating the agents' locations *"randomly"*. However, the only interaction capability is that the user-controlled agent can be moved by the user to either end of the train platform. One end of the train platform features a fire smoke simulated using Unity3D's built-in particle engine and is not based on real fire data. The user-controlled agent cannot interact with the *"randomly"* moving computer-controlled agents in any way.

Chittaro and Sioni [110] from Udine HCI labs developed a VTE featuring smoke, toxicity, fire and explosions. The aim of the VTE is to teach the user preparedness for different types of emergencies. The user is guided through a predetermined route and through a set of predefined checkpoints, where each checkpoint demonstrates a specific part of the training. The user has access to some artificial information such as a life bar, guiding arrows and highlighting of environment. Smoke and fire are an artistic representation that reduce the life bar of the user when user controlled agent is being exposed, e.g. *"the first time users try to exit the train, their life bar decrease is set to a speed that makes it impossible to reach the goal"*. Each checkpoint teaches a different type of hazard such as a smoke, toxicity, fire or explosion hazard. The values for each hazard are taken from *"traditional, publicly available civil defence materials"*. The user-controlled agent cannot interact with any other agent. Multiple simultaneous users are also not supported in the same simulation.

Viant et al. [111] developed a VTE using Unreal Engine to train fire fighters to fight virtual fires. The game engine (i.e. Unreal Engine) provides attractive graphics that are believable to the user. The training environment is limited to a single trainee at a time and does not allow multiple trainees to fight the same fire at the same time. The trainer has full control over the size and type of fire and the colour, density and type of smoke. The fire and smoke are an artistic representation of what the trainer wants it to be and its progress is also artistic and controlled by the trainer. The fire and the smoke are not based on any real-world data, including their obscuration of visibility or progress. The training environment does not include any interaction with any populations or individual agents. Interaction with the environment is achieved by highlighting interactive objects with a glow and allowing the user to click on them, which opens a menu to the user showing all possible interactions allowed with the selected object.

Ha et al. [112] developed a self-training VTE using Unity3D game engine. Unity3D game engine enabled Ha et al. to easily connect multiple users into the same environment using the Unity3D game cloud server with the built-in support for VR headsets. The smoke and fire were an artistic representation provided by the game engine and designed by the developer. The environment did not provide a population and therefore no interaction with computer-controlled agents. However, the interaction between the user-controlled agents were done by using gestures. The gestures were recorded from capture devices attached to arms and legs

and synchronised to the game engine.

The VR environment developed by Backlund et al. [113] is presented to the user in the same way as the VR environment developed by Lund University [114–117]. The user stands in a room where the surrounding walls show projected images from multiple projectors. The projected images are integrated to provide a complete surrounding VR environment to the user. Backlund et al. used Half-Life 2 game engine for the VR environment. The environment uses the game engine's features extensively, such as showing a health number of the user, which decreases when the user-controlled agent is in smoke and replenishes when standing next to the virtual trainer in a safe zone. The smoke is simulated with a custom simplified physics model [113, 118] which makes the smoke particles rise to the ceiling from a source location and start filling the room evenly from the ceiling all the way to the ground until the whole room is evenly filled with smoke. Other smoke behaviour and characteristics, such as visibility in smoke, are based on an artistic representation set by the developer. The training is split into multiple levels that are scripted to provide a specific set of training objectives. This limits the training from repeating the exercises as they will play out exactly as before, following a set of scripted events and triggers. The levels have been designed with an increased difficulty where the same smoke in a later level does more harm to the user than in an earlier level. The VTE does not simulate crowds or agents that can be interacted with. However, it does simulate victims as bodies that must be carried to safety.

This section has shown that there is a lot of interest in developing and researching VTEs for fire emergency scenarios and the need for better and more realistic immersion in VR. However, fire or its products (i.e. smoke, heat and toxic substances) are not represented accurately in any VTEs with publicly available research. Furthermore the VTEs for fire emergency training lack validated and realistic evacuation behaviour dynamics with the lack of or very limited interaction with computer simulated agents (i.e. one-to-one interaction). This highlights the urgent need for the IRIS system as described and presented in this thesis.

2.3.6 VR in pedestrian dynamics

The Division of Fire Safety Engineering in Lund University has been using VR in pedestrian dynamics research more broadly [114–117, 119–122].

Kinateder et al. [117] experimented on route finding and decision making in VR to see what path someone would take to evacuate and if the path would change when they see someone else evacuate before them. The results indicated that if the participant saw a virtual agent (another person) evacuating, the participant was likely to follow the agent and not take a different route. Another interesting finding was that when smoke was introduced, the participant's behaviour was in line with findings from real smoke evacuation where occupants used walls as means of orientation in smoke filled tunnels. There was no interaction between the user-controlled agent and the computer-controlled agent of any other kind than the user being able to see the computer-controlled agent.

Ronchi et al. [114, 115] ran VR experiments to test the effects of flashing lights at emergency exits. The research included variations in the colour of light, rate of flashing, type of light and layout of the lights. While the experiment did not include other agents, smoke or evacuation, the results showed the importance of testing real world visibility of signs in VR. The participants were able to see variations in emergency exit sign deployments without having to construct these in the real world. Moving the participant from one test to another is as effortless as a press of a button, which allows more tests to be performed in less amount of time while saving in the costs of actually building real world test environments.

Cosma et al. [121] used a VR environment to test effects of lighting in evacuation through a smoke-filled railway tunnel. Each participant was placed in one of two different way finding installations. Each participant had to evacuate through smoke, which was simulated using FDS [123]. While FDS can produce smoke visibility estimates as one of its output values [124], the research by Cosma et al. does not specify how the visibility was calibrated or tested to ensure the smoke is accurately visualised to users.

Andrée et al. [116] performed VR experiments on determining whether people would use lifts or staircases to evacuate. No smoke or fire were shown to the participant, but

some participants were told through announcements that there is a fire in the building. The participants started in a hotel room and had to navigate out of the hotel through a corridor, which provided options to either take a lift or a staircase. Various exit signs were placed as part of the experiment to test their effectiveness. The whole experiment was done in VR including the lift journey and descending through the stairway.

Arias et al. [120] developed a system called ForensicVR to use VR in forensic investigations of fire scenarios. The purpose of the system is to aid in the study of human behaviour in well documented real fatal fires. The key finding of the paper was that *"participants in the scenario behaved in a comparable way as the victims of the corresponding real fires"*. The participants made similar decisions to the victims in well documented incidents and experienced some level of stress despite knowing the threat was not real.

The prior research in pedestrian dynamics using VR has shown VR environments to be an effective tool in analysing and studying human behaviour as well as practising way finding and evacuation procedures. Smoke is often used as part of the evacuation exercises in VR even to the point of using CFD Fire simulations to calculate complex fluid behaviour, but the visual representation has been left as an artistic assumption of what the smoke visibility may look like. However, the participants have been found to be behaving similarly in VR to well documented real-life fatal fire victims.

2.3.7 Interaction interface with virtual characters

The aim of the IRIS system is to provide interaction between user-controlled agents and computer-controlled agents in a virtual environment. The interaction should be immersive and therefore natural/realistic between the virtual agents. This section will look into the importance of both gestures and voice in communication between humans and therefore what can be learned and implemented in the virtual environment.

In 2004 Dugdale et al. [12] proposed a VR emergency fire incident training tool, in which a trainee has conversations with virtual characters. The virtual training environment visualises a hotel fire and the following virtual characters: a hotel owner, several guests, support firemen and a chief fireman. Dugdale et al. [12] performed field studies of analysing

relevant documents, interviewing instructors and senior fire-fighters and gathered 12 hours of on-site video footage. The analysis of these studies was used to identify the relationship between gestures and verbal activities in human communication. The results from the field studies were compared to video footage recorded from interactions with the virtual training environment. Dugdale et al. [12] concluded that in a virtual world *"gestures play a crucial role in human communication and was necessary for the design of realistic interaction between the virtual characters"*. The studies also revealed that *"it is important that the user is able to drive the virtual person in such a way as to produce the appearance of orientating to other virtual persons. This could, for example, signal to another virtual character that they are looking at each other. Looking at another may be a device to judge whether a gesture is necessary."* [12] as it is used as part of the verbal and non-verbal (i.e. hand gesture) communication in real-world.

In 1992 McNeill [125] in his book explains in detail how different arm, hand, finger and body movements all affect how a visual gesture is understood and how these movements have close relationship to what the speaker is saying. He also points to other interesting findings, such as: *"Butterworth, a scholar in Britain has argued that many gestures arise in response to speech failures"* [125], but McNeill argues that gestures are used to emphasise the narrative of the speaker and can convey information that the speech left out and vice versa (i.e. speech can convey information the hand gesture left out).

In 1998 Cassell et al. [126] performed a study on how information is conveyed using only gesture or only speech or both combined and highlights the importance of using both gestures and speech together. Cassell et al. [126] write: "In summary, in demonstrating that speakers' gestures are integrated into hearers' representations of the information conveyed, this work supports the position that gesture and speech are two integrated systems, allied from the very moment they are viewed to the last stage of their comprehension and use in constructing a model of what was said.". In the study by Cassell et al. [126] discovered that the listener/observer (i.e. the recipient) of the message is more likely to understand and receive the communicated information if he can both see and hear the same information, and the probability is lowered if the recipient is only able to hear or only able to see the communicated message [126].

2.3.8 Natural graphics

To have a real human user to interact with the virtual world in a natural way, it is important for the virtual world to provide natural information that are required for the user to make informative decisions. As was demonstrated by Pan and Niemeyer from Disney Research [11], when a human user catching a real-life ball can see the ball in the virtual world, the user catches the ball naturally. However, when the ball is not visible, but the user has other unnatural information available to them such as a flight path or a landing zone painted in the air in real-time in the virtual world, the human user adapts to new artificial information in the absence of natural representation of the ball. As a result, the human user reacts and behaves very differently in this virtual world to successfully catch the ball, than he does in a real world where the information is presented differently i.e. the ball is visible, its flight path or the landing zone are not drawn in the air. This demonstrates a crucial point that for the user to have natural behaviour in a VR, it needs to have naturally available information to him as well as equally importantly have a lack of artificial information available to him.

Waller et al. [105] also discovered, in the 1998 experiment, large differences between men and women when using VR environments: *"women's performance on virtually all tasks tended to lag behind that of men"*, but only in the VR environment. Waller et al. [105] point out that in the real-world training women performed similarly to men. This was suggested to be based on research in 1995, which had indicated that men generally have more experience with video games. However, almost 20 years later in 2014, research done by The Guardian [127] indicates that women now play more video games than men, thus potentially allowing women to perform even better than men in today's virtual training environments. In 1998 Sandstrom et al. [128] did further research into differences between the male and female learning performance in spatial navigation using a virtual training environment and drew similar conclusions about the differences between the genders on the level of experience with video games. Sandstrom et al. [128] also discovered that subject females relied on landmark information in the spatial navigation, whereas males used both landmark information and geometric shapes in the spatial navigation, this became prevalent when the expected information was artificially removed in the virtual world. It is therefore important to provide an accurate representation of real-life structures in virtual environments, which enables the

users (both male and female) to naturally use the presented information during training and during the testing of procedures in wayfinding and learning spatial knowledge. Providing the expected natural information as discovered by Sandstrom et al. [128] is necessary to avoid placing users of the IRIS system into unnatural scenarios when immersed in the virtual environment, and thus artificially altering and limiting the real-life behaviour and capabilities of the user.

While the importance of natural graphics has been discussed throughout this Chapter, the study by Disney Research provides a very clear demonstration on how important the naturally available information is for natural human behaviour. The change in the human behaviour is very clear in the demonstrated video by Pan and Niemeyer from Disney Research [129].

2.3.9 Virtual training environments summary

This section has identified potential use cases for the IRIS system in training. Full-scale training is expensive, difficult to organise and time consuming. The interactive VTEs with simulated agents are non-existent in law enforcement officer training. The IRIS system enables training of crowd management, which is important in large public gatherings and often done by law enforcement as identified in Section 2.1.2. VTEs can also be used for learning spatial knowledge prior to real-life wayfinding, which presents an opportunity to use the IRIS system in a realistic fire emergency scenario with accurate smoke visibility representation to learn wayfinding skills in realistic fire smoke. See Section 2.2.1 more details about the effects of fire smoke in visibility.

This section has shown that there is a lot of interest in developing and researching VTEs for fire emergency scenarios and the need for better and more realistic immersion in VR. However, fire or its products (i.e. smoke, heat and toxic substances) are not represented accurately in any VTEs with publicly available research. Furthermore, the VTEs for fire emergency training lack validated and realistic evacuation behaviour dynamics with either the lack of or very limited interaction with computer simulated agents (i.e. one-to-one interaction). The realistic evacuation behaviour is important when the user wants to evaluate or

test the impact of people's actions on a VTE. This highlights the urgent need for the IRIS system as described and presented in this thesis.

Dugdale et al. [12] concluded from their studies that in a virtual world *"gestures play a crucial role in human communication and was necessary for the design of realistic interaction between the virtual characters"*. The studies also revealed that *"it is important that the user is able to drive the virtual person in such a way as to produce the appearance of orientating to other virtual persons. This could, for example, signal to another virtual character that they are looking at each other. Looking at another may be a device to judge whether a gesture is necessary."* [12] as it is used as part of the verbal and non-verbal (i.e. hand gesture) communication in real-world.

2.4 Computer models and simulations

The previous section identified limitations in the current VTEs for fire emergency training and especially in relation to interaction with computer simulated agents. The realistic evacuation in a VTE is important when the user wants to evaluate or test the impact of people's actions. Evacuation and pedestrian models are being researched and developed to simulate people behaviour in emergency scenarios. These models could provide the key characteristics for the agent behaviours currently missing in VTEs. Therefore, this section aims to identify known evacuation and pedestrian models found in the literature that are suitable for user immersion and interaction. Before suitable models can be identified, it is important to establish key technical characteristics and capabilities necessary for immersion and interaction, and their refinement levels.

In the 2016 edition of SFPE Handbook Kuligowski [26] has reviewed 26 computer evacuation models for buildings. This list is particularly useful, because it explores the current capabilities that these models have and draws comparison between them. Therefore, if a scenario, other agents and the movement space can be simulated in sufficient detail, such as to enable interactivity between the agents and allowing the user-controlled agent to explore and navigate in the available movement space, it would allow the immersion of a human user as a computer-controlled agent in the computer simulation. The coarse network system does not

have precise location information [26] on agents, which is necessary for the user-controlled agent to avoid obstacles and barriers [26] as identified by the user while navigating in the virtual environment.

As was identified in the previous section, the interactivity model between the user-controlled agent and the computer-controlled agents should be accomplished using natural human communication like hand gestures and voice commands with the ability to target either individuals or crowds. This implies that the refinement of the population, the modelling method, and whether it is stochastic or deterministic must be considered. The microscopic level of refinement (as explained by Kuligowski [26]) assumes the model represents the population as individual agents, whereas the macroscopic level does not. The individual representation alone is not enough: the model will need to be able to simulate cognitive behaviours on the individual computer-controlled agents regardless whether they are part of a crowd or not. The behaviour simulation is necessary for each computer-controlled agent to react to the gestures and voice commands issued by the user-controlled agent. The utilised model cannot be deterministic when a non-deterministic user-controlled agent is interacting with the simulation and the simulation needs to be able to respond to the user's actions, especially so when modelling user-controlled interactions with a crowd. Therefore, the model has to be stochastic and reactive.

Table 2.1: Minimum refinement levels for movement in evacuation models to be suitable for immersion and interaction

Movement characteristic	**Refinement**
Movement space	Fine or Continuous network
Population refinement	Microscopic
Behaviour refinement	Stochastic and Reactive
Modelling method	Behaviour

The computer model should also be available and preferably provide some level of validation, since the crowd reactions need to be calibrated to be as representative of real behaviours as possible, factoring actual response times, and include representative action times. From

Kuligowski's list of 26 computer evacuation models for buildings [26], the following models are suitable for user immersion and interaction (see Table 2.1): ASERI, buildingEXODUS, Legion, PEDFLOW, SpaceSensor, EPT, MassMotion, Myriad II, CRISP, EGRESS 2002, MASSEgress. In addition to Kuligowski's list, the following models are also suitable for user immersion and interaction: VADERE [130], JuPedSim [131] and Menge [132]. From the previous lists, only the following models are publicly available: ASERI, buildingEXODUS, Legion, PEDFLOW SpaceSensor, EPT, MassMotion, Myriad II, VADERE, JuPedSim and Menge. From the publicly available models, only VADERE, JuPedSim and Menge have publicly available source code. However, this research is completed in Fire Safety Engineering Group (FSEG) in University of Greenwich. Since this work is carried out within FSEG, the author has exclusive access to the buildingEXODUS source code and research experience within the group. Furthermore, buildingEXODUS has a more extensive validation history than the other models. It is important to note that buildingEXODUS could be replaced by VADERE[4], JuPedSim[5] or Menge[6], as a drop-in replacement, assuming these fit the needs of the scenarios being deployed and are capable of running the scenario in question in real-time. The concepts and the proposed solution can be adapted to any of the other suitable validated models as long as the necessary access to modify the software is granted and the software is capable of simulating constantly (i.e. throughout the simulation) in real-time or faster.

2.5 Summary

The literature review performed in this chapter provided the necessary background research to identify the gaps in knowledge, and capabilities and requirements, which should be incorporated in any system for training. The background research also identified potential use cases for the IRIS system. The key highlights from this chapter are:

Crowd dynamics (see Section 2.1) are complex and therefore multiple specialised theories exist to explain different aspects of crowd dynamics in targeted scenarios. However, it

[4]VADERE is freely available from `https://www.vadere.org/`
[5]JuPedSim is freely available from `http://www.jupedsim.org/`
[6]Menge is freely available from `http://gamma.cs.unc.edu/Menge/docs/code/menge/html/index.html`

is generally accepted (see Section 2.1) that crowds are not a mindless set of people and are formed of individuals who make cognitive decisions, but the decisions are heavily influenced by the social ties an individual has with the crowd, including role, perceived status, relationships and shared goals. This has an effect on what kind of influence an authority or outsider can have on the crowd as a whole, nevertheless, it is important to model the influence on individuals of the crowd to form an overall influence on the crowd. Previous research shows (see Section 2.1) that an authority is an important influence on evacuee behaviour (i.e. high compliance), especially if the authority has had prior interaction with the individuals or surrounding individuals are already compliant, however, the range of the influence is limited to those individuals who can comprehend and see the authority.

Meng and Zhang [15] research showed (see Section 2.2) that the higher fidelity of virtual environments is necessary to achieve more realistic behaviour from the user in emergency scenarios, while pointing out that the user behaves differently in non-emergency scenarios than he does in emergency scenarios. Hence the importance to include calibrated models, which represent cognitive behaviours. Furthermore, the visibility in smoke has not been accurately visualised in real-time virtual environments even though this is a crucial part of the immersion in VR to produce realistic behavioural results from the user. This thesis proposes a solution in Chapter 4 to this.

As a conclusion the IRIS system should include:

- A virtual user-controlled agent with interaction capabilities with individuals or crowds of computer-controlled and simulated agents.
- Plausible interaction between user-controlled agents and computer-controlled agents using natural human hand gestures and voice commands.
- Fine or continuous movement model capability from the behaviour model as well as the modelling of individual and crowd behaviours.
- Immersive (natural) representation of the virtual world in emergency scenarios, especially in terms of accurate representation of visibility in fire smoke, effects of toxic gases and heat on computer-controlled agents as well as the user-controlled agent.

- Accurate smoke visibility and fire hazard data from a sophisticated CFD fire simulation tool.
- Run in real-time.

The possible use cases identified for the IRIS system are:

- Testing of procedures in emergency and non-emergency scenarios with crowd behaviour management.
- Training first responders (i.e. firefighters and law enforcement) in fire and non-fire emergency scenarios with crowd management.
- Training spatial navigation in fire with or without the presence of a crowd.

Further implications are for the usage of accurate real-time smoke visibility rendering in more general applications such as in the real-time entertainment industry where virtual immersion is in high demand and accurate representation of real-life phenomena is required to enable that.

Chapter 3

THE SYSTEM

The previous chapter (Chapter 2) identified limitations in existing VTEs (see Section 2.3) and pedestrian simulation software (see Section 2.4), including the missing link between them. Current VTEs in law enforcement (see Section 2.3.3) or in firefighting (see Section 2.3.5), or VR in pedestrian dynamics (see Section 2.3.6) provide very limited interactivity, if any, with simulated individual agents and crowds (see Section 2.3 in the previous chapter). Utilising existing research software (namely pedestrian simulation software) to model agent behaviour in interactive VR environments allows the environments to provide more realistic and more sophisticated agent behaviour. The linking of the existing software components is the first research contribution (see Section 1.3.1 in Chapter 1) and the link is detailed in this chapter.

The key solution to linking the three software components together to form a single unified IRIS system (see Section 1.3, Chapter 1) is proposed in this chapter. Graphics developments are covered in the next chapter (Chapter 4) and user interactivity is covered in Chapter 5. The work presented here covers the key developments in the linking of existing software products and the development of necessary auxiliary programs that aided the development of the IRIS system. The proposed solution is split into three main categories: system architecture, network design, and network protocol implementation.

Figure 3.1 is presented on the next page to assist in describing how the different sections and components in this chapter and the next two chapters fit together as a complete system. The components developed as part of this research are presented by blue boxes and blue arrow. The thread model, network interface and communication protocol are detailed in this

chapter.

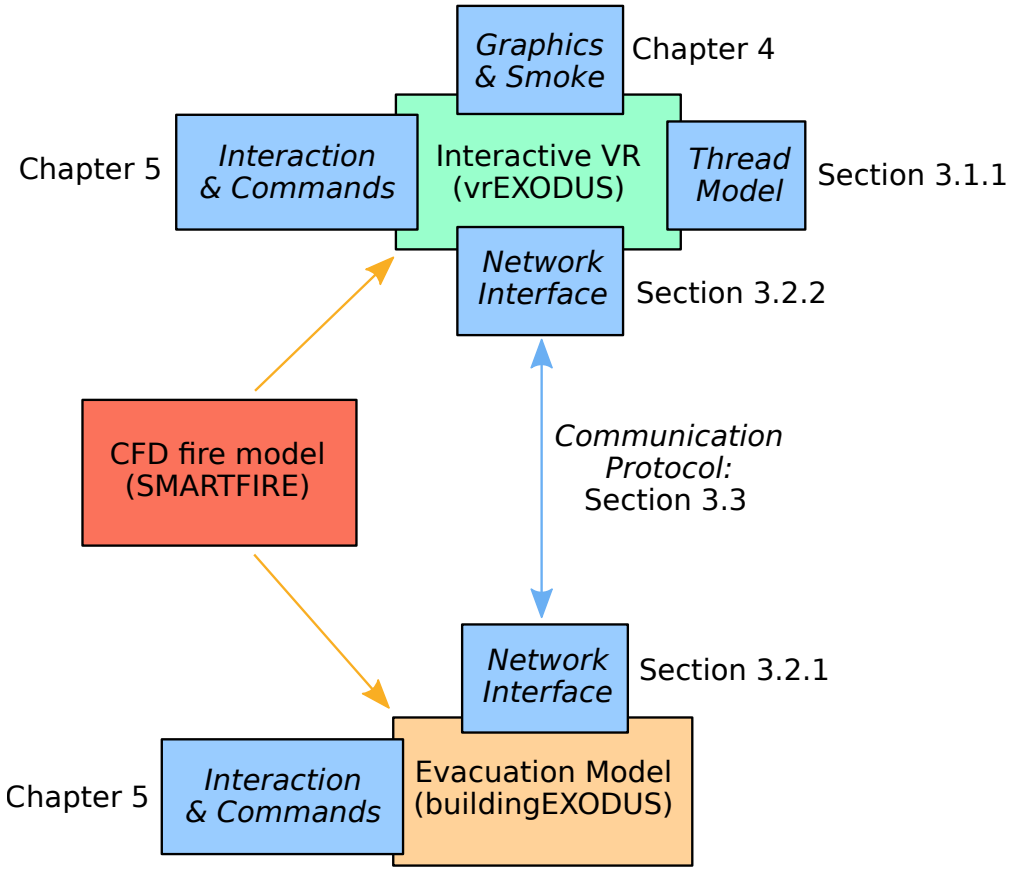


Figure 3.1: System overview to chapter and section mapping. New components introduced in this thesis are coloured in blue. The existing components, green, red and orange coloured boxes are as described previously in Chapter 1 Section 1.3.1.

System architecture is outlined in Section 3.1. The system architecture section delineates the required thread model (Section 3.1.1) for the virtual reality (VR) environment to enable faster graphics rendering performance with simultaneous network communication. It describes how the internal data structure is formed to allow concurrency as well as which CPU threads are accessing which data stores.

Network design is outlined in Section 3.2. The network design section covers both the server-side network interface in Section 3.2.1 and the client-side network interface in Section 3.2.2. Details on the individual components developed to enable highly concurrent and real-time network communication while minimising its impact on the simulation performance are described. It further details the flow of data from one component to another on both the server (evacuation model) side as well as the client (VR environment) side.

Network protocol implementation is outlined in Section 3.3 describing an efficient and extensible TCP binary communication format. The low-level details of the network protocol, including the disassembly of the message structure are analysed and described in the network protocol implementation section.

The aim of this chapter to answer the question of: *How could three existing software components, namely evacuation model, fire model and VR environment, be linked together to make a single interoperable system, the IRIS system?*

3.1 System architecture

System architecture is the most crucial part of any complex system design. System architecture defines and ultimately lays the foundation of how different parts of the system fits together. An overview of the linked parts being implemented as a complete IRIS system can be seen in Section 1.3, Chapter 1, which is also shown in Figure 3.1.

The most crucial change to the previous version of vrEXODUS was to make it multi-threaded and therefore enable the required performance for the goal of this research, the IRIS system. Multi-threaded design and communication is difficult in scientific and technical computing as identified by Pancake [133], especially when built in a way that enables higher performance by minimising communication between threads (i.e. thread independence) [134], see Section 3.1.1.

Another significant development was to add networking capabilities to two single-user programs, EXODUS and vrEXODUS. Section 3.2 details this, including the majority of the network communication.

3.1.1 CPU thread model

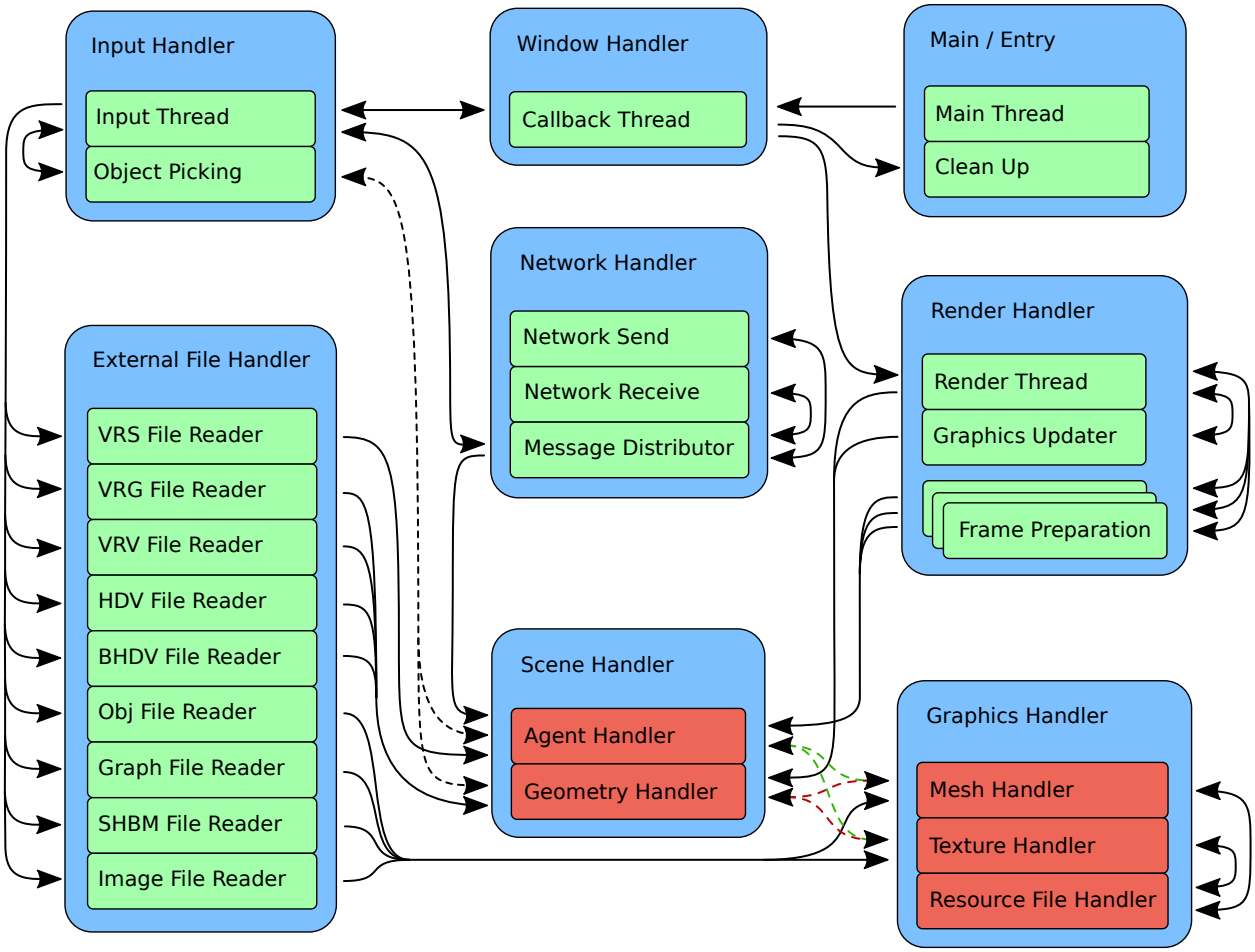


Figure 3.2: Multithreaded process model of vrEXODUS

Figure 3.2 shows the thread layout of the new vrEXODUS. A green coloured box shows a single thread with the exception of *Frame Preparation* threads (indicated with a stack of green boxes), where the size of the stack is determined with the following formula: $number_of_threads = max(number_of_cores - 3, 1)$. This would ensure that there is always at least one thread preparing the frames. The negative 3 in the equation was used to allocate a free CPU core for the *Render Thread* (see *Render Handler* in Figure 3.2) as well as one for the *Graphics Updater* (see *Render Handler* in Figure 3.2) as these two are normally the bottleneck of the entire program. This leaves still one free CPU core in multi-core environments for the rest of the threads and the operating system.

The boxes in red in Figure 3.2 are the main data handlers. These are greatly simplified

to keep the diagram readable and to help maintain the focus on the threads. To help follow the arrows from one box to another (in Figure 3.2), some arrows have a dotted line, whereas most have a solid line. The dotted lines were added to help the reader follow the arrows. There is no other significant difference between the arrows.

To understand the diagram (Figure 3.2), it is important to step through the process of how data travels from one thread to another. To keep the threads from halting each other with mutexes, most of the communication happens asynchronously with an implementation of non-blocking atomic queues by Michael and Scott [135].

At the start of the program and before the *Main Thread* starts the *Callback Thread*, the *Main Thread* reads settings files and initiates basic data storages. The settings from the settings file determine things like key bindings and previous window location etc. The *Window Handler* is initiated by the *Main Thread*, which in turn starts a *Callback Thread* for the main Microsoft Windows message loop. This is obligatory for any program that provides a graphical user interface (GUI) in a Microsoft Windows operating system.

Window Handler initiates *Input Thread* and will pass any input messages registered by the GUI window to the *Input Handler* such as mouse and keyboard messages. It will also listen for any user inputs to close the program, in which case it will start a *Clean Up* thread. *Clean Up* thread informs all other threads that the program is about to terminate and will clean up any allocations before shutting down. The *Window Handler* also starts the *Render Thread*, which is responsible for drawing everything visible on the GUI window.

Input Thread checks for new inputs 60 times a second, which is a typical monitor refresh rate. There is very little point registering input faster than it can be shown on the screen. If for some reason this frequency deems inadequate, it is trivial to change. The rest of the time the *Input Thread* sits idle. *Input Handler* starts the *Object Picking* thread. *Object Picking* thread's responsibility is to match the current mouse coordinates on the window to any objects seen in the geometry or agents. This picking functionality was separated into its own thread as it will take an unknown amount of time to go through all possible objects as sometimes there can be millions of walls or hundreds of thousands of agents to go through. In the extreme cases the selection speed of objects suffers but does not affect negatively the

rest of the program. *Input Thread* is also responsible for initiating user interface selections with a mouse and user input to text fields from the keyboard, and these are passed along to the *Window Handler*, which in turn will update the Render Handler if necessary. The *Input Thread* will talk to the *Network Handler*, by starting the *Message Distributor* and passing any inputs that need to be delivered to the server, to the *Message Distributor* thread. *Input Thread* is also responsible for starting all external file readers in their own threads as this is performed as part of a mouse drag event.

External File Handler threads are self-explanatory i.e. if a file type is recognised, it is will be read. Any errors will propagate in the typical error handling way of backtracking all the way to the *Window Handler*, which will display the error message. Once an external file is read correctly, its data is parsed and passed along to the appropriate data container.

Network Handler is exclusively dealing with the network traffic. More detail is covered in Section 3.2. The IO is separated into its own threads: *Network Send* and *Network Receive*. These threads can take as long as they need without negatively affecting the performance of the rest of the program. *Message Distributor* thread will exclusively filter the appropriate messages to and from the *Network Send* and *Network Receive* threads. It will update when necessary the agent data in the *Agent Handler*. Any necessary inputs and messages are passed along to the *Input Handler*.

Render Handler is the most central part of the program. Its function is to display everything that the user sees. It renders everything related to graphical output, such as the GUI window, the geometry, agents and hazards. The slowest part of the vrEXODUS program is the *Render Thread*. The data that it has to process is optimised as much as possible before it reaches the *Render Thread*. OpenGL allows a shared context object, which lets a separate thread share memory with another thread. There are some caveats and limitations to the shared context, such as during development it was found out that if another thread is uploading data to the GPU, the other thread cannot communicate with the GPU at the same time and is effectively being locked out by the uploading thread. Therefore, it is important that all uploads to the GPU are split into small chunks. Nevertheless, the context from *Render Thread* is shared with *Graphics Updater* thread (see Figure 3.2). The purpose of *Graphics Updater* is to make sure any necessary 3D models and textures are loaded onto

the graphics card memory before they are used by the *Render Thread*. This means that the *Render Thread* can keep updating the screen and keep it responsive to the user without suddenly having to upload large amounts of data to the graphics card, resulting in a noticeable delay between rendered frames. *Graphics Updater* will merge and compile multiple similar object types into single larger objects to help minimise the number of graphics calls the *Render Thread* has to make. Even as many as millions of walls can be reduced into as few as one or two graphics card calls. However, not all data can be loaded to the graphics card's memory, such as the agent movements and actions. The agent data is the most expensive data to process in real-time as it changes on every frame and often in large quantities. These optimisations are the most effective in large scale scenarios, where there are tens of thousands or hundreds of thousands of agents moving at the same time. The purpose of the *Frame Preparation* threads is to use the rest of the available processing power on the CPU in a multi-core environment to process the next animation frame the agents make and pass this pre-processed data to the *Render Thread*. If the frame takes a long time to prepare, the *Render Thread* will keep rendering the previous agent movement frame until an update is ready. This means that sometimes *Frame Preparation* threads need to skip animation frames to keep up with a real-time clock. The frames are separated from the *Render Thread* to keep the user interface and the program responsive. The user can still select and move around in the virtual environment in real-time even if the *Frame Preparation* threads struggle to keep up.

The *Scene Handler* and *Graphics Handler* represent two different types of data. The *Scene Handler* holds all the scene related data such as agents, walls, stairs, lifts, escalators, ticket gates and smoke. However, the *Graphics Handler* handles the actual raw 3D data in the *Mesh Handler* to represent any of the fore mentioned objects as well as textures in the *Texture Handler*. The *Resource File Handler* handles any bundled files that are supplied with the vrEXODUS program. These are packages in an archive with checksums to verify that the data is intact and present.

The main limitations of the thread model are the obligatory single threaded approach of OpenGL. As can be seen throughout this Section, most of the focus of the multi-threaded thread model was clearly around this limitation to minimize the impact of this bottleneck. Unfortunately, at the start of the research this was the best documented and best fit for

purpose library available. This has since been surpassed by the Vulkan library that was announced in 2015 [136]. An upgrade to use this superseding and modern multi-threaded library is left as future work as covered in Chapter 7.

3.2 Network design

Network design is the second major internal change to both vrEXODUS and EXODUS software. This enables the multi-user connectivity across different computers for the IRIS system. The overall network connectivity design is one of the most common types used across the Internet [137]: it features a centralised server with multiple clients. This is also commonly known as a *star network topology* [138].

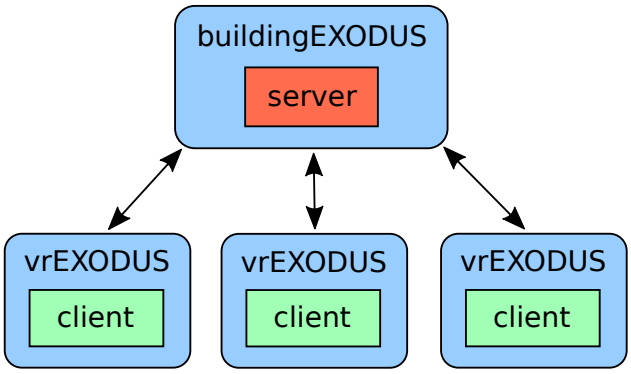


Figure 3.3: Server-client architecture

The high-level design is simple: EXODUS software (in this case buildingEXODUS) has a new network library implementation shown in red in Figure 3.3. vrEXODUS also has a new network library implementation shown in green. These network implementations are multi-threaded independently from the rest of the simulation and VR to minimise their negative performance impact. The network design is asynchronous by design to enable independent communication from the rest of the system (i.e. the simulation, rendering and file handling).

To enable client to client communication, EXODUS server acts as a hub between the clients. This allows the clients to sit behind firewalls and only the server has to expose an open port.

3.2.1 Server network library implementation design

This section discusses the lower level design of the server's network library implementation, see Figure 3.4.

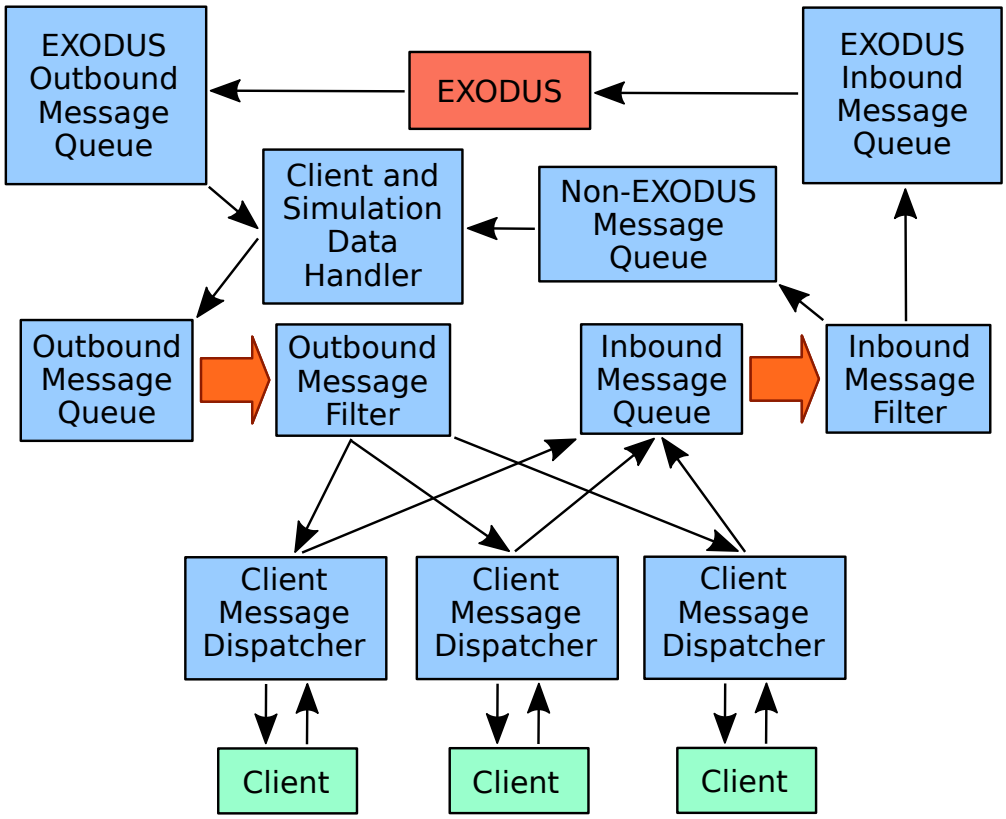


Figure 3.4: Server-side network design

The design choice was to minimise any additional load introduced by the added network capability on the EXODUS simulation. EXODUS simulation is running on a single program thread while most modern CPUs are multi-core CPUs favouring multi-threaded applications. The bottleneck of this system is the EXODUS program shown in red. To maximise the potential use cases of this implementation the aim is to minimise the load on EXODUS and utilise the multi-core computing architecture for the network communications.

Network messages arrive and leave the server asynchronously through the independent *Client Message Dispatchers* (see Figure 3.4), this minimises any slowdown effects from slow clients. As the server gets a message from the client, it is collected as a whole by the *Client Message Dispatcher* and placed into an *Inbound Message Queue* (Figure 3.4). The *Inbound*

Message Queue contains all the incoming messages from all the connected clients and are dealt on a FIFO (First In First Out) basis. The messages from the queue are sequentially interpreted by the *Inbound Message Filter* shown as a red arrow in Figure 3.4. The filter decides whether the message concerns EXODUS or if it can be dealt by the server program and therefore, minimising the load on EXODUS software.

EXODUS *Inbound Message Queue* and EXODUS *Outbound Message Queue* are based on the same atomic non-blocking queue by Michael and Scott [135] as was used in the thread model in Section 3.1.1. In comparison to a traditional blocking queue these reduce the impact that the server program has on EXODUS simulation performance.

When the messages are not directed to the EXODUS program, they are handled by the *Client and Simulation Data Handler*. These messages are not meant to affect the simulation and are for such purposes as: enabling client to client communication and client to server network communication handling messages.

3.2.2 Client network library implementation design

The client network library implementation design follows the same principles of the server implementation design with the exception of having to deal only with a single connection.

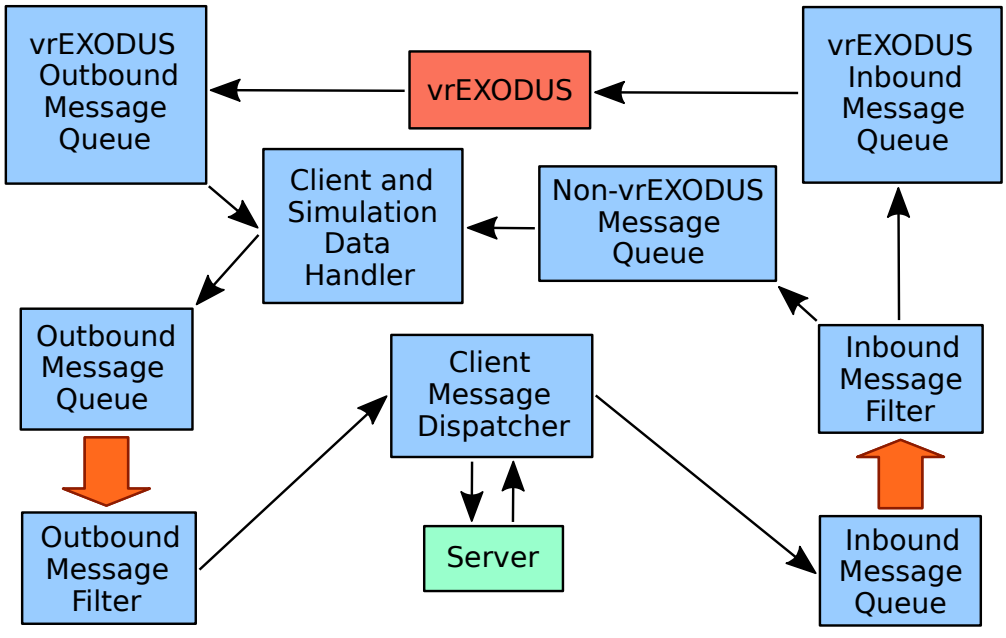


Figure 3.5: Client-side network design

vrEXODUS (the client) provides the immersive and interactive environment to the user. The client is responsible for collecting user input and providing feedback to the user in a responsive manner. The user input is sent asynchronously to the server by placing them to the vrEXODUS *Output Message Queue* (see Figure 3.5). The server messages that vrEXODUS should know about are retrieved from the vrEXODUS *Inbound Message Queue*.

There is only a single *Client Message Dispatcher* unlike in the server implementation as there is only a single target host to connect to. The *Client Message Dispatcher* is multi-threaded with *Network Send* and *Network Receive* threads as part of the *Network Handler* defined in Section 3.1.1.

The new messages received from the server are placed into the *Inbound Message Queue* and the new messages for the server are found in the *Outbound Message Queue*. Both of these queues have their own filters and this process is shown as a red arrow. The *Outbound Message Filter* does not have much logic to it as all the messages in this implementation go to a single host. However, this allows future enhancements to enable multiple servers such as in a cluster implementation of EXODUS software. The *Inbound Message Filter* splits the messages to two categories: network only related messages and vrEXODUS update messages.

3.3 Network protocol implementation details

This section details a new bespoke network protocol implementation for the IRIS system. An overview of the IRIS system can be found in Section 1.3, Chapter 1. The network protocol aims to provide the real-time link between the VR environment (clients) and the simulation software (host). See the previous section (Section 3.2) for more details on the network design.

To provide control over the simulation to the VR environment where the end users will be interacting with the simulation, it is necessary to define certain key messages. The user may want to load a specific scenario (Section 3.3.2) and a specific fire hazard (Section 3.3.3). The user may want to start at a specific part of the simulation, such as when a certain number of agents are present in the simulated geometry (Section 3.3.5). To aid in filling up the geometry to the desired agent count limit, the user is given control on how fast the geometry is being

filled during the simulation (Section 3.3.4). The rest of the messages detailed in this section are to do with synchronizing the real-time simulation and user inputs.

The user is given control over the messages to start an alarm (Section 3.3.7) and to start a fire hazard (Section 3.3.8). The user also has the ability to join the simulation at a desired location as a user-controlled agent (Section 3.3.13) and the simulation software always confirms the addition of the new agent (Section 3.3.14). The VR environment also informs the simulation software when the user-controlled agent moves (Section 3.3.10) or if the user-controlled agent exits the geometry through an exit (Section 3.3.11). The user interacts with the crowd indirectly by being part of the geometry, which forces the simulated agents to navigate around the user, or by issuing hand gestures and voice commands to other agents (Section 3.3.15). Some of the gestures and commands require the simulation to know the direction the user is currently facing, which is updated automatically as the user looks around in the geometry (Section 3.3.16). The hand gestures and voice commands are covered in more detail in Chapter 5.

While the simulation is running, the VR environments (clients) are updated with the progress of the simulation. The updated terms are time (Section 3.3.6), addition of agents (Section 3.3.9), removal of agents such as when exiting the geometry (Section 3.3.11) or killing of agents such as by fire (Section 3.3.12), and by moving the agents during the simulation (Section 3.3.10).

All network messages of the IRIS system are transmitted over a TCP/IP (Transmission Control Protocol) connection. TCP/IP offers reliability and streaming of data. Streaming is a higher-level abstraction on OSI level 4 (Open Systems Interconnection model) over network packets in OSI level 3 than UDP/IP (User Datagram Protocol). UDP/IP has been commonly used historically for lower latency implementations, where low latency has higher priority than reliability [139, 140]. The UDP/IP is still a viable choice over the Internet when quality of the transmission route is unpredictable [141, 142]. However, as networks get more reliable [143, 144] and especially when used in LANs (Local Area Network), TCP/IP can be a reliable option with little to no noticeable increase in latency. TCP/IP reduces the complexity of the network implementation by design and substantially simplifies the transmission of large quantities of data [141]. There are some drawbacks to using TCP/IP

over UDP/IP [141]. If a single packet is dropped using TCP/IP, the entire stream halts until that packet is successfully retransmitted. Also, as the typical MTU (Maximum Transmission Unit) size of a data packet for LANs is 1500 bytes [145] (Ethernet v2 framing) and TCP/IP header uses 20 bytes per packet, where UDP/IP header uses only 4 bytes (i.e. TCP/IP is more expensive, but not significantly). With the typically fast transmission rates within LANs, this trade-off for increased reliability and simplicity is acceptable.

The network communication protocol for this work, built on top of TCP/IP stream still requires additional specification to enable the definition of messages. A single message is a derived variation from the usual memory storage implementation of data arrays and character strings. The message starts with a length identifier, which enables the recipient to know exactly how much data belongs to each message (Table 3.1). The length identifier is followed by the message type, uniformly across all messages. This provides forwards compatibility as well as possible implementations of clients that do not need to understand all messages, such as the coordinator may use a different software to visualise and analyse the simulation, while the participants may use a different software that includes interactivity. For this research the capabilities for the coordinators and the participants are implemented into the same software, but it important to note that the IRIS system does not impose this limitation.

3.3.1 Network protocol message container

Table 3.1: Network protocol message container

Part:	length	type	body
Size:	32 bits	32 bits	(length * 8 bits) - type length

Every message contains *length* and *type* and *body* parts as seen in Table 3.1. The subsequent tables in this section will only contain the *body* of each message. However, a message is not required to have a body, such as when it is used to signal responses (i.e. a signal message). There are only 2 types of variables used: 32-bit big-endian signed integers (Int32) and 32-bit little-endian signed floats (Float32). The endianness was changed towards the end of the

research to provide cross-compatibility with AUGGMED Unity3D project (see Chapter 1). This is trivial to change in C/C++ when the program has been written in endian independent way [146, 147] and can be updated at any time.

The rest of this section details the final versions of the used network messages.

3.3.2 Load scenario message

Table 3.2: Load scenario message body

Parameter	**Type**	**Description**
Scenario Id	Int32	Index to a selection of pre-defined scenarios, which EXODUS should load.
Random Id	Int32	Index to a selection of pre-defined random values in EXODUS for this scenario.

Table 3.2 shows a typical first message sent from vrEXODUS to EXODUS. EXODUS can respond to this message with a *scenario loaded* or *scenario failed* messages. These are signal messages.

3.3.3 Load fire message

Table 3.3: Load fire message body

Parameter	Type	Description
Location Id	Int32	Index to a selection of pre-defined locations for the fire. These locations are carefully selected and simulated in SMARTFIRE ahead of time.
Fire Type Id	Int32	Index to a selection of pre-defined fires, which EXODUS can load. These fires were run by SMARTFIRE ahead of time in a specific location and necessary hazard file conversions are needed to map the fire correctly to an EXODUS scenario.

Table 3.3 is similar to the Table 3.2. The message is sent from vrEXODUS to EXODUS. EXODUS can respond with *fire loaded* or *fire failed*. These are also signal messages.

3.3.4 Set flow rate message

Table 3.4: Set flow rate message body

Parameter	Type	Description
Flow Rate	Float32	The combined flow rate of agents (PPM) into the structure.

Table 3.4 is yet another initialisation message. The message is sent from vrEXODUS to EXODUS. EXODUS responds with a *flow rate set* signal message. There is no reason why EXODUS would fail to correctly alter the flow rate with the information in this message. Perhaps a range of allowed flow rates could be defined, but as the maximum flow rate is defined by the size of the entry nodes: defining an unreasonably large flow rate would not be able to exceed the maximum physical space that the agents can enter. On the other

hand, defining an unreasonably small flow rate (such as a negative) value would only stop more agents from entering. Agents are not removed with a negative flow rate. However, not allowing new agents to enter the scenario would eventually diminish the number of agents until there are none left.

3.3.5 Set agent count message

Table 3.5: Set agent count message body

Parameter	**Type**	**Description**
Agent Count	Int32	The simulation will run until this (i.e. *Agent Count*) many agents are in the simulation and then stop, indicating with another message that it is ready.

The message detailed in Table 3.5 is used to pre-populate the simulation to a certain size. This is a useful way to make sure a very specific number of agents are present at the start of the real-time multi-user scenario. To use the populate message correctly, it needs careful consideration of other aspects of the scenario and the underlying model. For instance, setting a scenario to populate to 2 million agents would require a very large space and flow rate to accommodate the requirement. The simulation would also likely take a substantial amount of time to reach that population size. The theoretical maximum number of agents that can be populated with this command is $2^{31} - 1$ (which is more than the population of the largest country[1] in 2018) due to the Int32 data type. This message is sent from vrEXODUS to EXODUS. EXODUS will respond with a *ready* signal message once the required target has been reached.

[1]A list of countries in the world by population: `http://www.worldometers.info/world-population/population-by-country/`

3.3.6 Time update message

Table 3.6: Set current simulation time message body

Parameter	Type	Description
Time	Float32	Current simulation time in seconds at this very moment.

EXODUS updates all clients at the end of each time step. Time is moved forward with a message in Table 3.6. EXODUS calculates and finishes the current time step and once the time step is calculated, it composes the time update message and sends it to all the clients. The time does not move forward from this point onwards until EXODUS updates the clients with a new time update message. Immediately after the time update message EXODUS composes all other necessary update messages and sends those to the clients. Once these updates are composed and set on the EXODUS *Outbound Message Queue* (see Figure 3.4 and Section 3.2), EXODUS can then immediately continue to process the next timestep of the simulation without having to wait for each message to be delivered, because the networking is handled asynchronously in separate threads.

The clock rate of the simulation is controlled by the EXODUS server. The server does not need to know whether or not a client successfully received and understood the time update message as in the next clock tick a new time message is sent with an updated time value, which renders the previous message obsolete. However, the clients have total control of the playback state of the simulation, such as if it should start, continue or pause. A client can inform the EXODUS server about the new desired playback state with the following signal messages: *simulation start*, *simulation continue*, and *simulation pause*. The simulation can be forced to terminate with a *simulation terminate* signal message or reset with *simulation reset* signal message. The difference between terminate and reset is that reset is expected to revert back to the initial state of the simulation where it can be started from the beginning with the previously defined scenario parameters, whereas the terminate message will ask the simulation to finish (without resetting or continuing) and generate a simulation results file that can be saved on the disk and later analysed by the coordinator. An important feature

of the termination message is that it does not reset the simulation state back, but freezes it still, in the current simulation time, so that the current situation can be examined more closely (i.e. if there is a problem or the coordinator desires to inspect the current situation in detail).

To synchronize the state of the EXODUS simulation, EXODUS will alert the clients with *simulation playing* or with *simulation pausing* signal messages. The former message lets the clients know that the time should start moving forward now, however, EXODUS will tell when the actual time moves forward with the time update message. The latter message lets the clients know that no more time steps will be calculated.

3.3.7 Start alarm message

To sound the alarm in EXODUS (see Section 5.3 in Chapter 5 for more details about the alarm), one of the clients will send a *start alarm* signal message to EXODUS. EXODUS will respond with a message detailed in Table 3.7.

Table 3.7: Alarm started message body

Parameter	**Type**	**Description**
Time	Float32	The simulation time when the alarm was started (in seconds).

The *alarm started* message contains a timestamp to let each client know precisely how long the alarm has been running.

3.3.8 Start fire message

To start a fire in EXODUS, one of the clients will send a *start fire* signal message to EXODUS, similarly to the start alarm message. EXODUS will respond to this message with a reply detailed in Table 3.8

Table 3.8: Fire started message body

Parameter	**Type**	**Description**
Time	Float32	The simulation time when the fire was started (in seconds).

The time stamp in the *fire started* message allows each client to accurately draw the current development of smoke from the fire simulation data for the user. As all clients know exactly when the fire started, this allows all clients to stay in sync for the rest of the simulation. The fire and smoke data are not synchronized over the network, because SMARTFIRE CFD fire simulation cannot yet at this point in time run in real-time for the level of precision that is desired from the CFD simulation. Therefore, to save some network bandwidth while sacrificing some flexibility, each client must have the CFD fire simulation data available to be loaded. The loading time is optimised with a custom built BHDV binary file format for this research. The original file format from SMARTFIRE is in text format and would require converting into binary format during loading time, preprocessing the data into binary format so that the data can be directly copied into the memory without conversions enables faster loading times for the clients.

3.3.9 Add agent message

Table 3.9: Add agent message body

Parameter	**Type**	**Description**
Agent Id	Int32	Unique agent Id shared between clients and the server.
X coordinate	Float32	The location (in metres) of the agent along the X-axis (horizontal).
Y coordinate	Float32	The location (in metres) of the agent along the Y-axis (vertical).
Z coordinate	Float32	The location (in metres) of the agent along the Z-axis (horizontal).
Y rotation	Float32	The rotation (in degrees) of the agent along the Y-axis.

Table 3.9 details the *add agent* message that EXODUS sends to all connected clients whenever a new agent is introduced in the EXODUS simulation. The coordinate system used is a Cartesian coordinate system where XZ-plane forms a horizontal plane and Y-axis is perpendicular to the plane, indicating the height above or below the ground plane. The agent is only allowed to rotate around the Y-axis. The rotation starts from positive Z-axis and rotates clockwise towards positive X-axis when oriented towards negative Y-axis. The directional vector $\hat{v}_{xz}$ can be obtained from the angular rotation a_y with Equation 3.1. The directional vector is useful when calculating the agent's velocity. However, the angular rotation is useful in the calculation of rotation for the transformation matrix used by OpenGL.

$$\hat{v}_{xz} = \begin{bmatrix} \sin a_y \\ \cos a_y \end{bmatrix} \tag{3.1}$$

3.3.10 Move agent message

Table 3.10: Move agent message body

Parameter	Type	Description
Agent Id	Int32	Unique agent Id shared between clients and the server.
X	Float32	The location of the agent on X-axis (in metres) at the time set by the Time parameter.
Y	Float32	The location of the agent on Y-axis (in metres) at the time set by the Time parameter.
Z	Float32	The location of the agent on Z-axis (in metres) at the time set by the Time parameter.
Velocity X	Float32	Constant velocity of the agent (in metres/second) on X-axis.
Velocity Y	Float32	Constant velocity of the agent (in metres/second) on Y-axis.
Velocity Z	Float32	Constant velocity of the agent (in metres/second) on Z-axis.
Time	Float32	The exact simulation time (in seconds) when the agent was in X, Y, Z location.
Action Id	Int32	Defines the type of action the agent is performing since the time set by the Time parameter.

To move agents EXODUS informs the clients with the message in Table 3.10. This is only sent when the previous move command is no longer valid i.e. when the agent changes speed, direction or action. A stop command is indicated with zero velocities on all axis. X, Y and Z parameters are related to the origin of the geometry and therefore most of the time Y value is zero on the ground level. Agents in EXODUS are incapable of jumping and therefore in most instances the Y velocity is also constant, except on stairs, escalators and lifts. The message design (in Table 3.10) allows the developer of the IRIS system to design a range of specialised movement actions such as climbing over bodies or climbing out of a window as is necessary by using the *Action Id* parameter.

This *move agent* message is also used by the client to inform EXODUS to move the

user-controlled agent.

3.3.11 Remove agent message

Table 3.11: Remove agent message body

Parameter	Type	Description
Agent Id	Int32	Unique agent Id shared between clients and the server.

The EXODUS server will need to inform clients when an agent has left the simulation, e.g. by exiting via an exit. This is reducing the total number of agents currently in the simulation and does not count towards mortuary results table but is instead added to the exit results table. This is done by using the message detailed in Table 3.11. If the agent is instead killed or incapacitated, the message in Section 3.3.12 is used.

3.3.12 Kill agent message

Table 3.12: Kill agent message body

Parameter	Type	Description
Agent Id	Int32	Unique agent Id shared between clients and the server.
Reason Id	Int32	This is either: 1=Toxicity or 2=Heat. This is different from the removing of agent message as this adds the agent to the mortuary table.
Time	Float32	The exact simulation time (in seconds) when the agent was killed.

If an agent did not exit the bounds of the simulation safely, it can be removed with the *kill agent* message indicated in Table 3.12. This is similar to the *remove agent* message in Section

3.3.11. However, the *kill agent* message moves the agent to the mortuary results table with a reason (i.e. death by heat or toxicity) and time stamp specified in this message.

3.3.13 Add client agent message

Table 3.13: Add client agent message

Parameter	**Type**	**Description**
Request Id	Int32	Unique agent Id shared between clients and the server.
X	Float32	The desired location of the agent on X-axis.
Y	Float32	The desired location of the agent on Y-axis.
Z	Float32	The desired location of the agent on Z-axis.

When a user wants to join the simulation run by EXODUS, the client software will indicate it with the message in Table 3.13. If the values are acceptable EXODUS will confirm this with the message in Section 3.3.14. The *Request Id* does not represent the *Agent Id* in the simulation but is chosen at random by the client. The client will need to remember this to match the request with an *Agent Id* when EXODUS replies. This message can be easily expanded to include other favourable attributes such as the agent's looks, gender, ethnicity, age, weight and height as well as other more advanced capabilities that EXODUS can use such using a hospital bed [148].

3.3.14 Confirm the addition of client agent message

Table 3.14: Confirm the addition of client agent message

Parameter	**Type**	**Description**
Request Id	Int32	Unique agent Id shared between clients and the server.
Agent Id	Int32	Unique agent Id shared between clients and the server.

EXODUS replies to the message in Section 3.3.13 with a confirmation message in Table 3.14. Once this message is received and the *Request Id* matches the one the client sent, the client can now assume that it has control over the agent specified by the *Agent Id*.

3.3.15 Gesture and voice command message

When the user issues a gesture command, this is sent by vrEXODUS to the EXODUS program as listed in Table 3.15. The Section 5.1 provides full details of the gesture implementation. Suitable values for the Type parameter are: 1 = Hand gesture and 2 = Voice command. Suitable values for the Gesture Id parameter are: 1 = Stop, 2 = Go, 3 = Start evacuating, 4 = Get down, 5 = Get up, 6 = Get out of the way, 7 = Evacuate in direction, and 8 = Move Over There. To issue both hand gesture and voice command at the same time: this message is sent twice with the same time parameter value. To apply the command exclusively to a specific agent only, the Target Agent Id must be set to non-zero value. The optional X Target, Y Target and Z Target parameters are used for the Move Over There command.

Table 3.15: Gesture and voice command message

Parameter	**Type**	**Description**
Agent Id	Int32	Unique agent Id shared between clients and the server.
Type	Int32	How the gesture is being issued
Gesture Id	Int32	The command the agent is attempting to issue.
X	Float32	The X coordinate of the issued command location.
Y	Float32	The Y coordinate of the issued command location.
Z	Float32	The Z coordinate of the issued command location.
Y rotation	Float32	The rotation (in degrees) of the agent along the Y-axis.
Time	Float32	Current simulation time (in seconds).
Duration	Float32	How long the command was issued for (in seconds).
Radius	Float32	The maximum reach of the command (in metres).
Close Probability	Float32	The number of agents / second obeying a command at 0 metre distance from the issuer.
Far Probability	Float32	The number of agents / second obeying a command at the maximum radius from the issuer.
Compliance	Float32	Coefficient of how many more times likely an agent is to obey a command when others around them are already obeying the command.
Min Agent Delay	Float32	Minimum amount of time (in seconds) an agent will continue to obey the command.
Max Agent Delay	Float32	Maximum amount of time (in seconds) an agent will continue to obey the command.
Target Agent Id	Int32	Apply this command exclusively to Target Agent Id.
X Target	Float32	Optional X coordinate that agents are to be sent to.
Y Target	Float32	Optional Y coordinate that agents are to be sent to.
Z Target	Float32	Optional Z coordinate that agents are to be sent to.

3.3.16 Update agent facing direction message

Table 3.16: Update agent facing direction message

Parameter	Type	Description
Request Id	Int32	Unique agent Id that is the same between the client and the server.
Y rotation	Float32	The rotation (in degrees) of the agent along the Y-axis.
Time	Float32	The exact simulation time (in seconds) when the agent was facing this direction.

When the user issues a gesture/voice command, EXODUS assumes it keeps issuing it for the total duration indicated in the gesture/voice command message. This means the client needs to keep the server updated on the current rotation of the agent. This also helps the user to see where other user-controlled agents are currently looking at. Therefore, this message (Table 3.16) is sent whenever the user rotates the camera in Y-axis.

3.3.17 Network message optimisations considerations

Ideally for the maximum performance, each network packet should be filled with useful data, this is 1480 bytes for TCP/IP within Ethernet. Fortunately, the socket implementation for Microsoft Windows treats TCP/IP connection as a stream of data and transparently packs the packets in the background. Therefore, the currently used message sizes would need to be optimised instead.

As can be seen from the previous message body definitions the longest network message is 19×4 bytes long and there are fewer than 256 different messages. Thus, the message header could use a single byte to indicate the length of the message and another single byte to indicate the type of the message, instead of using 4 bytes. These savings alone would make the Move agent message 38 bytes long, which is about 86% of the original size (44 bytes), which is the most commonly sent message type during a simulation.

To optimise the *Move agent message* even further, a new bit mask of length 8 can be used after the *Agent Id* parameter. This bit mask would have 1 bit for each of the subsequent parameter, indicating whether or not the parameter is present. If the parameter is absent, the recipient should use a default value, such as 0. As most of the time an agent is travelling on a flat ground and will eventually reach a ground level in a successful building evacuation: this means that on a flat ground level the agent would have zero value in both Y and *Velocity* Y parameters. Also due to the nature of how EXODUS geometries are built in the finite model, the agent movement space is effectively a grid. If the agent is moving only in a horizontal direction the Z velocity would be absent. Also, if the agent is moving only in a vertical direction the X velocity would be absent. A typical walk action could be set as 0 for *Action Id*, this would mean that for all agents that are walking, this parameter could be absent. Using a bit mask, an agent walking in a horizontal or vertical direction on a ground level could be represented with message that is only 19 bytes long, which is only about 43% of the original size with no loss in quality.

Currently the number of actions that are available for the agent are less than 256, which would allow the Action Id to be reduced to a single byte.

Further optimisations would be difficult without sacrificing accuracy. However, some optimisations by sacrificing accuracy with minimal negative impact can be done. The accuracy of the X, Y and Z values can be reduced due to the grid-based nature of a finite EXODUS model. When assuming the spacing is half a metre apart, the X and Z values could be a 2 bytes long integer representing half metre spacing. This would allow the representation of a 32km by 32km area. Similar optimisation could be done for the Y value, reducing its accuracy to 1cm increments with a 2 byte integer value. This would allow the representation of up to 655.36m above the ground level. This covers all other buildings except the single tallest building in the world. The velocities are difficult to reduce in accuracy as all inaccuracies accumulate on each time step. Reducing the velocity parameters to 16-bit floating point values would save half the space.

The time parameter accuracy could also be reduced. EXODUS represent time in time steps, which happen 12 times a second. This could be represented with a 2 byte integer value, making the maximum simulation time 91 minutes. This would obviously not be suitable in

a complex multi-stage scenario where agents are not evacuating immediately.

Using the bit mask with the previously mentioned optimisations, including the reductions in accuracy, would make the *Move agent* message of an agent walking in horizontal or vertical direction on a ground level only 13 bytes long (i.e. the ideal situation). 13 bytes is only about 30% of the original size.

For this research, the network speed was not a bottleneck, therefore, these optimisations were not necessary for the test cases detailed in this thesis. However, it is important to note that the network message optimisations are possible with the few examples provided in this section. For a production quality implementation, where the network speeds are substantially lower (such as a domestic internet connection vs gigabit ethernet LAN), these optimisations could become invaluable.

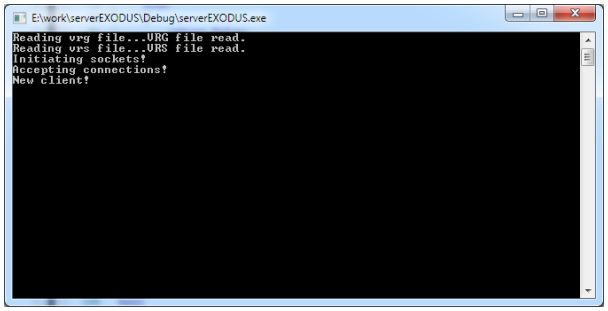
As a benchmark test, EXODUS was able to simulate and move over a memory bus network connection on a localhost over 15,000 agents in real-time in a very simple geometry. However, once 4 or more users were issuing gesture commands simultaneously in a more complex geometry, such as the airport geometry described in Chapter 6, in a simulation size of 600 agents or more, the simulation speed occasionally dropped to below real-time (see more details of this test in Chapter 6). This is due to the way the gesture influence areas are calculated with ray traced visibility checks in 3D space. Until more optimisations are done for EXODUS software, the network messages detailed in this thesis are adequately efficient.

3.4 Client-Server architecture development phases

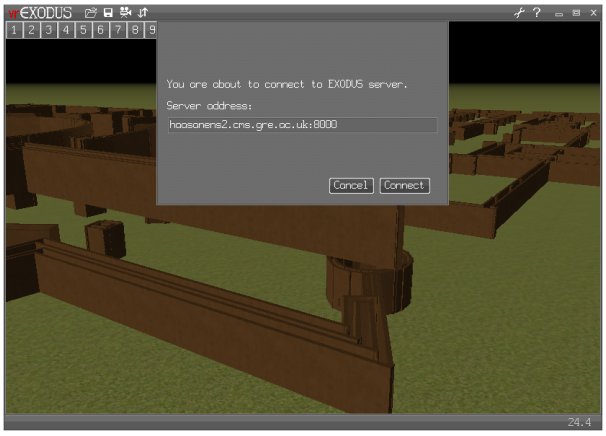
This section details experimentations and earlier versions of the IRIS system. The reasons for abandoning real-time editing and point-and-click functionality is discussed in this section.

EXODUS software is a very complicated piece of software, spanning over twenty of years' worth of development through many operating systems and hardware architectures [149]. Therefore, to speed up the development of the network library implementation, it was necessary to split this into a separate application program.

Knowing that vrEXODUS would be the client in this research, its existing capabilities can be utilised to an advantage. For example, vrEXODUS is capable of loading VRG and VRS files, representing geometry and scene (agent) data respectively. Therefore, the initial version of the network library replicated the loading of VRG and VRS files remotely with a single difference: the VRS (the scene information, such as agent movements) file was sent progressively in real-time playback speed from the server to the client. This can be seen in Figure 3.6.



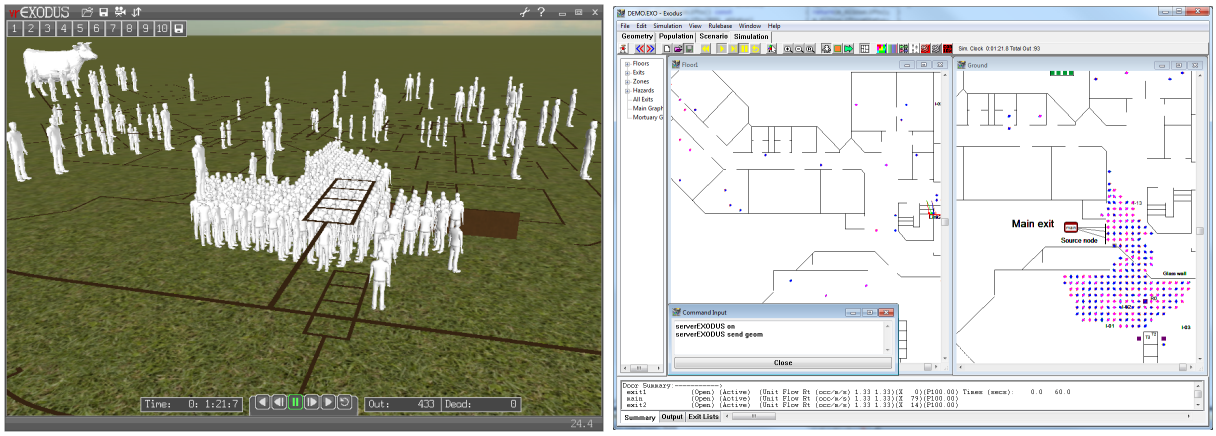
(a) Dummy simulation server mimicking EXODUS as a console application



(b) vrEXODUS connected to the dummy server

Figure 3.6: The first version of the network model implementation

The initial version of the network library implementation enabled the refinement of the network protocol to what it is now, while also allowing to test its robustness in sending different types of messages. However, the client software, the vrEXODUS (Figure 3.6b), could not have any user control over the agents as it is a dummy console application (Figure 3.6a) which does not simulate human behaviour. At this point it was necessary to move the network code to the main EXODUS software application and integrate it with the existing simulation code.



(a) vrEXODUS **(b)** EXODUS

Figure 3.7: vrEXODUS connected to real EXODUS server

Figure 3.7 demonstrates the network library implementation working in EXODUS. Figure 3.7a shows an early version (in 2013) of vrEXODUS with the graphics in an early development phase. The new models were not yet available at this time. The Figure 3.7a also shows a 3D cow model that was used as a test case to verify the correct functionality of the developed Wavefront OBJ file format reader, was also later re-used in the smoke validation tests in Chapter 4.

The early version of the IRIS system allowed the selection of agents and issuing direct orders by pointing and clicking as if each agent was commanded with extreme precision with an instantaneous response over a radio. This utilised the itinerary system of EXODUS by giving individuals or groups of agents new itineraries.

The point and click functionality did not translate very well into the first-person view. The point and click functionality is very useful from a top down view when the ground can be seen even in high crowd densities. When the user-controlled agent is surrounded by other agents, then moving by pushing through the crowd becomes difficult from a first-person perspective. The initial implementation included making the surrounding agents invisible when issuing a movement command so that the user can see the ground where to click to issue a new movement command. This immediately broke the immersion of the system. Then a hybrid solution was developed, which included the point and click functionality with the ability to issue move/push forward commands when in crowded situations. When implementing this it became clear that the point and click functionality would not be suitable for a first-

person camera. Thus, the point and click system was abandoned in favour of issuing direct move forwards, backwards, left and right commands by pressing the assigned buttons on a keyboard or a controller.

The initial network version included the transmission of the geometry over the network (as can be seen the previous Figures 3.6 and 3.7). The early research proposed the idea of being able to modify the geometry and the scenario from the client-side, from vrEXODUS, and the modifications would be transmitted to the EXODUS server. The user would have the ability to add and remove walls, floors and other parts of the geometry. However, these capabilities were already present in EXODUS software. It takes a significant effort to build and modify a complex geometry based on a detailed CAD drawing. To modify the geometry in "real-time" the user would typically pause the simulation, make the modifications and then press continue. Depending on the size of the modifications, the simulation could be paused for minutes at a time. The client side geometry modification capability was abandoned in favour of loading a 3D model of the geometry from a professional 3D modelling (or CAD) software, such as Blender, directly into the VR software. The users can thus benefit from their expertise with existing professional 3D modelling tools and import detailed existing geometries into the IRIS system.

3.5 Summary

Chapter 2 identified gaps in knowledge in previously known systems and the capability requirements for the IRIS system. This chapter proposed the key solution to linking the three software components together to form a single unified system, the IRIS system (see Chapter 1). The required upgrades to the graphics fidelity to advance the state of the art is covered in the next chapter (Chapter 4). The human to simulation interface which includes hand and voice gestures to enable interaction between user-controlled agents and computer-controlled agents are covered in Chapter 5.

This chapter proposed a unique, minimalistic and efficient TCP/IP communication system to enable real-time high performance asynchronous multi-user connection between the linked software components of the IRIS system (i.e. VR clients and evacuation model). The

custom-built asynchronous network communication protocol minimises the negative performance impact on the linked software components using modern multi-core CPU architecture, thereby allowing higher fidelity real-time simulation and rendering of sophisticated scenarios. A custom-built and designed CPU thread model allows efficient non-blocking communication, creation and manipulation of data between network components as well as between CPU threads. The non-blocking model allows real-time updating of graphics with asynchronous simulation updates over a network connection. It also enables real-time graphics updates from local storage, which is covered more in the next chapter (Chapter 4).

Chapter 4

GRAPHICS FIDELITY

As was discussed in Chapter 2 the VR environment in the IRIS system should provide natural information to the user and not provide artificial information, to avoid artificial behaviour and to enable natural behaviour within the simulated virtual world. A review of the current state of the vrEXODUS software that is used in this research to provide the VR environment for the IRIS system can be found in Appendix F. The review highlighted the need to improve the graphical fidelity of vrEXODUS. This chapter discusses the necessary graphical enhancements to enable an immersive VR environment for the IRIS system, such as higher fidelity textured 3D models (Section 4.2) and an accurate visualisation of smoke visibility. To enable higher graphical fidelity in a real-time virtual world software program, some performance enhancements are necessary, and these are discussed in this chapter. The first section discusses the importance of 3D model importing capabilities and the need for a graphics library upgrade.

Chapter 2 identified the current understanding on how humans behave in fire (see Section 2.2). One of the key aspects in the behaviour is visibility and navigation in smoke. Accurate smoke visibility rendering has been used in ray-traced applications; however, it has not been possible to use accurate smoke visibility rendering in real-time applications (see Section 2.2.1). Furthermore, some real-time applications have been using normalised soot density values as an indication of visibility in smoke, which render inaccurate results (see Section 2.2.2). A novel smoke rendering technique is proposed to enable efficient and accurate smoke visibility rendering in real-time in Sections 4.3 to 4.6. This is the second research contribution in this thesis as described in Section 1.3.2, Chapter 1 and is further detailed in this chapter.

A novel test benchmark is proposed in Section 4.7 for the validation of rasterised smoke visibility in a real-time virtual environment. This test compares the virtual results to Jin's experiment from 1970 [18] and validates the smoke rendering algorithm proposed in this Chapter.

Finally, in Section 4.8 a demonstration case of an aircraft fire shows how the new smoke rendering technique discussed in this chapter can be utilised to visualise a complex fire scenario from a CFD fire model. The demonstration case is also used to highlight the importance of using high fidelity rendering of smoke to immerse users (i.e. a low-quality smoke changes user behaviour).

4.1 Upgrade from Direct3D 9 to OpenGL

The latest version of vrEXODUS was written to use Microsoft Direct3D 9 for Windows XP (more details on the earlier developments of vrEXODUS are given in Appendix F). However, Microsoft abandoned the Direct3D 9 API structure in the next version of Direct3D (version 10) and in later versions [150]. To upgrade the vrEXODUS software program to a newer version, the graphics software code must be rewritten to use the new structure of any of the newer versions of Direct3D.

To avoid repeating the problem with abandoned APIs the new graphics library for the new version of the vrEXODUS was carefully considered. At the beginning of this research in 2011, Microsoft had already released Windows Vista and Windows 7 operating systems, which used newer Direct3D versions in comparison to the Windows XP version. However, there was no guarantee that the Direct3D version supplied with Windows 7 would not be abandoned with the next version of Windows. Therefore, the new version of vrEXODUS will use a different hardware accelerated graphics library namely OpenGL. OpenGL provides backwards compatibility of the API all the way to the version 1.0 [151] released in 1992 and is not locked to a specific version of Windows, or Microsoft products. This change in the graphics library enables the IRIS system to be ported to other platforms such as GNU/Linux and Mac OSX, if necessary. OpenGL also provides forwards compatibility, meaning new capabilities from newer versions can be utilised in future research without rewriting all the

graphics code.

Once vrEXODUS was successfully rewritten for OpenGL version 3.0+, it became possible to take advantage of some of the advanced features available for OpenGL, such as combining multiple graphics card calls together using display lists [152]. This allowed combining most of the graphics card calls for the static geometry into few calls i.e. in a case of 1.7 million different size walls were combined from individual graphics card calls per single wall to a single graphics card call for every 50,000 walls, resulting in over 100 times speed up in complex geometries. Direct3D version 9 does not have this capability. Rendering multiple 3D models with display lists do not require complex vertex array manipulations with the sorting of materials (such as textures and colours) [152], greatly reducing the development time while enabling fast hardware acceleration.

Valve reported in August 2012 that their Left 4 Dead 2 game runs faster on OpenGL than on Direct3D on Windows 7 and Linux [153] due to a smaller overhead on the graphics library implementation. This supported the decision to use OpenGL at the beginning of this research. However, a new graphics library called Vulkan was released by Khronos Group [136] in 2015 to supersede OpenGL, which may become a necessary upgrade in a future version of the IRIS system (see Chapter 7 for future work).

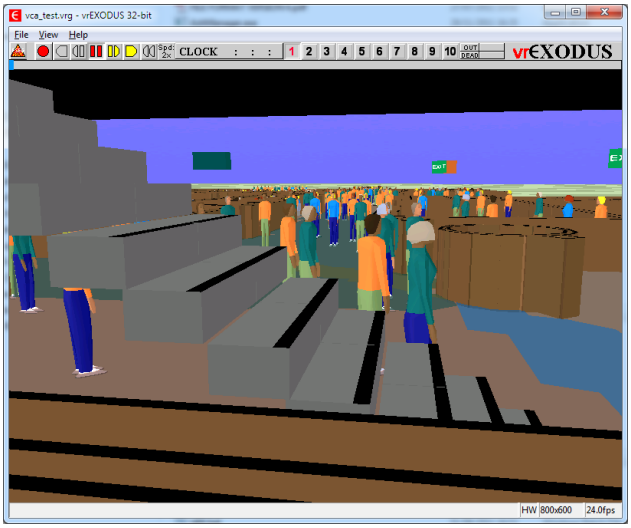
4.2 New 3D models

The current version of vrEXODUS uses the same Superscape VRT graphics created by Cooney in 2000 [154] (more details on the earlier developments of vrEXODUS are given in Appendix F). A screenshot of these are provided in 4.1a. The 3D models lack any texturing which is expected in modern 3D graphics.

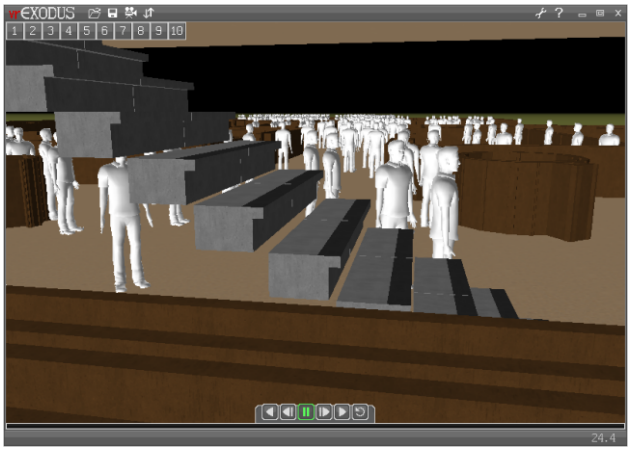
To avoid spending invaluable research time drawing 3D models and to enable easy future upgradeability, these graphics are replaced by off the shelf 3D models supplied by FSEG. The initial version of implementing textured objects such as floors, wall and stairs can be seen in Figure 4.1b. This version also included an initial implementation of reading Wavefront OBJ file format (the white 3D agent models as seen in Figure 4.1b).

Due to feedback from the users of the IRIS system the colour palette was changed from the Superscape VRT colour palette (Figure 4.1b) to what can be seen in Figure 4.1c. This Figure also shows the completed implementation of Wavefront OBJ file format with supporting materials.

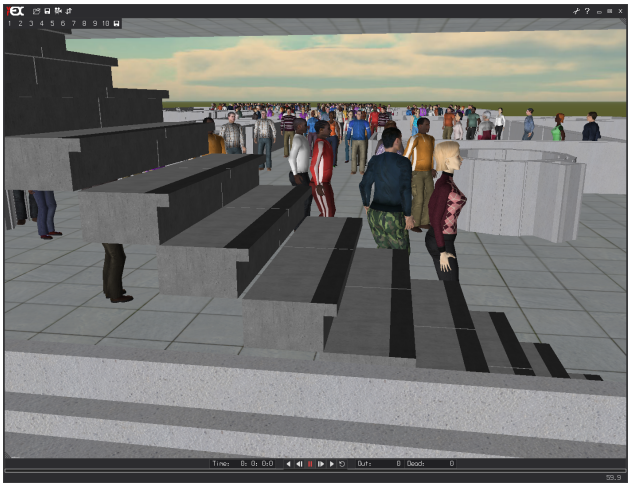
The set of 3D agent models provided by



(a) Current vrEXODUS graphics



(b) New vrEXODUS graphics, step 1



(c) New vrEXODUS graphics, step 2

Figure 4.1: Graphical improvement steps

FSEG is split into 2 different 3D models, a female and a male mesh. Reusing the same mesh saves memory and reduces loading time. To increase variation between the models a set of 20 textures were provided per model resulting in a total of 40 unique looking 3D agent models.



Figure 4.2: New 3D agent models

A screenshot of the 40 models in a crowd size of 540 agents can be seen in Figure 4.2. The screenshot shows a good variety of models, it is not immediately obvious that there are only two different 3D meshes and most of the variation comes purely from unique textures. For this research, these are the models that are available and provided by FSEG. Due to the tools that were built as part of this research, new models (meshes + textures) can be added quickly and when necessary, for example by splitting textures and varying texture colours. However, when adding more models and animations, they come with the expense of longer loading times and higher memory requirements.

To get the most performance out of the new models, the animations were pre-calculated in a 3D modelling software and exported as a flipbook animation of still models. The flip-book animation removes the skeleton animation calculations done on GPU. This is a trade-off of sacrificing memory to gain performance. As a result of having fluent 60 frames per second flip book animations, while each purchased model consists of about 20,000 vertices, a single one second animation results into about 1,200,000 vertices. Some animations are significantly longer than 1 second, such as a standing still animation where the agent is swaying and appears "breathing" ever so slightly to give a human like appearance.

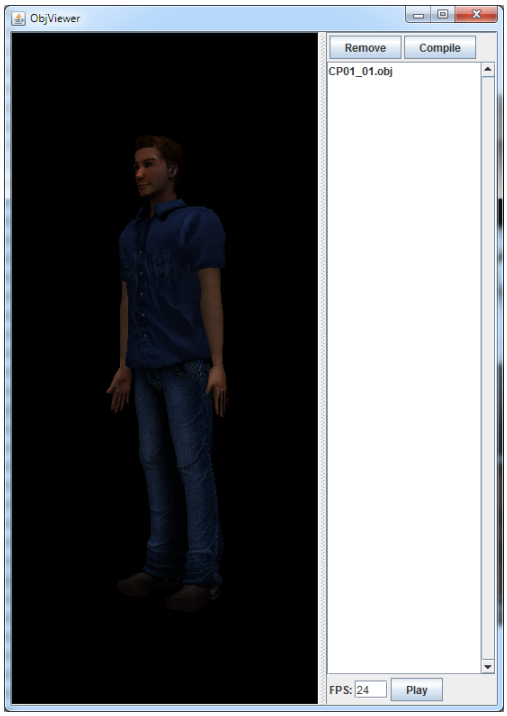


Figure 4.3: Animation flipbook compiler

From the animations it is very clear that some vertices may not move at all or move only very slightly from one frame to another. This presented an opportunity to write an optimiser auxiliary tool to merge the flipbook animation files of 3D models into a single file.

Figure 4.3 shows what the optimiser auxiliary tool program looks like, to optimise the flipbook animation files of 3D models. It has currently a single frame loaded, which is indicated on the list at the right-hand side. It allows the user to preview the animation at desired frames per second (FPS) as well as remove any unwanted frames and when satisfied with the animation, the user can compile the whole animation into a single file. This compilation can take a significant amount of time, which in the case of about 10 seconds long standing still animation was a few hours. However, the overall memory cut of this particular animation was over 90% purely by the removal of duplicate vertices by truncating the data to 32-bit floating point from 64-bit and performing the comparison in 32-bit floating point values.

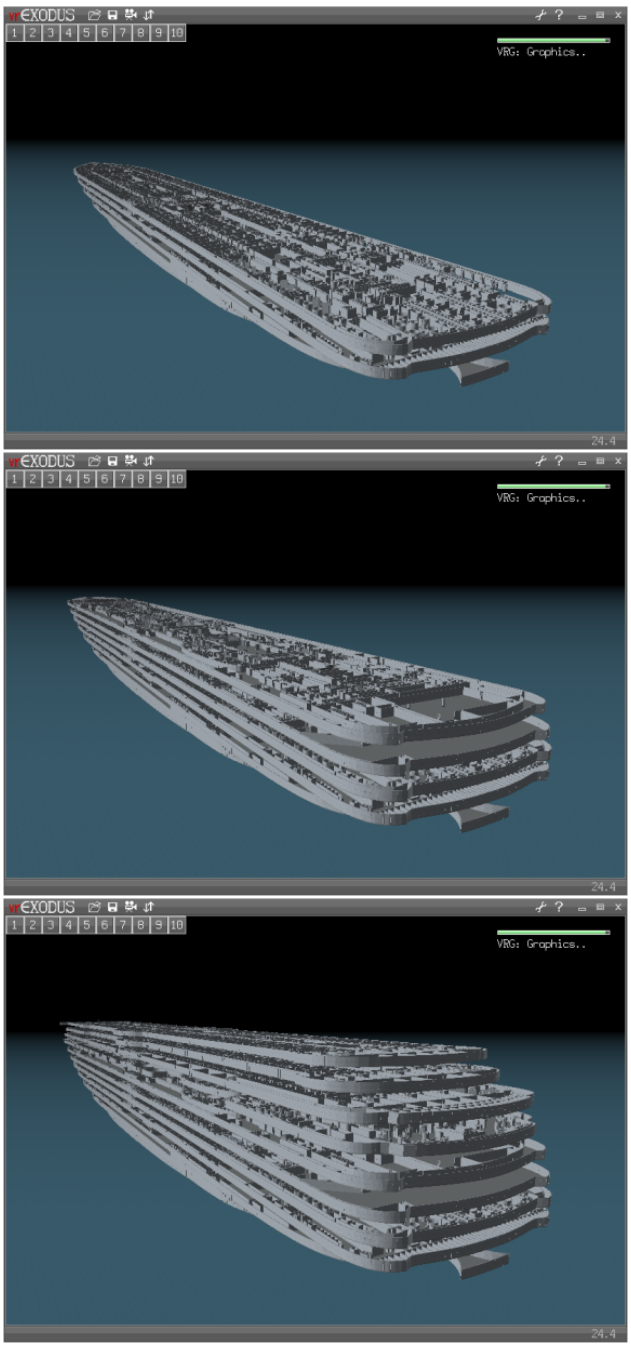


Figure 4.4: Parallel visualisation and updating of graphics data from multiple threads with a shared OpenGL context.

Due to the large sizes of some geometries (i.e. with over 2 million walls) the loading times can take many minutes. Sharing the OpenGL context between multiple threads as was discussed in Section 3.1.1 allows the 3D geometry (and models) to be displayed to the user while it is loading (see Figure 4.4). This allows the user to inspect the geometry while it is loading and abort the loading if necessary.

4.3 Translucent blending equations for smoke rendering

It is important to distinguish some key differences between the multiplicative and additive blending to establish the reason why the default additive blending is unsuitable for the smoke rendering described in this chapter, and the proposed solution requires the use of multiplicative blending. The final equation can be found in Section 4.6.

Rendering smoke requires blending of semi-transparent (i.e. translucent) objects with opaque objects. Depending on the algorithm used, the translucent objects may need to be sorted in a certain order to achieve correct results. Further self-overlapping objects may need to be split into non-overlapping objects to satisfy the sorting requirement. Other techniques include per-pixel sorting, such as sorting linked-lists of overlapping pixels [155], or per-pixel splitting, such as depth peeling rendering one layer at a time [156, 157] OpenGL SuperBible specifies an algorithm to combat the time-consuming sorting requirement in an order independent fashion [158]. It defines using a multiplicative blending instead of additive blending. This saves a lot of computing time by removing the sorting of objects but adds complications and specific requirements. To describe how the multiplicative blending differs from the default [159] additive blending technique and how this is necessary for accurate smoke rendering: some definitions (such as colour vector and blending equations) are necessary. The definitions set in this section will also be used in the later sections in this chapter.

Before the blend functions and the smoke rendering algorithm can be discussed, it is necessary to define the data structure for colour and colour channels. The colour attribute is defined as a vector $\vec{C}$ in Equation 4.1. The colour vector specifies 4 colour channels c_R, c_G, c_B, c_A as red, green, blue and alpha channels respectively [160].

$$\vec{C} = \begin{bmatrix} c_R \\ c_G \\ c_B \\ c_A \end{bmatrix}, \text{ where } c \in \mathbb{R}, 0 \leqslant c \leqslant 1 \tag{4.1}$$

OpenGL specification also provides a set of default [161] values for the colour vector $\vec{C}$. The default values have been converted into the colour vector format (see Equation 4.1) and summarised in Equation 4.2. The $\vec{C}_0$ sets all colour channels to zero (minimum) value, where the opposite $\vec{C}_1$ sets all colour channels to the value of one (maximum). $\vec{C}_c$ sets all channels to the value of c. $\vec{C}_{(c,c,c,1)}$ sets all colour channels to the value of c with exception of the alpha channel, which is set to the maximum value of one.

$$\vec{C}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \vec{C}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad \vec{C}_c = \begin{bmatrix} c \\ c \\ c \\ c \end{bmatrix} \qquad \vec{C}_{(c,c,c,1)} = \begin{bmatrix} c \\ c \\ c \\ 1 \end{bmatrix} \tag{4.2}$$

The OpenGL specification [161] sets strict rules on the combination of colour vectors that can be used in the blend function. Based on the colour data structure and the default values that were defined in Equations 4.1 and 4.2 the possible colour vector combinations used by the OpenGL blending function can be defined. The combination rules are denoted by $\vec{C}_k$ in Equation 4.3. In this section and in the future sections in this thesis i indicates the current value used in the equation and $(i+1)$ indicates the new value to replace the i value in the computer memory and the value to be used in the subsequent equations.

$$\vec{C}_k := \begin{cases} \vec{C}_i & \vec{C}_i \in \{\vec{C}_s, \vec{C}_d, \vec{C}_{c_{(s,A)}}, \vec{C}_{c_{(d,A)}}, \vec{C}_{c_A}, \vec{C}_{c_C}, \vec{C}_0, \vec{C}_1\} \\ \vec{C}_1 - \vec{C}_i & \vec{C}_i \in \{\vec{C}_s, \vec{C}_d, \vec{C}_{c_{(s,A)}}, \vec{C}_{c_{(d,A)}}, \vec{C}_{c_A}, \vec{C}_{c_C}\} \\ \vec{C}_{(c,c,c,1)} & \end{cases} \tag{4.3}$$

A single blend function from the OpenGL specification [161] can be defined in the colour vector format (see Equation 4.1) and is denoted by $B(\vec{C})$ in Equation 4.4.

$$B(\vec{C}_{(k,1)}, \vec{C}_{(k,2)}) = \begin{bmatrix} c_{(k,1,R)} c_{(k,2,R)} \\ c_{(k,1,G)} c_{(k,2,G)} \\ c_{(k,1,B)} c_{(k,2,B)} \\ c_{(k,1,A)} c_{(k,2,A)} \end{bmatrix} \tag{4.4}$$

The blend equation used in the OpenGL blending pipeline [159] uses the blend functions (defined in Equation 4.4) to determine the final colour from the source colour and the destination colour values. The blend equation in colour vector format (see Equation 4.1) is denoted by $\vec{C}_{(d,i+1)}$ in Equation 4.5. The blend equation $\vec{C}_{(d,i+1)}$ combines two blend functions $B(\vec{C})$ in three possible combinations [159].

$$\vec{C}_{(d,i+1)} := \begin{cases} B(\vec{C}_{(k,1)}, \vec{C}_{(s,i)}) + B(\vec{C}_{(k,2)}, \vec{C}_{(d,i)}) \\ B(\vec{C}_{(k,1)}, \vec{C}_{(s,i)}) - B(\vec{C}_{(k,2)}, \vec{C}_{(d,i)}) \\ B(\vec{C}_{(k,2)}, \vec{C}_{(d,i)}) - B(\vec{C}_{(k,1)}, \vec{C}_{(s,i)}) \end{cases} \tag{4.5}$$

As was discussed before, the additive blending [158] is the default [159] blending operation set by OpenGL if no parameters are provided. Additive blending is defined in colour vector data format (see Equation 4.1) in Equation 4.6. In additive blending the rendering order matters as the final colour is the accumulation of all the translucent colours blended on top of each other additively, but by also reducing the value of the previous colour (i.e. the colour already at the destination) $\vec{C}_{(d,i)}$ each time. Therefore, the order in which the previous colour $\vec{C}_{(d,i)}$ values were reduced determines the value of the accumulated new colour $\vec{C}_{(d,i+1)}$ if more than one translucent object is blended on top of another. In additive blending (see Equation 4.6) the blended value of the source colour $\vec{C}_{(s,i)}$ is scaled by the alpha colour channel $c_{(s,A)}$ value of the same source colour. However, the destination colour $\vec{C}_{(d,i)}$ is also scaled by the alpha colour channel $c_{(s,A)}$ of the source colour, but in inverse: $(1 - c_{(s,A)})$.

$$\vec{C}_{(d,i+1)} = B(\vec{C}_{c_{(s,A)}}, \vec{C}_{(s,i)}) + B((\vec{C}_1 - \vec{C}_{c_{(s,A)}}), \vec{C}_{(d,i)}) \tag{4.6}$$

The multiplicative [158] blending can be defined in the colour vector format in Equation 4.7).

$$\vec{C}_{(d,i+1)} = B(\vec{C}_{(s,i)}, \vec{C}_{(d,i)}) \tag{4.7}$$

If the additive blending is used repeatedly with the same translucent object, the final des-

tination colour will eventually become the same colour as the translucent object, due to the discrete memory space and the limited amount of memory on computer graphics hardware. However, if the multiplicative blending is used repeatedly with the same translucent object, the final colour will be black, due to the same hardware limitations with discrete memory space and the amount of memory. The difference in the blending effects between additive and multiplicative blending can be described as: the additive blending displays the translucent objects as if they were all independently lit (i.e. illuminating), whereas the multiplicative blending displays the translucent objects as if they are all unlit (i.e. not illuminating) and the light is reflected from the opaque object behind the translucent objects. Therefore, the multiplicative blending is a better fit than additive blending for rendering visibility and light obscuring smoke. The calibration and the accuracy of the multiplicative blending is further discussed in Section 4.6.

4.4 Multi-pass smoke rendering phases

Before the final rendering equation is established in Section 4.6, it is important to describe the multi-pass process of rendering translucent smoke as proposed in this thesis. Section 4.3 established the need to use the multiplicative blending. This section proposes the necessary multi-pass smoke rendering steps in a higher level before they are discussed further in a lower level in Sections 4.5 and 4.6

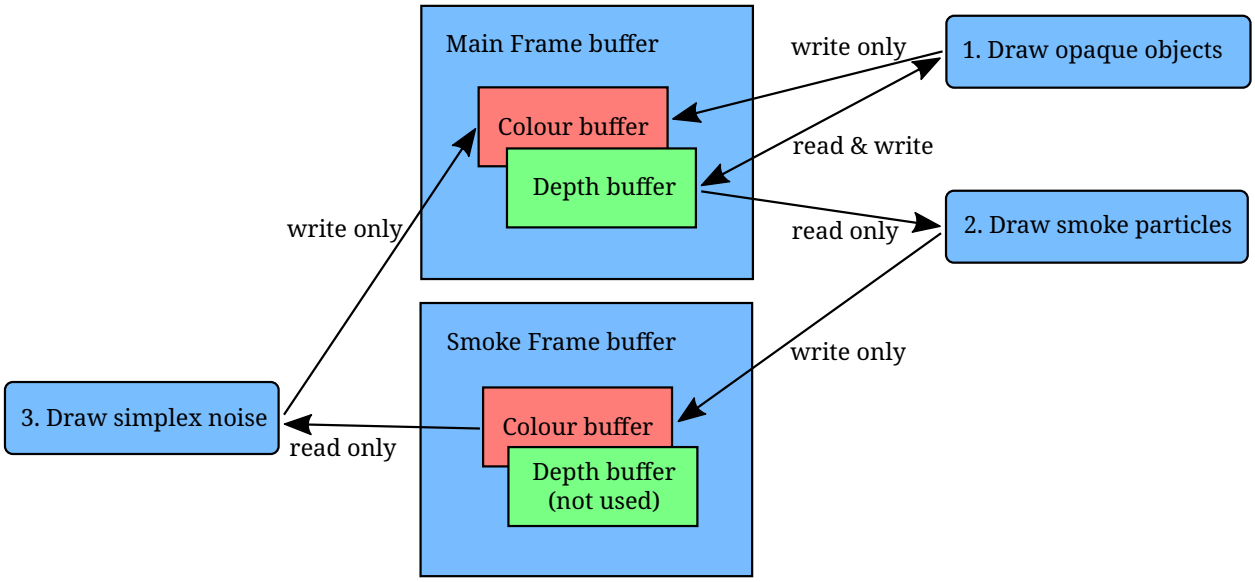


Figure 4.5: Multi-pass smoke rendering

Figure 4.5 demonstrates how the smoke particles are rendered in the scene. All the opaque objects are rendered first in step 1 in Figure 4.5. The rasters of the opaque objects are stored in the *Colour buffer* of the *Main Frame buffer*. Also, the distances from the camera are stored in the *Depth buffer* of the same frame buffer. It is important to render all the opaque objects first as the translucent smoke particles will not update the *Depth buffer* as can be seen in Figure 4.5. If any opaque objects would be rendered after the smoke particles are rendered, assuming some parts of the opaque object's raster would pass the depth test, the pixels of the opaque object would replace the pixels of the smoke.

The smoke particles are rendered in step 2 (i.e. *Draw smoke particles*) in Figure 4.5. The smoke particles will perform a depth test with the opaque objects. The part of the raster that passes the depth test will be blended with the destination image with the multiplicative blending (see Equation 4.7) onto a separate frame buffer. The depth value of the raster is discarded. The initial colour value of the *Smoke Frame* buffer must be white (i.e.$\vec{C}_1$) for this algorithm to produce correct results, or in the case of only using a single colour channel, the initial value of this channel (i.e. c_R) must be 1. Once all the smoke particles are rendered, the *Smoke frame buffer* would be a greyscale image. However, if only a single colour channel such as red c_R is used, the image instead results into a monochromatic image. The chosen colour channel is used in step 3 (i.e. *Draw simplex noise*) in Figure 4.5, where the smoke is coloured and the final image blended back to the *Main Frame buffer* on top of the opaque

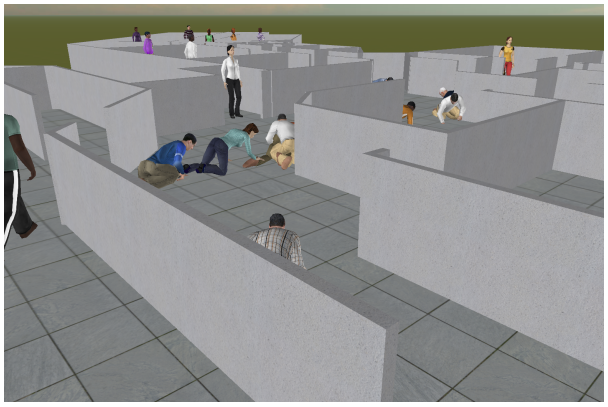
render, using additive blending (see Equation 4.6).

In step 3 (i.e. *Draw simplex noise*) in Figure 4.5, the *Colour buffer* from the *Smoke Frame buffer* is read. As the image is monochrome, the previously used colour channel (i.e. red c_R) must be used. The chosen colour channel (red c_R) is treated as an alpha value c_A when the simplex noise N (see Equation 4.16 in Section 4.6) is drawn on the *Main Frame buffer*. The 2D simplex noise N_{uv} image can be animated when time t is used as the 3rd dimension resulting in N_{uvt} (see Equation 4.16 in Section 4.6). A greyscale render of the simplex noise is used, but it could be set as any colour.

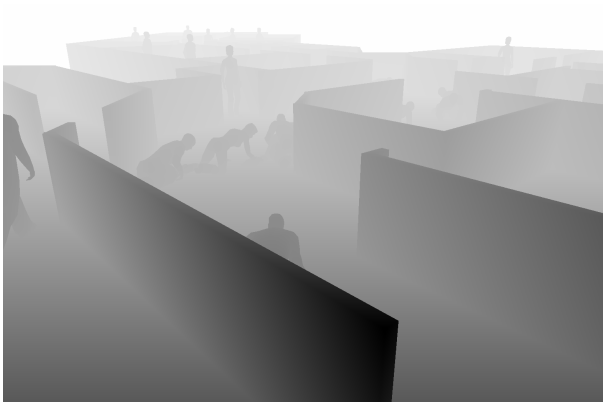
The end result is semi-transparent smoke that: is non-uniformally greyscale, contains a smoke looking texture, is animated over time, is volumetric and efficient (i.e. no sorting required). To make the smoke rendering accurate, some calibration is necessary, which is discussed in Section 4.6.

The main limitation of the multi-pass smoke rendering approach proposed in this section is that it is not influenced by lighting. If the environment is well lit or not lit at all, the smoke looks the same. However, as EXODUS does not implement the concept of light, this is an unavoidable limitation that does not limit the underlying simulation. The main advantage is that this can be calibrated to look correct (see Section 4.6) without having to sort the translucent particles and thus saves significant computing time (i.e. eliminates the sorting of particles). As a result, it enables rendering much more complex smoke data with finer granularity of particles.

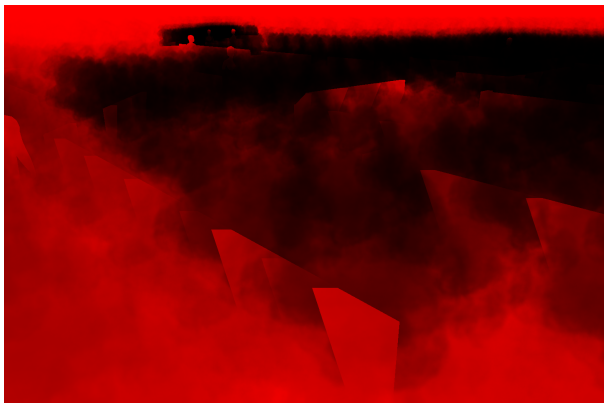
The multi-pass smoke rendering steps from Figure 4.5 are visualised in a real use case in Figure 4.6.



(a) Step 1: Opaque objects in Main Frame Colour buffer



(b) Step 1: Opaque objects in Main Frame Depth buffer



(c) Step 2: Smoke particles in Smoke Frame Colour buffer. Note: only the red colour channel is used resulting in a monochromatic image.



(d) Step 3: Coloured smoke in Main Frame Colour buffer

Figure 4.6: Combining the visualisation of heat and toxic products on agents with the multi-pass smoke visibility visualisation.

In Figure 4.6, it can be seen that the heat and toxic products from the smoke and fire impact the behaviour of the agents. Combining the visual representation of the impact of toxic products and heat on agents (see Figure 4.6a) and the visual representation of smoke visibility Figure 4.6d gives the user of the IRIS system a more realistic view and understanding of the situation. Figure 4.6d shows billboard clipping. From literature, various techniques exist to reduce, remove or even avoid billboard clipping, such as spherical billboards [162], displacement-mapped billboards [163], impostor layers [164–166], simple weighting function between the two depths [167], slicing with lookup tables [168], random displacement and blurring using a fractal cube [169], and metaball textures [170].

4.5 Smoke particle texture animation

To expand Section 4.4, it important to note that smoke is not rendered as a volumetric voxel grid (see Chapter 2), but split into multiple evenly spaced out individual smoke particles (see Section 4.6). How the rendered smoke particle texture is generated from a smoke animation is covered in this section.

The smoke animation is rendered from a 3D modelling application called Blender[1], which uses a CFD simulation by Kim et al. [171]. The total length of the animation is 600 frames, which is 10 seconds long when rendered at 60 frames per second. 60 frames per second was chosen as it matches the refresh rate of the used computer monitor.

Figure 4.7 shows a side view of the CFD simulation of the smoke. It has a finite space and starts just below the centre point. The camera is set above the smoke to the desired effect of smoke coming towards the camera due to it rising. In the final animation, this will give the appearance in the VR that the smoke is travelling towards the camera. In first-person perspective, when not immersed within the smoke, the smoke is likely to move towards the camera as the smoke spreads to areas where there is no smoke, filling the available space. This animation is generated for indoor use and not applicable for outside applications, because the smoke may not spread towards the first-person camera in an outside environment as it has room to spread upwards. If the IRIS system is used with an outside scenario, the smoke animation should be updated to more accurately represent the simulated smoke.

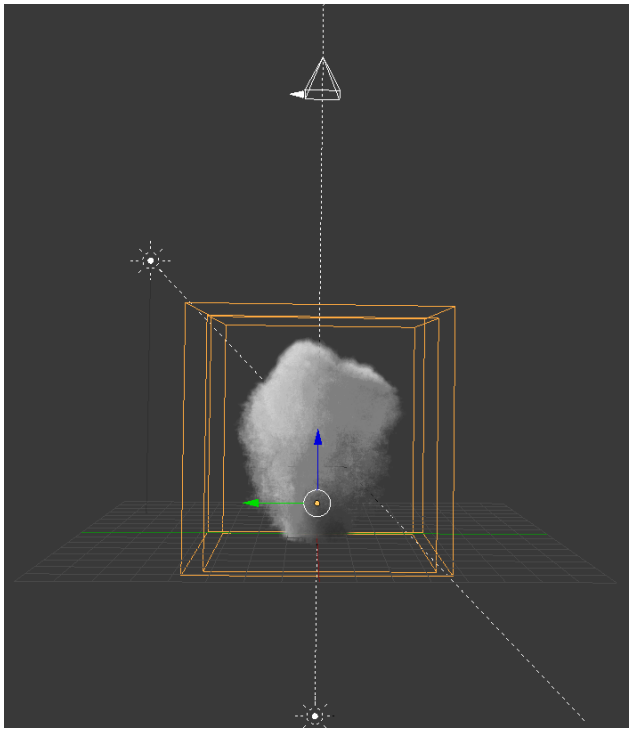


Figure 4.7: Smoke CFD and rendering in Blender

[1]Blender is freely available from `https://www.blender.org/`

The smoke is also lit by two lights from different directions. The first light is coming from above the camera, and from slightly behind it. In Figure 4.7 the top part of the camera is indicated by an arrow. The first light gives the effect of a ceiling light that is above the user and towards the smoke from the user's point of view. The second light is behind the smoke. This aims to give the appearance that there is some light on the other side of the smoke. These lights aim to compensate for the lighting limitations in EXODUS.

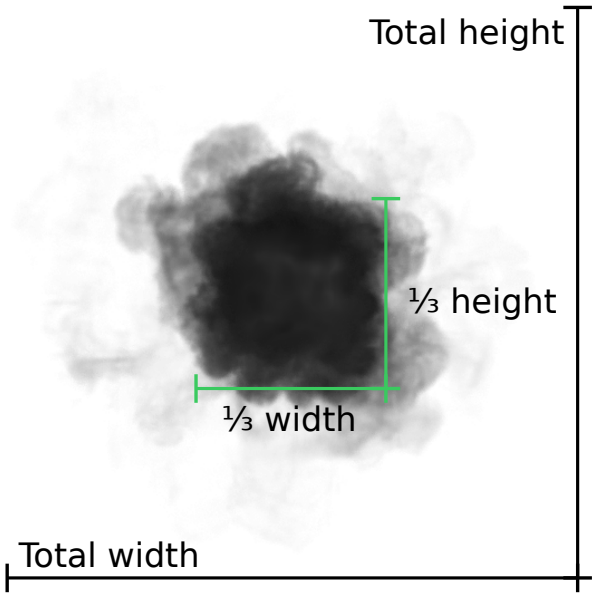


Figure 4.8: The inverted texture of smoke particle at frame 300 of the animation. The darkest area is roughly about a 3^{rd} in width and in height of the total width and height.

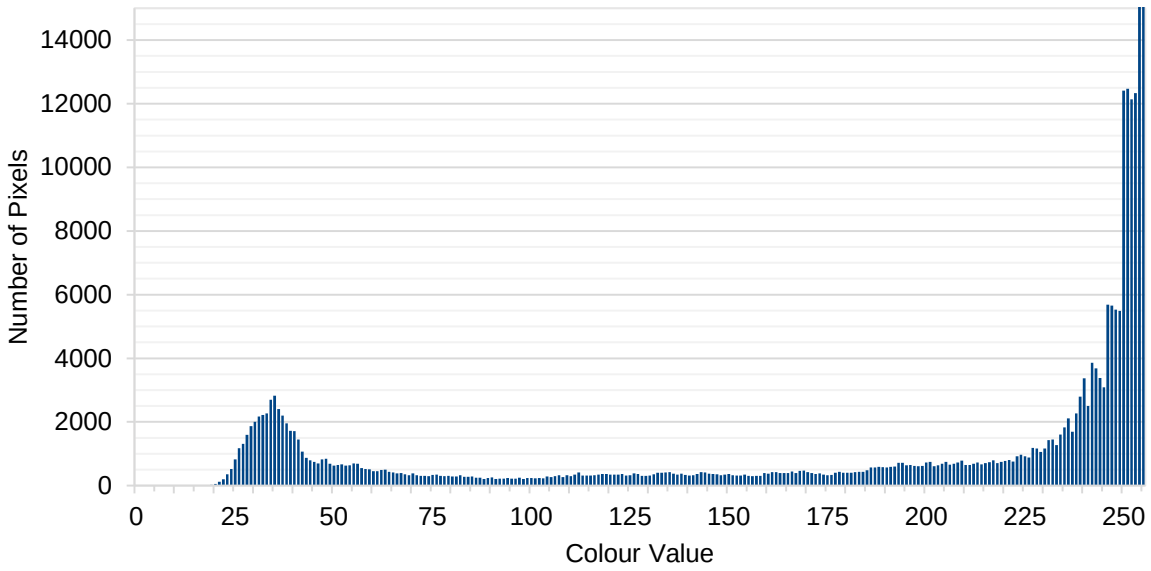


Figure 4.9: The texture histogram. Shows the number of pixels representing each colour value. Almost 10% of the colour (below 25) is missing (i.e. not utilised). The white colour count, values 254 and 255, are 146,985 and 146,726 respectively and off the chart to keep the scale of the chart usable.

Figure 4.8 shows a smoke particle number 300 out of 600. The image is inverted in colour to be suitable for the multiplicative blending, which is described in Sections 4.3 and 4.4. As can be seen in Figure 4.8 the majority of the smoke is located in the centre of the image in an almost round or square pattern. Having majority of the smoke in the centre is desirable as it means there is not a lot of overlap when the particles are placed next to each other.

The histogram in Figure 4.9 is of the same smoke particle found in Figure 4.8. The histogram was produced by the image editing software called GIMP[2]. The histogram shows that the darkest part of the smoke is roughly at the 10^{th} percentile (24/255). The histogram value is compensated (i.e. the colour values are scaled and clamped to 1 and 0) for in the Equation 4.12.

To reuse the smoke particle animation in all the rendered smoke particles without making the smoke look artificially uniform, it is necessary to begin the animation for each particle from a different part of the animation. Therefore, a random starting location for the animation is assigned for each rendered smoke particle. However, the smoke animation has a

[2]GIMP is freely available from `https://www.gimp.org/`

finite length and must be seamlessly looped to enable continuous animation of smoke. In order to enable the smoke to loop seamlessly from the end of the animation to the beginning, careful consideration must be made to choose the beginning and the end points for the loop. It was necessary to remove the first ≈ 200 frames from the animation until a suitable smoke movement was found, that appeared similar to the smoke movement at the end of the animation. Once the suitable transition location was found the last 20 frames were blended progressively to the beginning of the animation.

The flip book animation of smoke covered in this section produces a realistic looking animation, but at the expense of having to store hundreds of texture images in memory. A significantly more memory efficient, but also less realistic looking (i.e. not CFD simulation based) smoke particles can be implemented using the approach in Diablo 3 game as presented in GDC 2013 by Love [172].

4.6 Calibrating the smoke rendering

Chapter 2 established the lack of accurate real-time smoke rendering with correct visibility representation in published research. This section proposes how the multi-pass smoke rendering technique described in Sections 4.3, 4.4 and 4.5 can be calibrated against the published research on smoke visibility rendering. As discussed in Chapter 2 the literature provides some examples on ray-traced smoke visibility rendering from fire data produced by a CFD fire model [24, 90]. The approach proposed in this section is adapted from the published ray-tracing algorithm by Kang [24, 25]. The ray-tracing algorithm fits accurately with the order independent transparency described in Section 4.3 enabling the algorithm to be rendered efficiently. The transparency of the smoke is based on a light extinction coefficient (typically noted as K or α depending on who wrote the paper). The light extinction coefficient used by Kang is based on Jin's work from 1970 [18] and 1971 [19]. Jin's smoke visibility formula is the only formula, since there is no other and it is the one specified by the SFPE handbook of Fire Protection Engineering [21].

To determine the visibility impact of smoke on the light that travels through it, Kang

provides the following equation for ray-tracing algorithms, Bouguer's law. [24]:

$$I/I_0 = exp(-\int_s \alpha(s)\mathrm{d}s) \tag{4.8}$$

The light extinction coefficient of the smoke is denoted by α. The distance within the smoke (in metres) that the light extinction coefficient α applies to is denoted by s. The variable I is the light intensity once through the smoke and I_0 is the initial light intensity before entering the smoke. The integral in Equation 4.8 is applicable to continuous space, however, Kang provided a technique to approximate this indefinite integral in Equation 4.9 [24]. The approximation is used to represent the continuous space in discrete steps in computer graphics when using billboard particles.

$$I/I_0 = exp(-\sum_i \alpha_i s_i) \tag{4.9}$$

If the initial light intensity I_0 is set as 1, then I is expected to be between $1 > I > 0$ and never reaching 0 or 1. In this case I can be used as a percentage value (a coefficient) of how much light is left of the original value. The graphics processing unit used in this smoke rendering research has an 8 bits b per colour channel c thus limiting the opacity to be in 256 or 2^b discrete steps. Lowest opacity value, before hitting zero, is thereby $\frac{1}{255}$ or $\frac{1}{2^b-1}$.

The smoke particle texture (as described in Section 4.5) can be rendered multiple times per one-metre distance. Therefore, depending on the length of the uniform smoke area, multiple particles will likely end up being rendered in the same screen space in front of and behind other smoke particles from the camera's perspective. Instead of calculating the sum of all the various light extinction coefficients as $\sum_i \alpha_i s_i$, the sum is broken into individual iterative steps with n_p representing the number of particles per metre.

The following substitutions were used to map Kang's [24] equations to the equations presented in this thesis:

$$c_{(A,n_i)} = I/I_0 \qquad a = \alpha \tag{4.10}$$

In the smoke rendering the red colour channel (c_R) is used to store the opacity value of the smoke while being stored temporarily in the *Smoke Frame buffer* (see Section 4.4). Thereby the $c_{(R,t)}$ represents the opacity value in the *Smoke Frame buffer* when stored as a texture data type inside the memory of the graphics processing unit.

Thus, Equation 4.9 becomes:

$$c_{(R,t)} = exp(\frac{-a}{n_p}) \tag{4.11}$$

The histogram compensation described in Section 4.5 and shown in Figure 4.9 has been compensated with the following equation:

$$c_{(R,t,i+1)} = 1 - \frac{(c_{(R,t,i)} - 0.1)}{0.9} \tag{4.12}$$

In step 2 of Figure 4.5, multiplicative blending (Equation 4.7) is used to render the inverse smoke particles in Figure 4.8 against a white background (or a maximum value in the monochrome channel) as described in Section 4.4. The resulting multiplicative equation in step 2 of Figure 4.5 can be represented with Equation 4.13.

$$c_{(R,d,i+1)} = c_{(R,d,i)}(c_{(R,t)}c_{(A,s)} + (1.0 - c_{(A,s)})) \tag{4.13}$$

As was discovered by the histogram in Section 4.5, the generated smoke particles were not fully black as is required for accurate smoke visibility representation. To compensate for the low black levels in the histogram (Figure 4.9) the histogram compensation Equation 4.12 needs to substitute $c_{(R,t)}$ in Equation 4.13. The substitution results in the final Equation 4.14 for step 2 in Figure 4.5.

$$c_{(R,d,i+1)} = c_{(R,d,i)}(\frac{c_{(R,t)} - 0.1}{0.9}c_{(A,s)} + 1 - c_{(A,s)}) \tag{4.14}$$

Equation 4.14 is computationally minimalistic, which is beneficial for performance and

enables rendering more smoke particles for less processing power. However, the end result is black smoke, which is the same as Kang's rendering result [24]. When real smoke is lit by a light source, it appears as a lighter colour [90]. However, as EXODUS lacks lights, it is not possible to accurately render the lighting of the smoke in colour. A grey colour value of the smoke was picked from the electronic version of the Rubini's paper [90], which should digitally hold the original colour value. However, the proposed algorithm is flexible enough to allow a different colour value to be set if necessary. The greyscale colouring value is defined as $f_s \in c, 0 \leqslant c \leqslant 1$. The colour value from Rubini's paper [90] was about 25% black in greyscale, which makes the colouring value $f_s = 0.25$. The colouring value is added to the final value from the simplex noise algorithm N_{uvt} (see Equation 4.16) [173, 174]. The use of a 3D vector as a 2D uv-texture coordinate with a time value is explained in Section 4.4. The simplex noise algorithm returns values in the range of -1 to 1 [173, 174]. However, the final value of the colour must be clamped to 0 and 1 and therefore it is necessary to add $+1$ and $\frac{1}{2}$ compensations into the Equation 4.15 for step 3 in Figure 4.5. The full range of the noise is not desirable (see Figure 4.10a) and should be around the target grey colour f_s (see Figure 4.10b). Therefore, the total range of the noise is reduced to $f_n = \frac{1}{50}$ of the original value. As described previously, the red c_R colour channel is used for transparency, thus $c_{(R,s)}$ represents the opacity value in step 3 from the source texture in step 2 (Figure 4.5). The above manipulations, with the expensive simplex noise function, are combined with Equation 4.6 to get the final Equation 4.15 for step 3 in Figure 4.5.

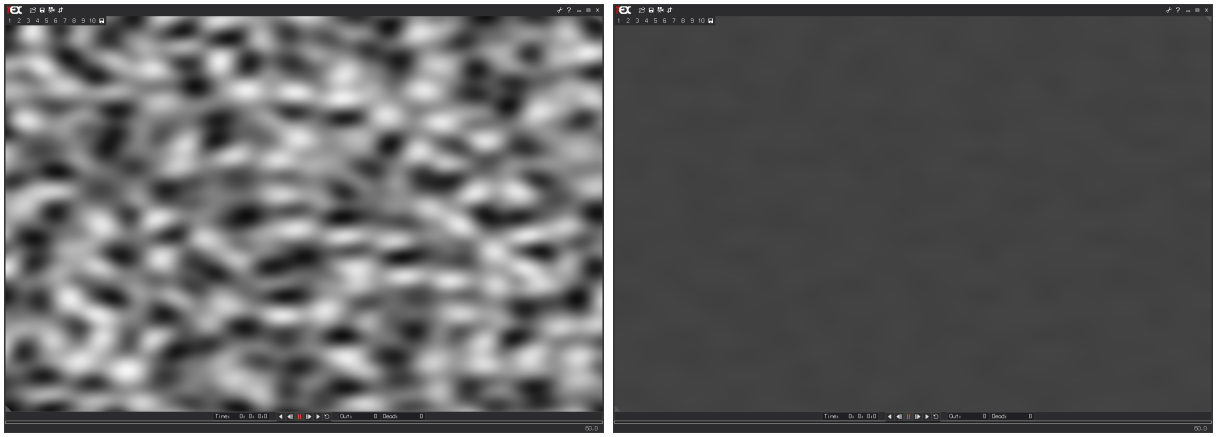
$$\vec{C}_{(d,i+1)} = (1 - c_{(R,s)})(\frac{N(10\vec{V}_{uv}, t) + 1}{2f_n} + f_s) + \vec{C}_{(d,i)}c_{(R,s)} \tag{4.15}$$

Equation 4.15 is the second and final render pass of the smoke. The final render pass is only 2 triangles and most of the computational expense is spent on the fragment shader rendering the pixels.

The 3D simplex noise [173, 174] function with the texture coordinates T_u and T_v, and

time t is defined as:

$$\vec{V}_{uvt} = \begin{bmatrix} T_u \\ T_v \\ t \end{bmatrix} \quad \vec{V}_{uv} = \begin{bmatrix} T_u \\ T_v \end{bmatrix} \quad N(\vec{V}_{uvt}) = N(\vec{V}_{uv}, t) = N_{uvt}, -1 \leqslant N_{uvt} \leqslant 1 \tag{4.16}$$

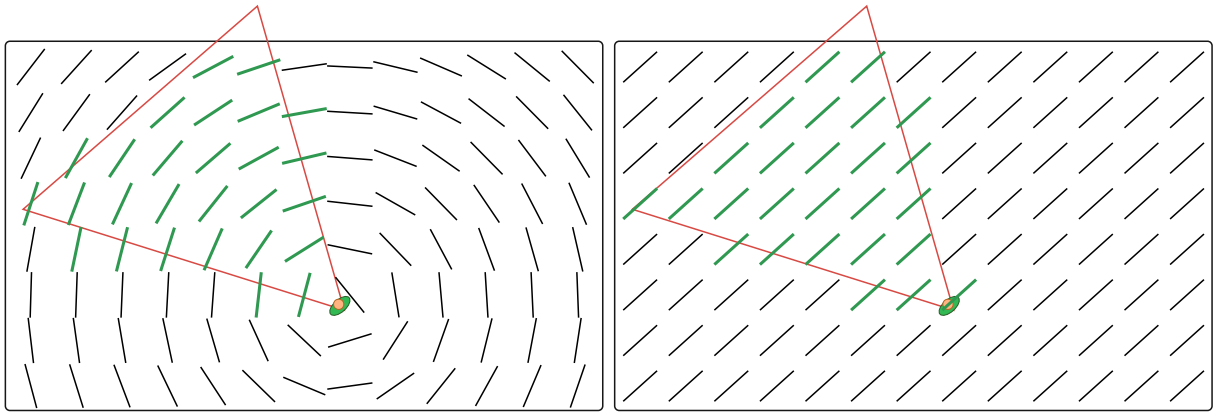


(a) Clamped to 0.0 and 1.0 **(b)** +0.25 grey with $\frac{1}{50}$ the original value

Figure 4.10: Simplex noise colour adjustment

Figure 4.10 shows the final colour adjustment used in Equation 4.15. As can be seen in Equation 4.15, the colour change does not affect the opacity of the smoke.

Even though this thesis demonstrates a simplex noise colouring of the smoke particles, this colouring does not affect the visibility calculations of the smoke and is completely optional. The simplex noise effect is very small but gives the appearance of continuously moving turbulent smoke. The smoke could be coloured to any desired colour in step 3 in Figure 4.5. However, the colour of the individual smoke particles must be monochrome for the smoke algorithm to accurately represent the smoke visibility.



(a) Viewpoint-oriented billboards **(b)** Viewplane aligned billboards

Figure 4.11: Smoke particle billboard system. The green billboards are visible to the user.

Figure 4.11 shows two different ways on how typically quads are aligned as billboards to the screen [169, 175]. The billboard particles in Figure 4.11b are computationally more efficient than the billboard particles in Figure 4.11a. This is due to the sharing of the rotational value between billboard particles in Figure 4.11b, while the billboard particles in Figure 4.11a have individual rotational values. The VR in this thesis as part of the IRIS system uses the method in Figure 4.11b. The user is inside a smoke zone and the smoke zone is divided evenly into a grid of smoke particles. This is similar to the layered billboarding utilised by Smith and Ericson [99]. The size and the quantity of smoke particles depends on the dimensions of the smoke zone. Equation 4.11 allows the definition of the number of smoke particles per metre n_p thus allowing the grid size and density to change dynamically as needed. Due to the billboard system and the grid-based layout, there are some minor variations in the smoke particle density depending on the direction the user looks. If the user's camera points diagonally towards the grid (see Figure 4.11b) the smoke may appear slightly less dense than if looking along the direction of column or row of the grid.

The smoke rendering technique proposed in this section can also be used immediately in game engines such as Godot[3], Unity3D[4] and Unreal Engine[5]. This enables realistic and immersive real-time smoke visibility dependent applications beyond the proposed IRIS system.

[3]Godot - Spatial Material `https://docs.godotengine.org/en/3.1/tutorials/3d/spatial_material.html#blend-mode`

[4]Unity3D - ShaderLab: Blending `https://docs.unity3d.com/Manual/SL-Blend.html`

[5]Unreal Engine - Material Blend Modes `https://docs.unrealengine.com/en-US/Engine/Rendering/Materials/MaterialProperties/BlendModes`

4.7 Smoke validation

This section aims to validate the smoke visibility algorithm described throughout this chapter based on the validation definition by the ISO (International Organization for Standardization). The ISO definition of validation is *"process of determining the degree to which a calculation method is an accurate representation of the real world from the perspective of the intended uses of the calculation method"* [176, 177].

This validation case aims to reproduce Jin's experiment from 1970 (see Section 2.2.1 in Chapter 2). The smoke implementation details are covered earlier in this chapter (see the four previous sections, Sections 4.3, 4.4, 4.5 and 4.6). In this section the smoke rendering calibration is being validated. The smoke visibility is based on Kang's work [24], which is ultimately based on Jin's work in 1970 [18]. The industry standard smoke visibility calculations in SFPE Handbook of Fire Protection Engineering [20] are also based on Jin's work in 1970.

To correctly validate the visibility of the smoke, it is necessary to look back into the original experiment almost 50 years ago and to mimic the results in the virtual environment. Jin's paper [18] describes an experiment where a container is filled with smoke. The container is of a size 1.2 metres wide, 1.2 metres high and 5.5 metres long. An electric furnace burns 10g of filter paper (translated type: *"Toyo furnace Paper No. 2"*) at $400°C$. The smoke is allowed to gather for about 2-4 minutes, until it is observed. The container houses a small fan to make the smoke uniform.

The paper [18] described that the observer looks through a glass into the container, trying to spot and interpret a placard. The placard has a *"pattern of sign"*. The sign is interpreted by the translate program to be e.g. a horse, a duck or a cauldron. (For this experiment the actual sign is unimportant, as long as the sign is clearly recognisable.) The sign is black in colour on a placard (stamp paper) against white painted walls. When looked at from 5.5m away the placard is 5cm in diameter, from 10.5m away the placard is 10cm in diameter and from 15.5m away the placard is 15cm in diameter. Jin explained that the distance was varied by using 2 mirrors in the 5.5 metres long smoke chamber and the size of the marker was

changed for each distance to keep the size of the observed placard constant to the observer over multiple distances. [18] The container is illuminated with multiple lights.

The paper proposes the following visibility distance calculation (derived from Equation 5 on page 5 in [18]):

$$V = \frac{B}{a} \tag{4.17}$$

where a is the extinction coefficient, V is the visibility distance in metres, and B is the brightness of the sign. The unlit sign has a B value approximately between 2 and 4, and a lit sign approximately between 5 and 10.

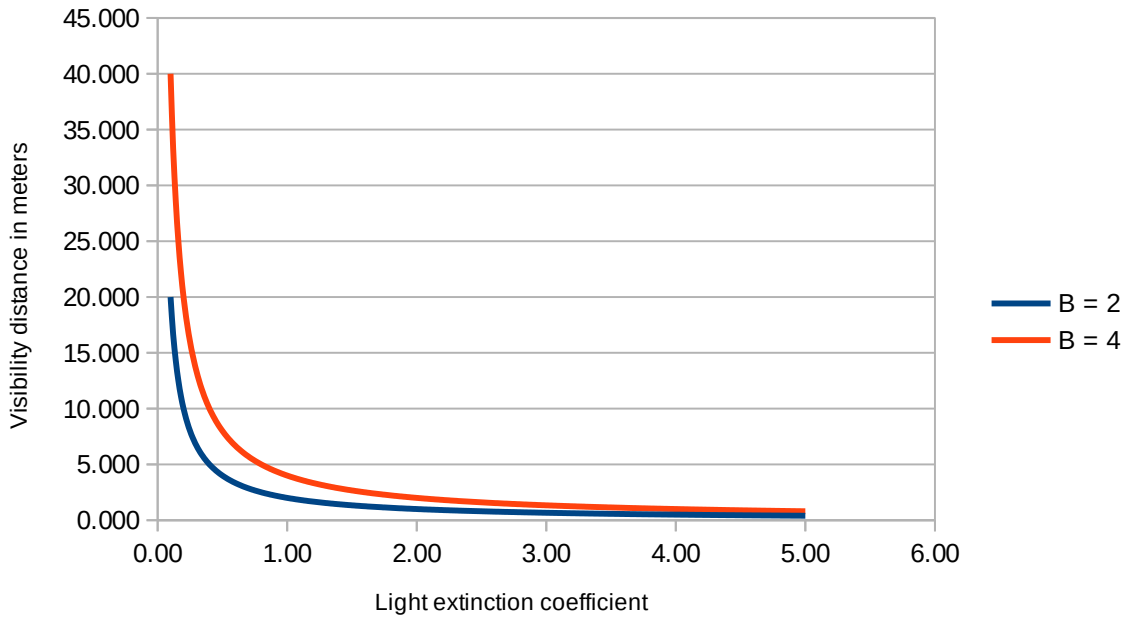


Figure 4.12: Visibility of unlit sign to light extinction coefficient.

Figure 4.12 shows what Equation 4.17 looks like for an unlit sign. As can be seen from the graph: as the light extinction coefficient grows, the visibility of the unlit sign drops significantly and fast. The range between the two curves is the grey area where some observers are able to distinguish the sign, and some are not. Any distance beneath the bottom curve means everyone with normal vision should be able to see the sign.

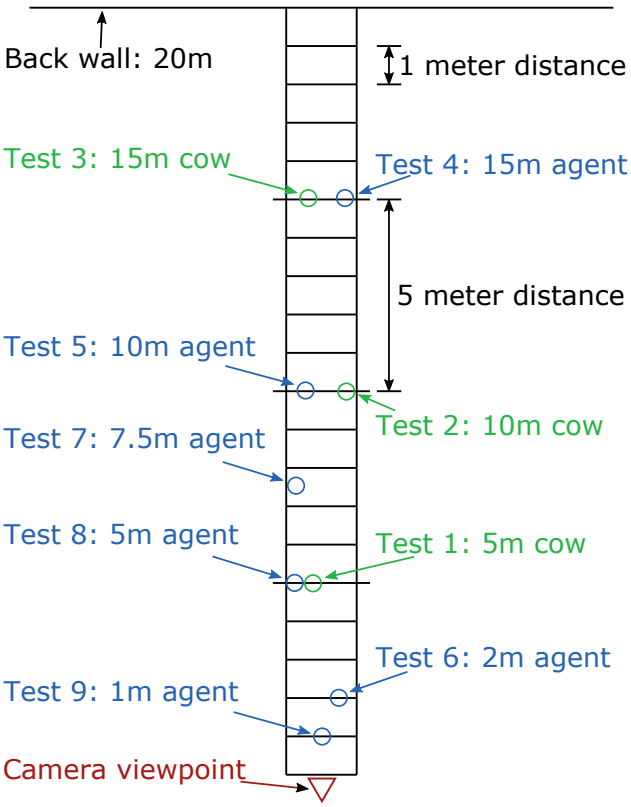


Figure 4.13: Smoke test geometry in EXODUS

The test case used to validate the smoke visibility is shown in Figure 4.13. It is a 20 metres long geometry, where the user is looking across the geometry. Jin had to use mirrors to achieve 10 metre and 15 metre distances within a 5.5 metre chamber, however this is not necessary in VR, and the geometry has been made sufficiently long to avoid having to use and simulate mirrors. To have a better understanding of the gradual gradient effect of the rendered smoke, short walls have been placed at 1-metre intervals throughout the geometry. It is important to note that the distances as stated in Figure 4.13 are from the location of the camera and the total obscuration distance within smoke. The smoke obscuration starts at the location of the camera which is the location of the computer monitor in VR and the distances in Figure 4.13 do not include the additional space between the user and the monitor in the physical world where there is no smoke.

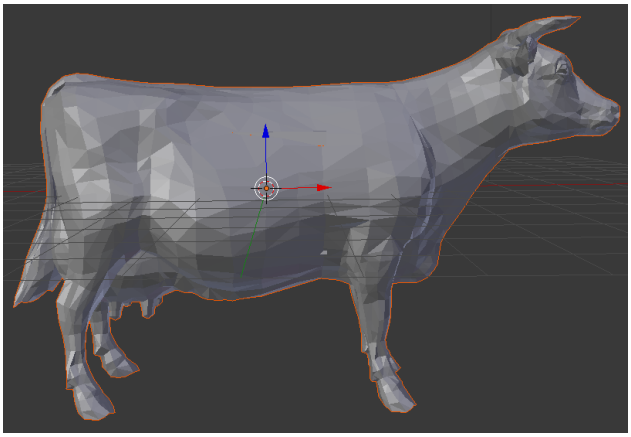


Figure 4.14: 5cm long test image

The 3D test model (cow) to be seen from 5 metres away at the total width of 5cm (test 1) is shown in Figure 4.14 to replicate the test in the original experiment by Jin [18]. The model is scaled to 10cm and shown at 10 metres away for test 2 and scaled to 15cm and shown at 15 metres away for test 3. The cow model in the three tests is the same size and at a similar distance to what Jin used in his tests. The model is drawn in white colour against black smoke. This is reversed from the black sign against white smoke that Jin used [18]. Nevertheless, the total contrast between the sign and the smoke remains (i.e. comparing black against white or white against black). Both white and black smoke do produce very similar results in real-life experiments [21].

The second model used in the smoke visibility tests is a full-size 3D model of an agent. Jin did not use a full-size person in his tests and therefore it cannot be used to validate the smoke visibility algorithm. Nevertheless, it is included in the user tests as the user exercises in Chapter 6 use the 3D agent model to represent both user and computer-controlled agents in a smoke environment. Comparing its performance in the same smoke tests as the 3D cow model helps to understand its visibility in smoke in comparison to the validation tests done against Jin's results. As this section aims to validate the smoke algorithm utilising the cow model, the full-size agent results are not presented here and can instead be seen in Appendix E along with further screenshots of the smoke tests.

The computer monitor that was used to display the VR image was 32" 16:9 aspect ratio BDM3270 from Philips and the resolution was 1440p (QHD)[6]. This was the most colour accurate monitor available, provided by FSEG. There was no further colour calibration performed in addition to the factory calibration by the manufacturer. The colour palette used was sRGB with 50% brightness, 50% contrast and 50% sharpness. The gamma was 2.2 and colour temperature 6500K. Brightness was set to 50% as the experiment was done in an old dimly lit room in the Old Royal Naval College and at 100% brightness white colour is uncomfortable to the eyes of the user. Smart contrast was off and over scan was off. Another 32" monitor[7] with the same aspect ratio and pixel density was available and was placed next to the test monitor for image comparison. Multiple users compared the smoke visibility on both monitors and found no significant difference between the resulting images.

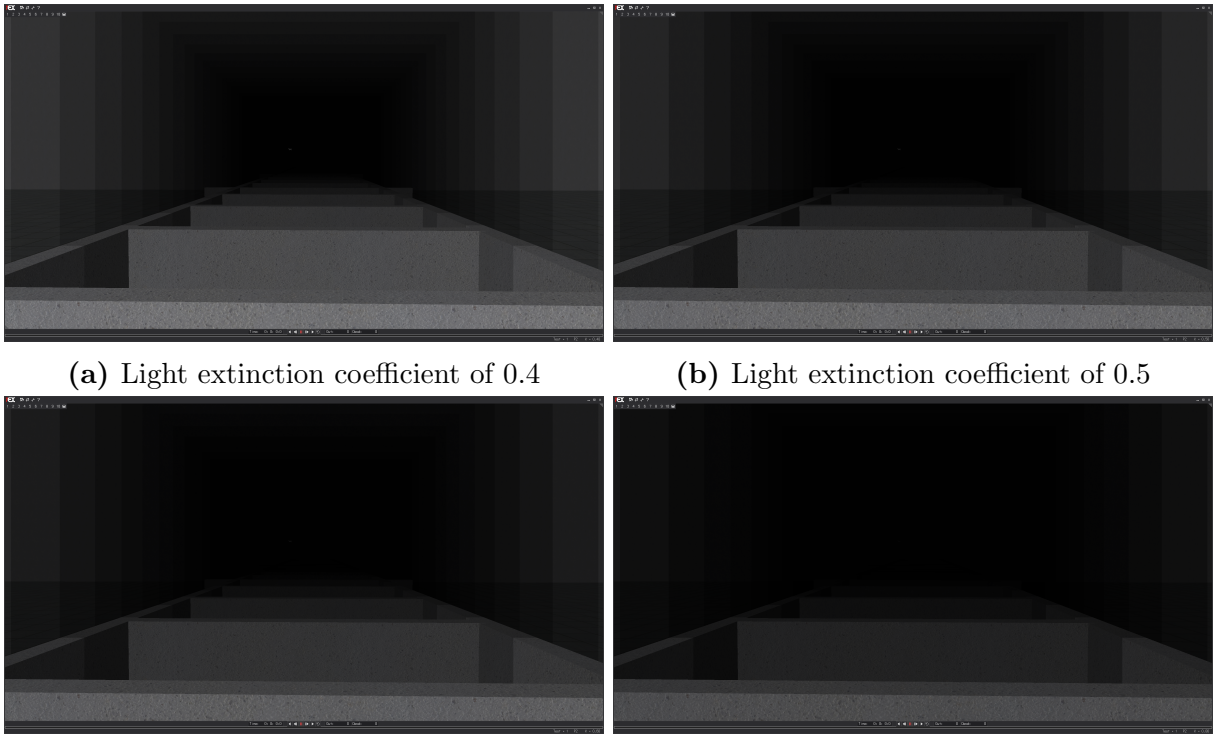
The field of view used for the view frustum in the perspective projection matrix of the virtual environment (camera) was calculated as such:

$$d = \frac{w}{2 \times tan(\frac{\alpha}{2})} \tag{4.18}$$

45° degree angle (α) was used as the field of view. The user was sitting at d distance away from the monitor. The d is also the near view plane used in the virtual environment. w is the width of the displayed image on the monitor, which in the case of 16:9 aspect ratio 32" monitor is about 70.84cm. The end result of Equation 4.18 is the correct viewing distance d for the field of view, which in this case is the user's eyes must be about 85.5cm away from the monitor.

[6]Philips BDM3270QP/00 LED-backlit LCD Display with MultiView specifications: `https://www.philips.co.uk/c-p/BDM3270QP_00/brilliance-led-backlit-lcd-display-with-multiview/overview`

[7]Samsung LC32JG50QQUXEN: `https://www.samsung.com/uk/monitors/c27jg50/LC32JG50QQUXEN/`



(a) Light extinction coefficient of 0.4

(b) Light extinction coefficient of 0.5

(c) Light extinction coefficient of 0.6

(d) Light extinction coefficient of 0.8

Figure 4.15: Smoke validation: 2 particles per metre with varying light extinction coefficients

The validation was done using uniform smoke, due to Jin's experiment also using uniform smoke [18]. Test 1 from Figure 4.13 can be seen in Figure 4.15 demonstrating the effectiveness of the visibility in smoke with different light extinction coefficients. The figure uses variables from Equation 4.11, where light extinction coefficient is denoted as a and number of particles is denoted as n_p. The σ is the optical density of smoke and light extinction coefficient a is calculated as $a = \sigma * 2.303$ [178], which is derived from [24] (i.e. *optical density per unit distance* (m^{-1}) = *mass optical density* (m^2/kg) * *mass concentration of fuel burned* (kg/m^3)) [179].

According to Jin's test [18], all observers should be able to see the sign on the unlit placard in Figure 4.15a, with visibility of the sign being between 5 metres and 10 metres (see Equation 4.17). Figure 4.15a can be seen scaled up in Figure 4.16 with the sign highlighted with a red circle. Note: the user is looking at the original image in full screen on 32" monitor 85.5cm away. As can be seen in Figure 4.16: there are not many pixels left on a QHD monitor for the 5cm sign at 5m away leaving the sign indistinguishable from other signs. However,

the presence of the sign is clearly visible against the contrasted background. Therefore, the actual sign used in the experiment was not important. A higher resolution monitor would help in identifying the 5cm sign at 5m away as there would be more pixels available to the raster of the sign to be rendered in a higher quality.

The sign in Figure 4.15b demonstrates the first cut off point where not all observers can see the sign, with the visibility of the sign being between 4 metres and 8 metres. The visibility reduces further in Figure 4.15c with the visibility of the sign being between 3.33 metres and 6.67 metres. Finally, the Figure 4.15d shows the sign visibility between 2.5 metres and 5.0 metres, where most observers would be able to see the sign.

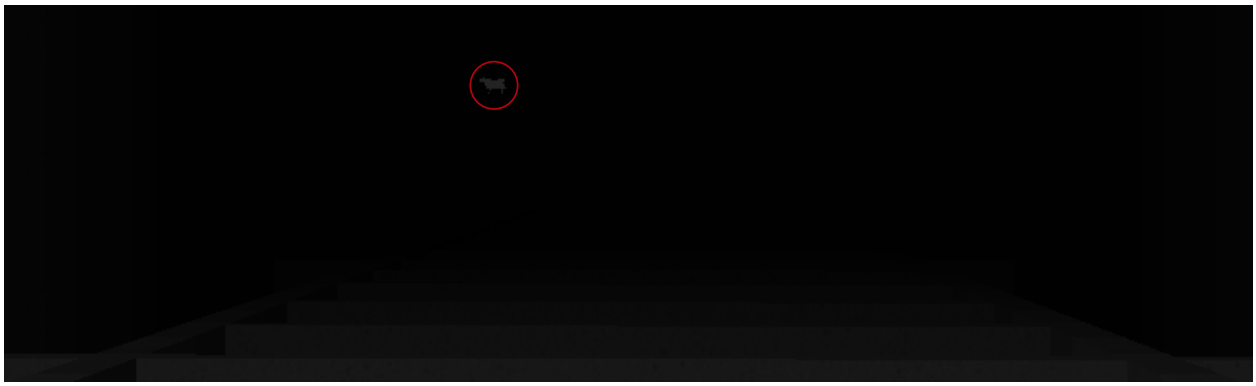
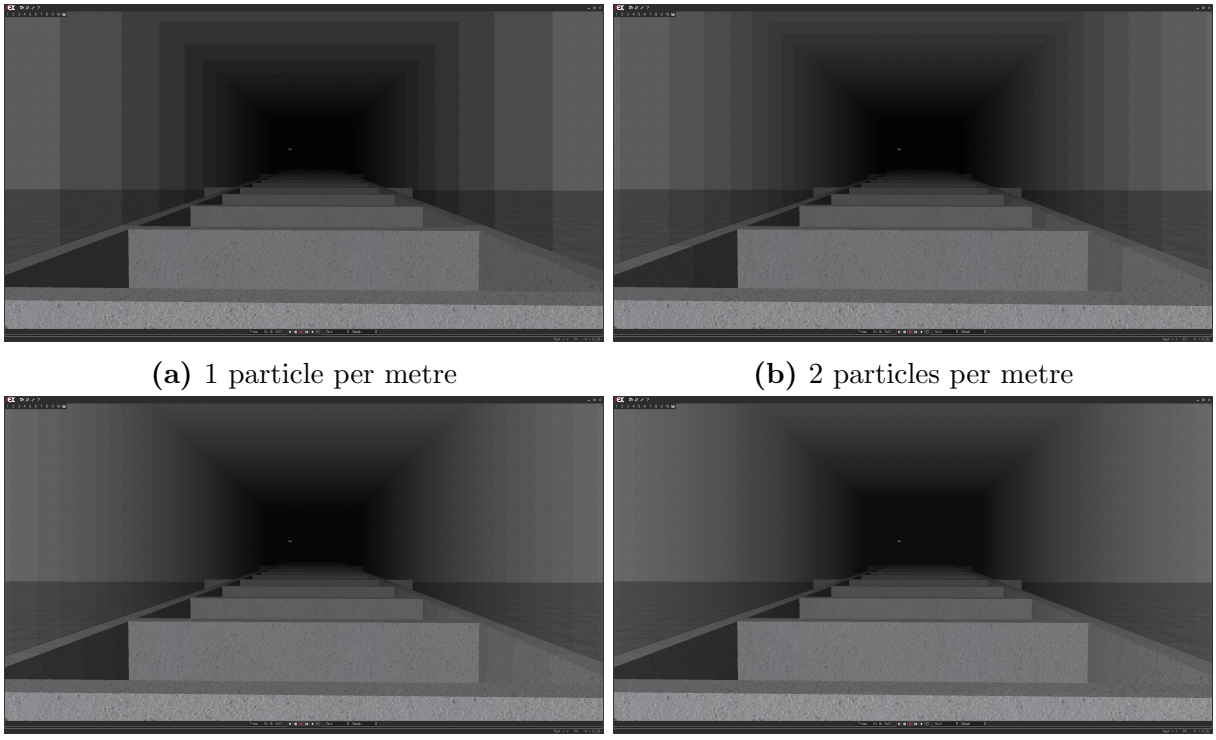


Figure 4.16: Figure 4.15a zoomed in.



(a) 1 particle per metre

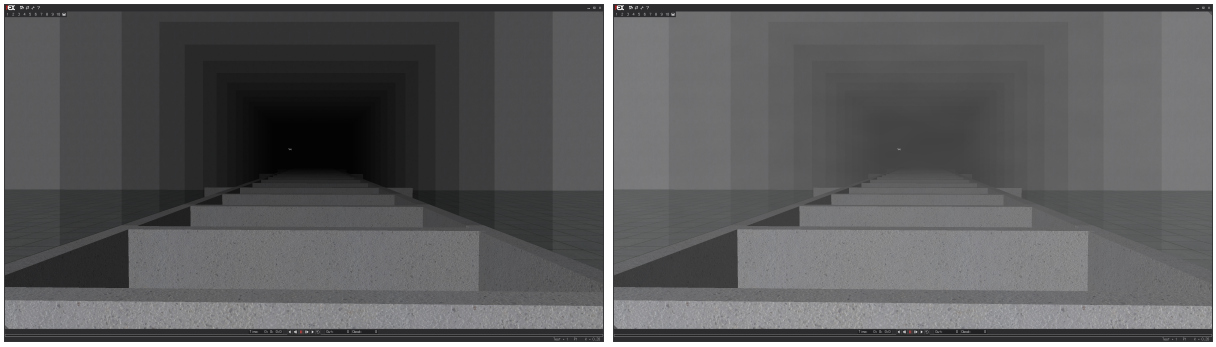
(b) 2 particles per metre

(c) 4 particles per metre

(d) 8 particles per metre

Figure 4.17: Smoke validation: 0.2 light extinction coefficient with varying number of particles per metre

Figure 4.17 demonstrates how the visibility in smoke does not change when the number of smoke particles are increased, using the calibrated algorithm detailed in Section 4.6. However, to be exhaustive: due to the discrete nature of computer processors and memory, the smoke does change in a minuscule level when the number of smoke particles per metre are changed. This change is dependent on the hardware that was used and is not visible to the user as long as a sufficient accuracy is used in the smoke rendering calculations. In normal 8 bit colour channel space, the difference is not noticeable, when the number of particles are increased up to 8. When more particles are introduced per metre, the colour accuracy of the user colour channel should also be increased.



(a) Black smoke **(b)** Coloured smoke

Figure 4.18: Smoke validation: Black smoke vs. coloured smoke. 1 particle per metre with 0.2 light extinction coefficient.

Figure 4.18 demonstrates how the smoke colour is changed from black to gray. This change in colour does not change the overall obscuration of smoke as seen in Figures 4.18a and 4.18b. The chosen colour of the smoke may help or hinder the visibility of some objects, such as when the hue of the smoke is similar to the hue of the obscured object, the object may therefore blend into the smoke. However, this is a different test from the light extinction coefficient test performed in this section and is not performed in this thesis.

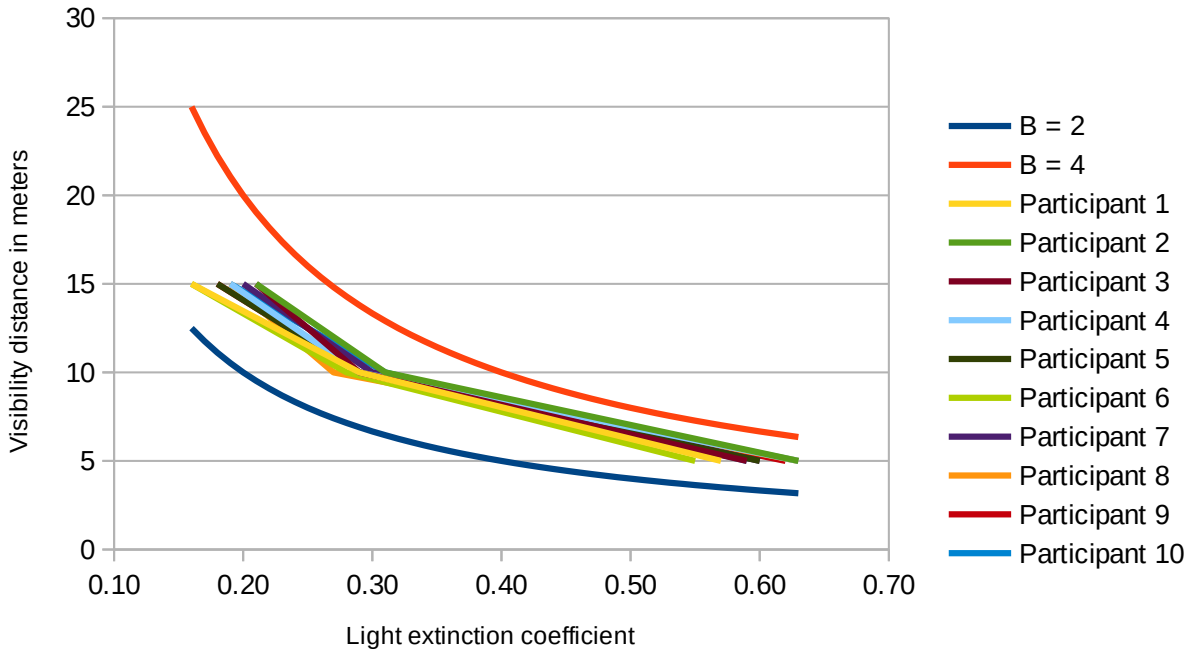


Figure 4.19: Smoke validation: Visibility of 3D cow model by participants in comparison to Jin's experiment.

Figure 4.19 shows the observed visibility of 10 participants who took part in the visibility experiment as described earlier in this section. The experiment has been approved by University of Greenwich Research Ethics Committee (see Appendix B for the approval letter). It is clear from the figure that the observed results are well within the two B values for non-lit sign as stated by Jin (see Equation 4.17).

Each participant was seated in front of a computer monitor at roughly 80cm distance as described earlier in this section and took the test independently from each other without discussing the results or seeing another participant taking the test. Each participant had full control over the smoke visibility. The program showed each participant a K value, which indicates the *light extinction coefficient*. The participants were asked to record the shown K value at the point when the object was barely visible. Smoke visibility was controlled with the mouse-wheel at increments of 0.01 with a minimum value of 0.00 and maximum value of 10.00 allowing for a total of 1000 levels of granularity. The participants also compared visibility differences between different numbers of particles per metre, which ranged from 1 particle per metre to 10 particles per metre. Table 4.1 below shows that there is very little deviation in the observed visibility if the smoke is shown at different number of particles per metre.

Table 4.1: Smoke validation: Observed visibility variations in the number of particles per metre

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9
Mean K-value	0.60	0.29	0.19	0.23	0.31	1.56	0.37	0.62	3.02
Standard Deviation	0.005	0.0044	0.0199	0.0374	0.0480	0.0400	0.0093	0.0051	0.1155

Full screenshots of each test at key visibility values (i.e. the B values) can be seen in Appendix E. The questionnaire the participants answered can be found in Appendix C.

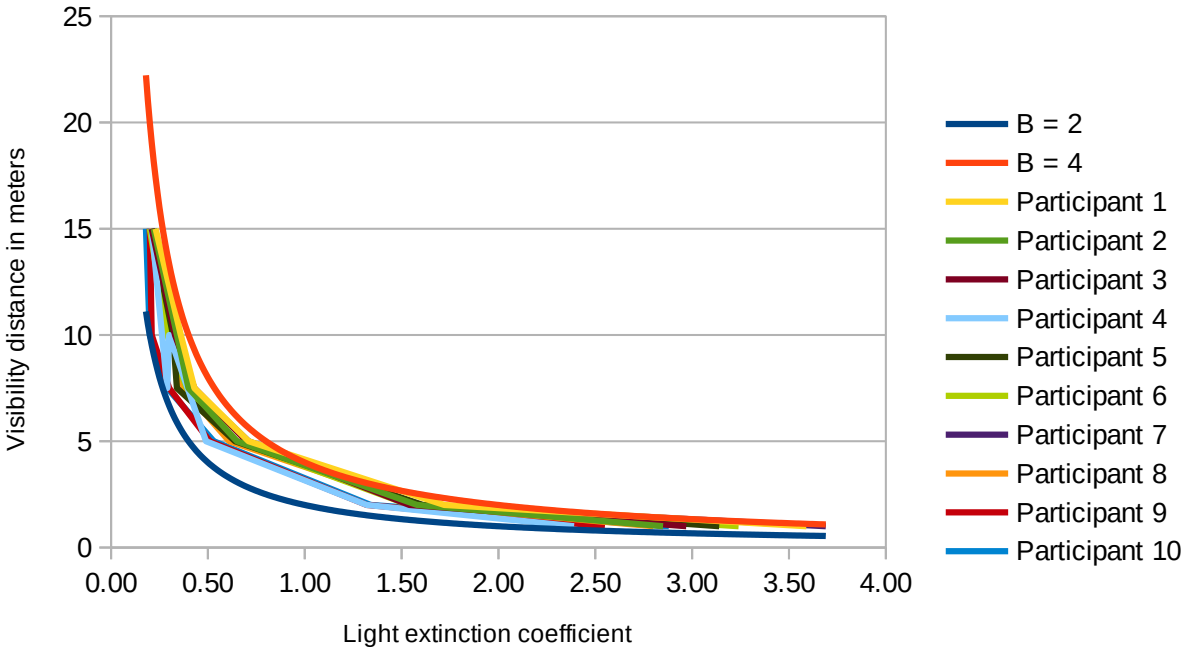


Figure 4.20: Observed visibility of 3D agent model.

Figure 4.20 represents the observed visibility of the 3D agent model by the participants. Jin did not include full-size humans as observed targets within smoke and thus the Figure 4.20 cannot be used as part of the validation results. However, when comparing the visibility of the agents to the expected visibility range of non-lit signs, the visibility of the agents falls clearly within the expected range as seen in Figure 4.20.

It is clear from the validation results that the proposed smoke algorithm for accurate real-time smoke visibility representation is indeed accurate to the current understanding of smoke visibility. Furthermore, the real-time algorithm allows granular scaling of the render quality to reach desired performance levels without affecting the accuracy of the smoke visibility.

4.8 Aircraft fire

This demonstration case aims to show the effectiveness of the new calibrated real-time smoke rendering technique. The airport geometry was used initially, but due to the large space and the lack of narrow corridors, even as large as 8MW fire can take 5-10 minutes to fill a single

room, and as long as 20 minutes or longer to fill both air side and land side. Appendix A shows what the smoke looks like in the airport geometry.

However, a more interesting demonstration includes a maze-like structure with narrow spaces with multiple corridors that will prove challenging to navigate with smoke obscured vision in comparison to an open space. Fortunately FSEG had successfully completed NACRE EU-Project (Project ID: 516068) in 2010 as one of the project partners [180], which included a large Blended Wing Body aircraft (see Figure 4.21) [181].

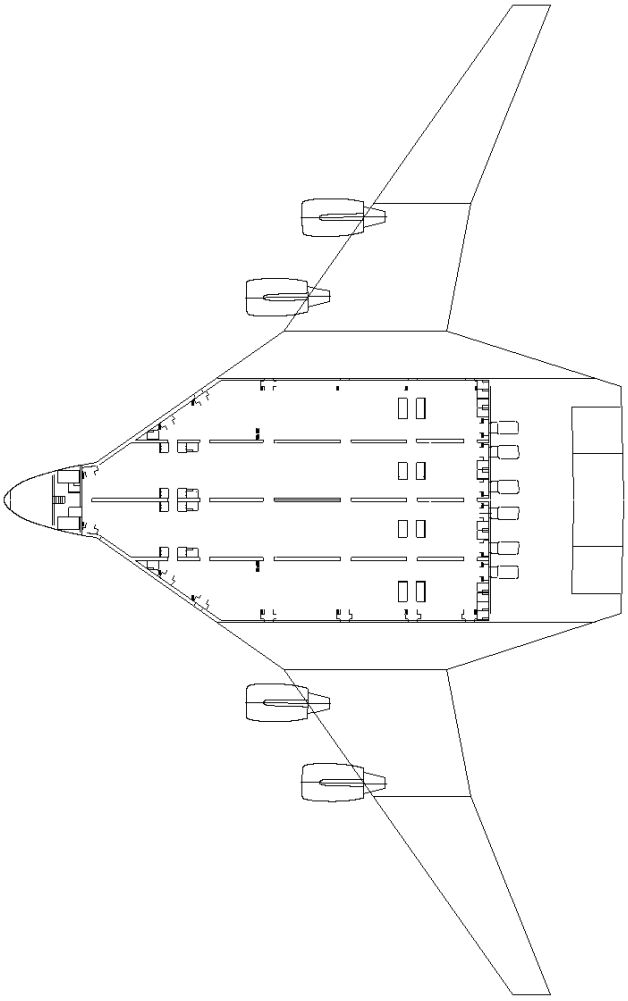


Figure 4.21: NACRE Blended Wing Body aircraft CAD drawing

As a result of the NACRE project this aircraft was proposed as a plausible future passenger aircraft capable of carrying more passengers than current airplanes, as larger aircraft may be necessary as the world's population increases. This is an interesting case to demonstrate the effectiveness of the smoke due to a wide space with low ceiling as smoke will quickly fill a wide area. The internal layout (see Figures 4.22 and 4.23) is more complex than a typical aircraft [181–183].

The research conducted as part of the NACRE project demonstrated the effects of fire and its products (i.e. smoke, heat and toxic substances) on simulated computer-controlled agents in a virtual reality (VR) environment during in an evacuation [182–184]. The fire data in the published research was produced by the SMARTFIRE CFD fire model [182, 183]. It is important to note that equivalent fire data from a different CFD model such as FDS[8] could be used instead. However, the fire data from SMARTFIRE is already available and is thus used in here. This demonstration case will therefore focus on what was not

[8]FDS is freely available from `https://pages.nist.gov/fds-smv/`

previously demonstrated: realistic smoke visibility visualisation in a VR environment. This demonstration case will show the smoke visualisation from both the coordinator's point of view (i.e. bird's-eye view) as well as from the participant's user-controlled agent's point of view (i.e. a first person view).

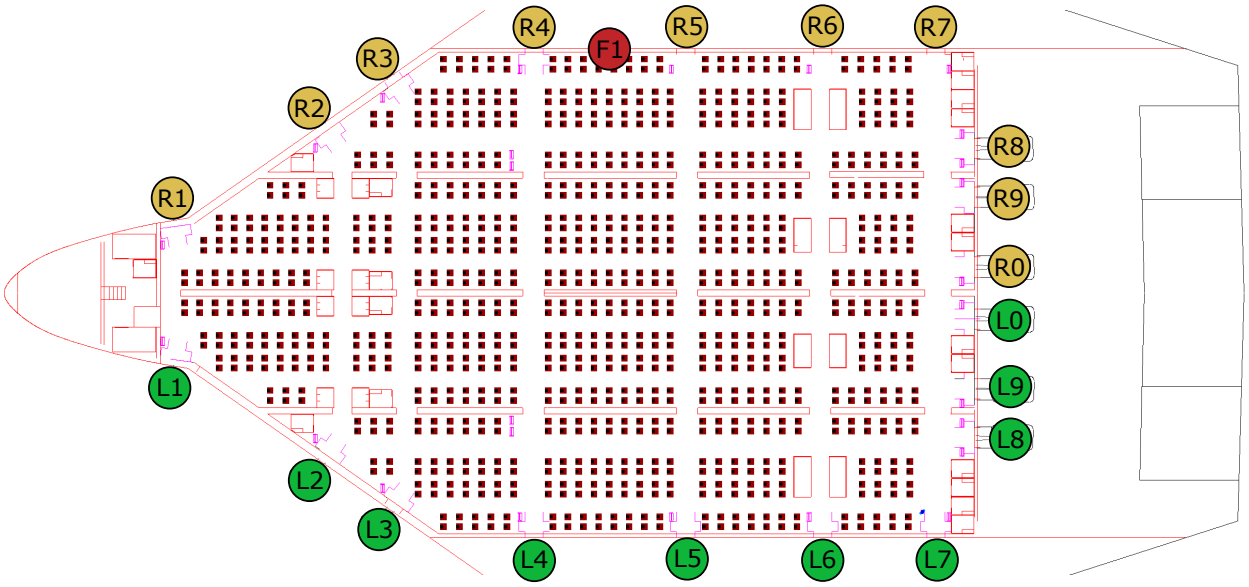


Figure 4.22: CAD drawing in EXODUS, including seats

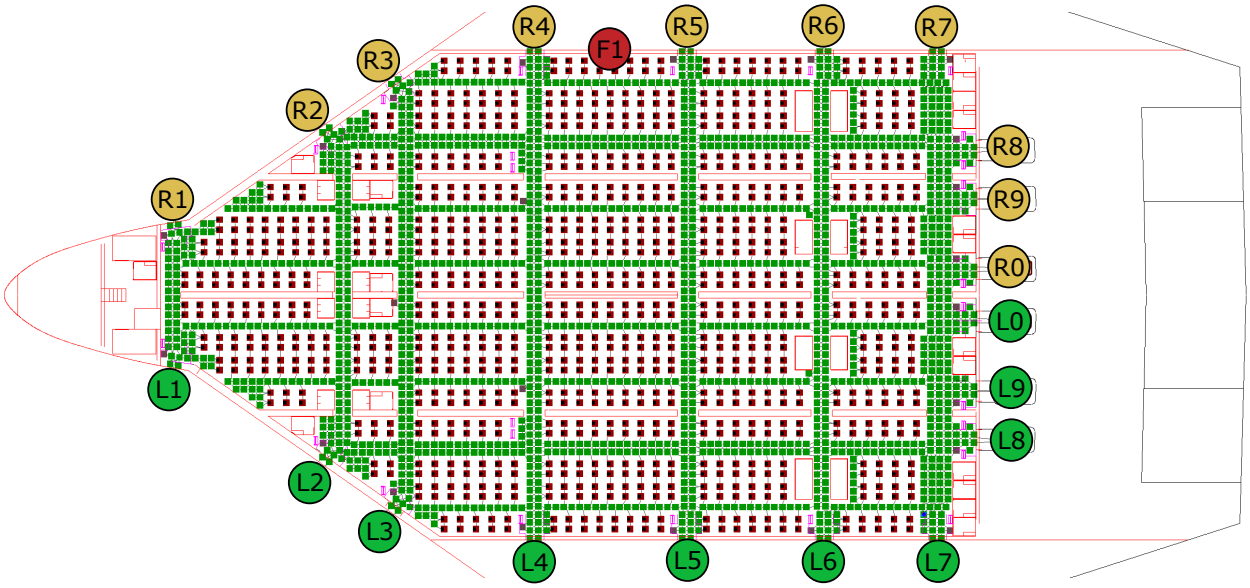


Figure 4.23: CAD drawing in EXODUS with movable space in green

The internal movable space within the aircraft is 24 metres wide (from wing to wing) and 40 metres long (along the middle isle from the back exit to the captain's cabin). The aircraft is compartmentalised into 4 distinctive zones, two inner zones which are 5.5 metres

wide, and two outer zones which are 6 metres wide. These zones are separated by walls and doorways (see Figure 4.22). The outer zones feature a wider walk way than the inner zones, which add half a metre to the overall width (see Figure 4.23). The internal height of the aircraft is uniform at 2.4 metres from the floor to the ceiling.

The aircraft also has 20 exits, named R0-R9 (in yellow) and L0-R9 (in green) for right and left side of the aircraft respectively (see Figures 4.22 and 4.23). This demonstration case scenario features a fire (the same fire as in the EU NACRE project) with the starting location marked as F1 (see Figures 4.22 and 4.23).

Togher wrote in her Phd thesis in 2014 [48]:

"The evacuation certification trial requires that 50% of exits be made available (see FAR 25.803). In contrast to these requirements, the AASK V4.0 study suggests that a third (33%) of the emergency evacuations examined involve aircraft in which less than 50% of the exits are available. The data also suggests that the available exit distribution for small (i.e. aircraft with three exit zones) and large aircraft (i.e. aircraft with four exit zones) is different with smaller aircraft having a greater tendency than larger aircraft to have less than 50% of their exits available during an emergency evacuation. In addition, the accident analysis suggests that over half (55%) of the accidents investigated involve a cabin section in which no exits were available. It is therefore suggested that further simulations be carried out and outputs analysed for both narrow bodied and wide bodied aircraft with less than 50% of the exits being available."

This demonstration case proposes how immersive and validated visibility in smoke can be used to enhance the graphical fidelity and realisms that a human participant experiences in a computer simulation. As was discovered in the literature review: for the user to be able to behave naturally in a simulated environment, naturally available information has to be available also in the virtual world in a natural way. This is why hindering user's visibility in smoke by a natural amount (as validated against previous data and experiments from literature) is crucial for the user to behave in the virtual world as he would in the real world.

For this demonstration case at least 50% of the exits are unusable, which includes the right-hand side named R0-R9(in yellow) as seen in Figures 4.22 and 4.23. The right hand

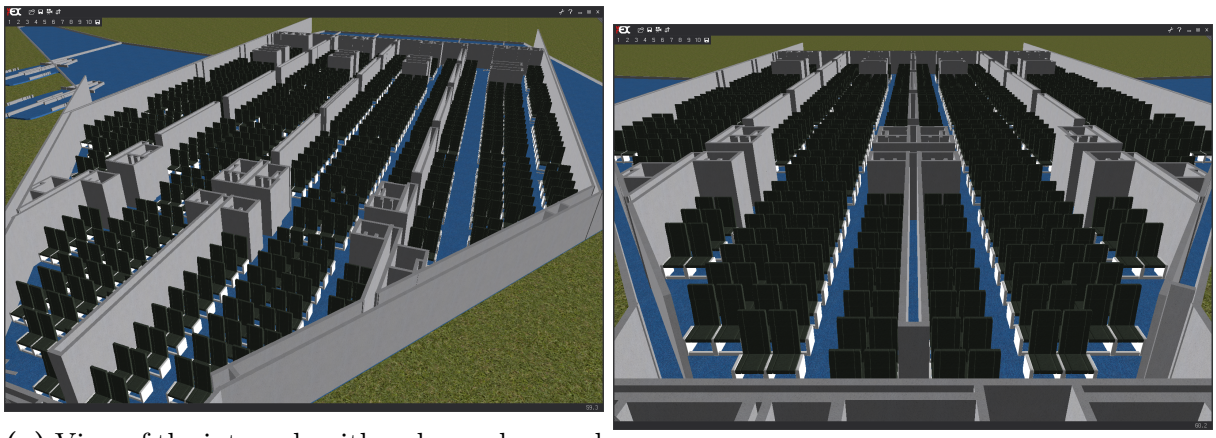
side is also the side where the fire initiates and the smoke start to spread.



(a) The whole aircraft in vrEXODUS **(b)** Closer view of the internals in vrEXODUS

Figure 4.24: 3D view: Outside view of the NACRE aircraft in vrEXODUS

Figures 4.24a and 4.24b show how the aircraft CAD drawing is interpreted in the vrEXODUS program using the same textures as in the airport geometry but includes 3D seat models provided by FSEG.



(a) View of the internals with a changed ground texture

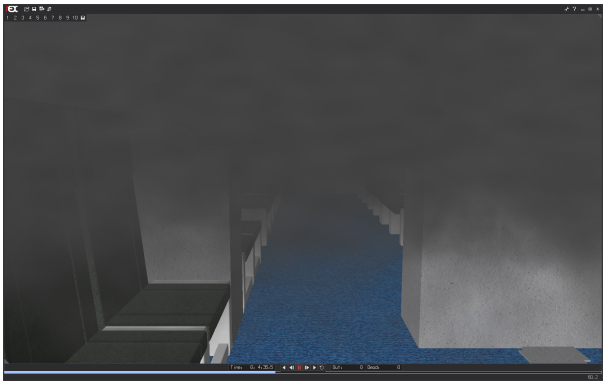
(b) View from the front

Figure 4.25: Close up outside view of the NACRE aircraft in vrEXODUS

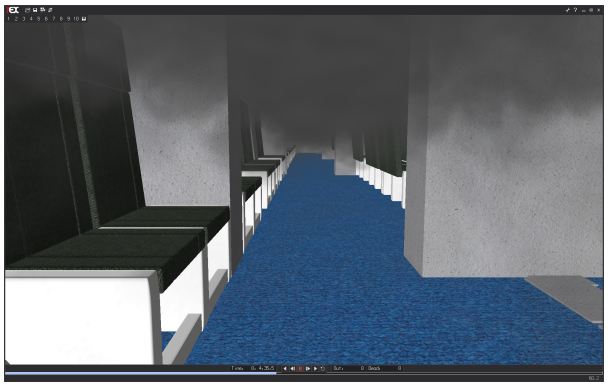
Figures 4.25a and 4.25b show the aircraft closer, but with a blue carpet texture on the ground instead of the tiled concrete slabs. The blue carpet adds contrast to the floor and is a better fit for an internal aircraft environment.



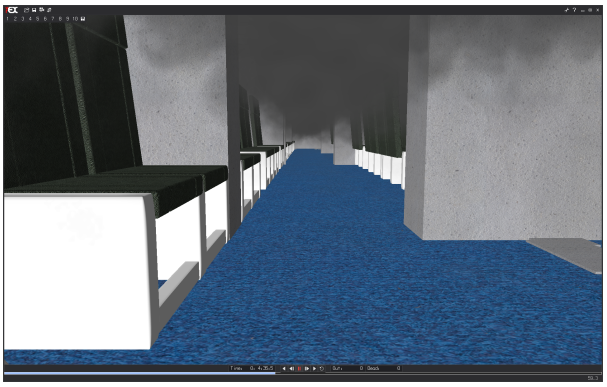
(a) Smoke obscuring vision at eye level when standing



(b) Smoke clearing when moving to crouching position



(c) Smoke obscuration level at crouching position



(d) Smoke obscuration at prone position (below the upper smoke layer of 1.4 metres)

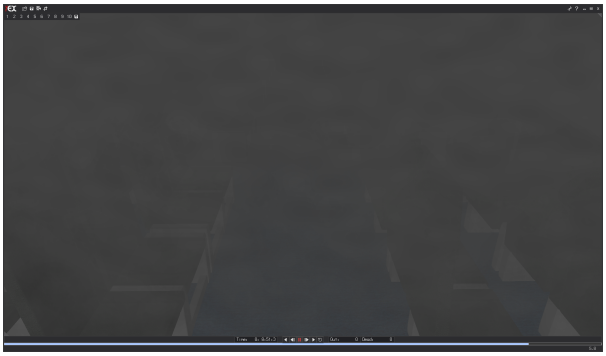
Figure 4.26: Internal NACRE aircraft smoke visualisation in vrEXODUS with 2 layer smoke representation

Figure 4.26 demonstrates how moving below the upper smoke layer effectively gives a much clearer view of the surroundings. Moving in crouching or in prone position is more difficult and takes longer than when standing upright, but at the same time allows easier navigation.

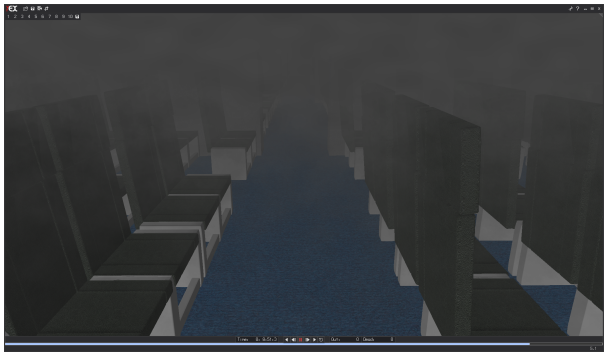
The smoke shown in Figure 4.26 is using 2 smoke layers from EXODUS simulation called upper smoke layer and lower smoke layer. EXODUS uses 2 layers as an agent can be either in standing position or in a crouching position, and depending on which position the agent is at, the relevant smoke layer will affect the agent. This is sufficient for a computer-controlled agent as the simulation only has to store and process relevant information for a simplified computer simulated vision in two different 2D planes. However, this introduces problems in a virtual world for the human user as the user can see a larger vertical space (compare Figures 4.26 and 4.27).



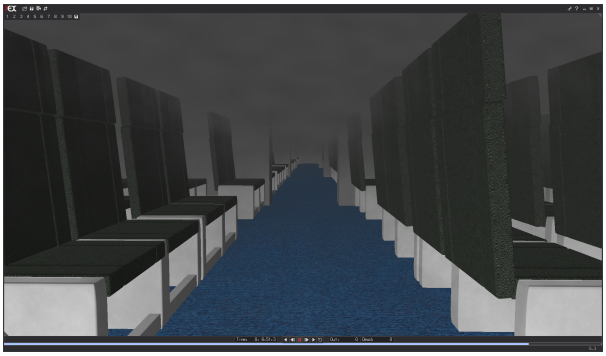
(a) Smoke obscuring vision at eye level when looking forward while standing



(b) Some geometry can be seen when looking downwards



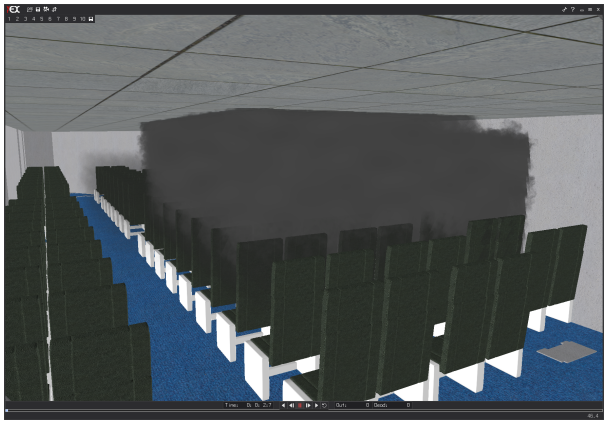
(c) Visibility in smoke becomes clearer when leaning down



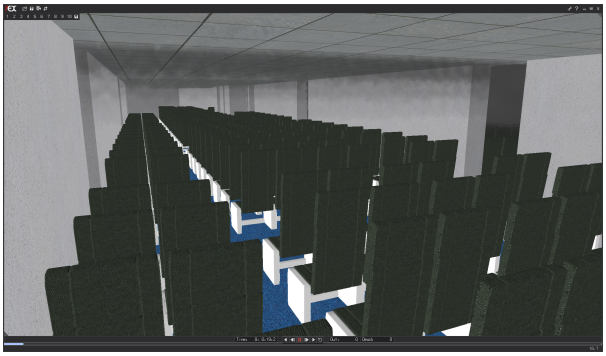
(d) Navigation becomes possible in crouching position

Figure 4.27: NACRE aircraft: Internal smoke visualisation in vrEXODUS with 10-layer smoke representation

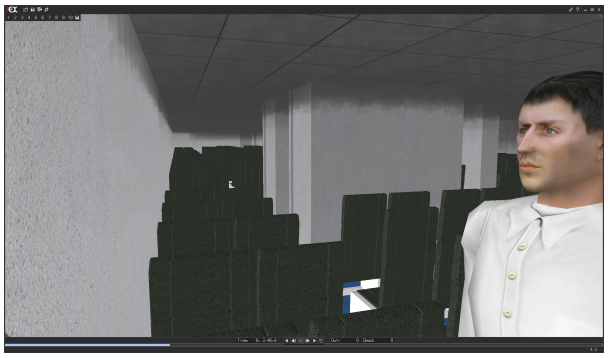
Figure 4.27 demonstrates visibility in a 10 vertical layer smoke zone model and how gradual the actual CFD results are in comparison to the 2 vertical layer smoke zone model in Figure 4.26. Figure 4.28 shows the difference between 2 and 10 vertically divided smoke layers. The user can still crouch and gain better visibility closer to the floor, but the smoke does not suddenly clear away (see Figure 4.27d).



(a) Smoke progression on the first 2 smoke zones at 2.7 seconds into the simulation when using the original smoke zones from the EU NACRE project



(b) Smoke enters the second aisle of seats through the door way as seen at 19.2 seconds into the simulation



(c) The furthest exit has a thick smoke layer in the ceiling at 2 minutes 46.8 seconds into the simulation



(d) Once the smoke has filled majority of the aircraft at 4 minutes 42.7 seconds into the simulation, the aircraft becomes difficult to navigate

Figure 4.29: NACRE aircraft: Internal smoke visualisation in vrEXODUS in key locations

The smoke rendering significantly improved vertically, but the horizontal smoke zone model still provides some artificial artefacts, such as the first smoke zone appears to have an invisible force field around it holding the smoke inside (see Figure 4.29a), but this becomes a lot less apparent within seconds beyond the initial time step.

Ideally the smoke would be rendered exactly as is simulated in a CFD engine, however, the total number of particles (zones/voxels/hexahedron subdomains - depending on the CFD model) can easily exceed hundreds of thousands or even millions of individual particles. This NACRE aircraft is a much smaller geometry than the airport featured earlier in this chapter, and the aircraft alone was divided into over 300,000 individual hexahedron cells in the SMARTFIRE CFD simulation program. Implementing this adds optimisation challenges

in terms of performance and memory, as well as rendering challenges as the subdomains may be any arbitrary shape. This was left as a further work and is discussed in Chapter 7.

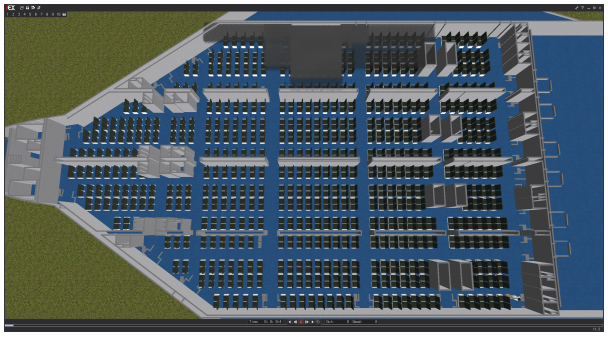
Due to the 10 layered approach, the user can get a visible cue sooner than he would have on a 2 layered approach. This is apparent on Figure 4.29b as the smoke travels through the doorway and appears near the ceiling. The smoke is still above the user and does not hinder his visibility and therefore the user has more time to react. Similar smoke behaviour can be seen in Figure 4.29c where the ceiling is filled with smoke, but the smoke layer has not yet lowered to the user's level.



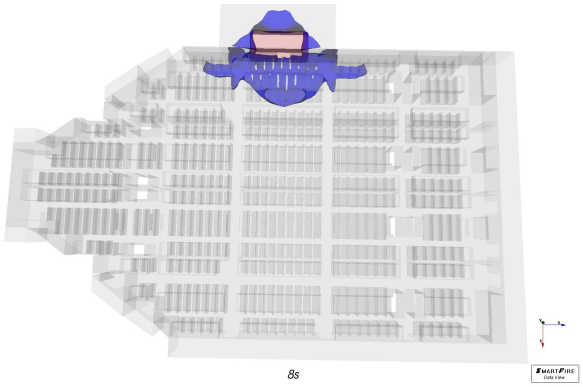
Figure 4.28: Difference between 2 (on the left) and 10 (on the right) vertical layers of smoke.

Figure 4.29d shows the user moving from R6 exit towards L6 exit (see Figure 4.23) and the smoke has covered most of the user's vision, making it difficult to navigate.

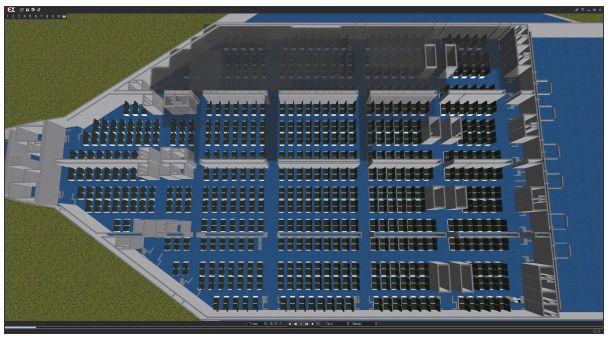
In the Figures 4.30, 4.31, 4.32 and 4.32 the CFD images do not show smoke visibility, but they show an iso-surface visualisation capture of smoke mass fraction. The value of the iso-surface is at 0.0001 kg/kg (mass fraction of smoke (kg) per unit mass of air (kg)) in all captures. This results in some discrepancies between the visualised smoke in the VR view and the CFD view. However, it is interesting to see the progress of the smoke in the CFD fire simulation in comparison to the progress in the VR. In the Figures 4.30, 4.31, 4.32 and 4.32 the ceiling is removed to enable the trainer (or the user testing procedures) to visualise the smoke spread in the geometry. However, the immersed user (i.e. the participant, who is different from the trainer) does not have the ceiling removed and will be unaware of any spectators assessing the situation from above or within the geometry.



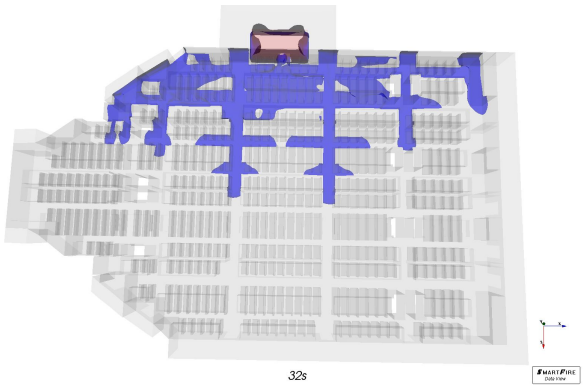
(a) VR view: Smoke progression at 8.4 seconds into the simulation



(b) CFD view: Smoke progression at 8 second timestep of the simulation



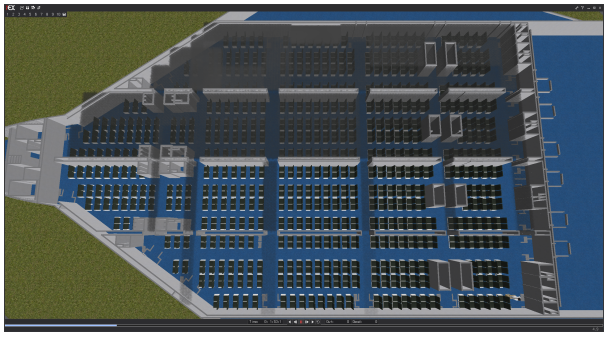
(c) VR view: Smoke progression at 31.3 seconds into the simulation



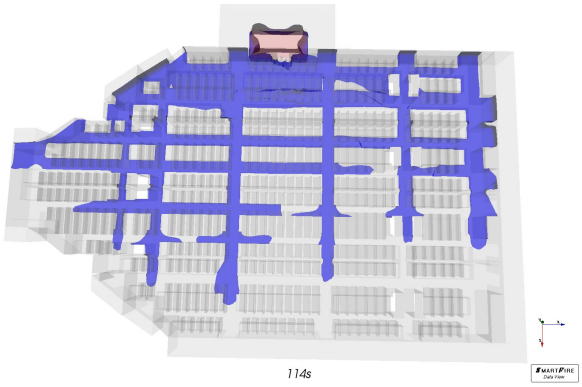
(d) CFD view: Smoke progression at 32 second timestep of the simulation

Figure 4.30: NACRE aircraft: Side angle smoke progression visualisation in vrEXODUS

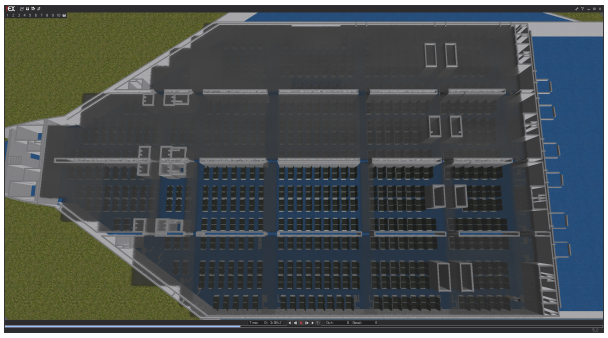
In Figure 4.30 shows the smoke entering the aircraft from the wing fire (see also Figures 4.22 and 4.23). Figure 4.30 also shows that the first two aisles on the right-hand side of the aircraft are covered in smoke in about 31 seconds into the simulation.



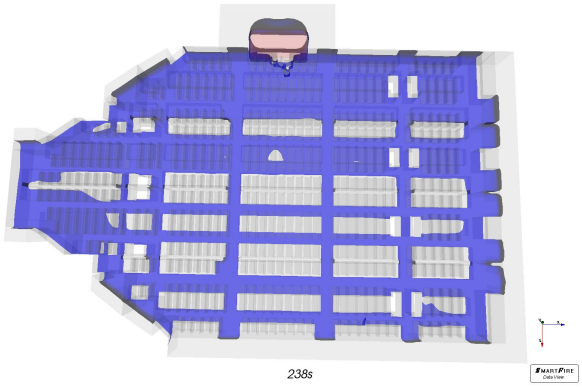
(a) VR view: Smoke progression at 1 minute 53.1 seconds into the simulation



(b) CFD view: Smoke progression at 114 second timestep of the simulation



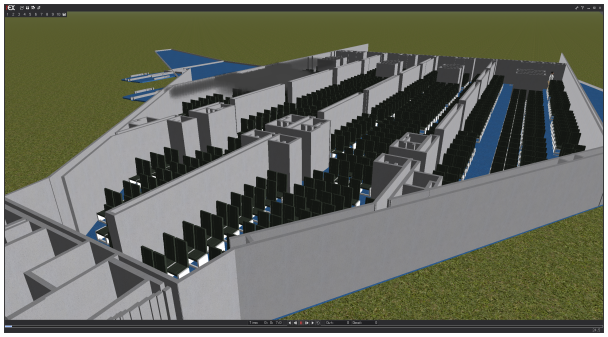
(c) VR view: Smoke progression at 3 minutes 58.2 seconds into the simulation



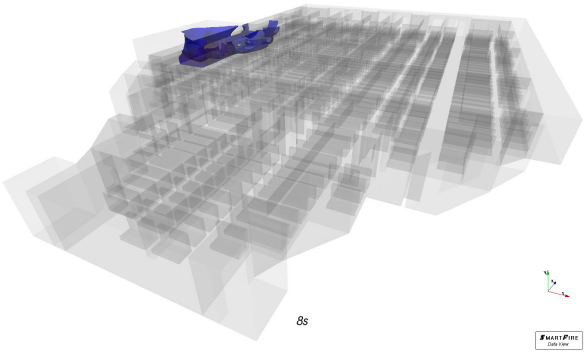
(d) CFD view: Smoke progression at 238 second timestep of the simulation

Figure 4.31: NACRE aircraft: Side angle smoke progression visualisation in vrEXODUS

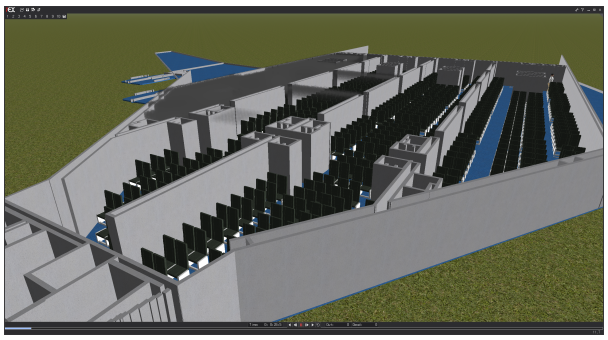
Figure 4.31 shows that about half of the aircraft is filled in smoke about 1 minute and 53 seconds into the simulation and after about 4 minutes the whole aircraft is filled with smoke.



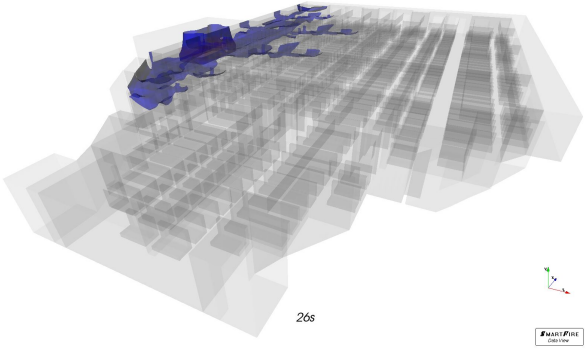
(a) VR view: Smoke progression at 7.0 seconds into the simulation



(b) CFD view: Smoke progression at 8 second timestep of the simulation



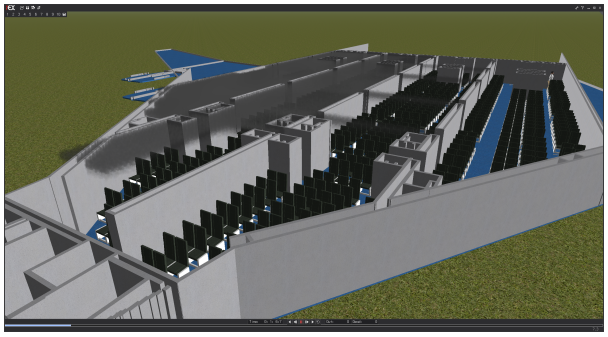
(c) VR view: Smoke progression at 26.5 seconds into the simulation



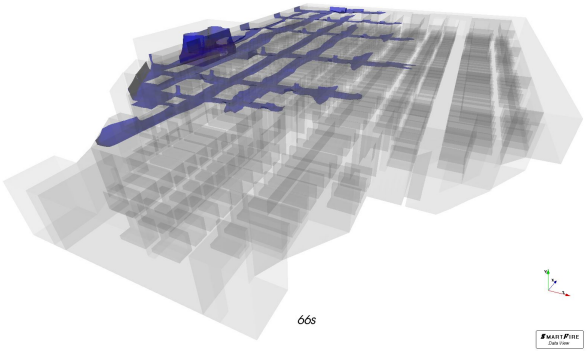
(d) CFD view: Smoke progression at 26 second timestep of the simulation

Figure 4.32: NACRE aircraft: Top down angle smoke progression visualisation in vrEXODUS

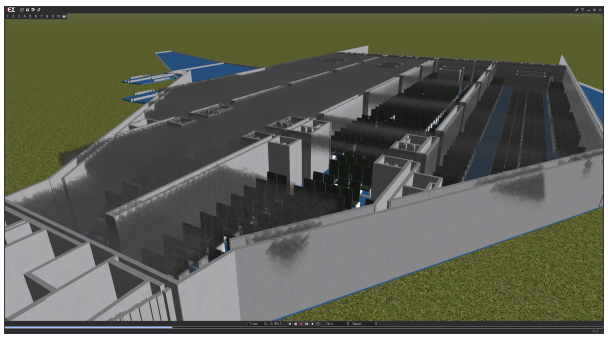
In Figure 4.32 also (compare to Figure 4.30) shows the smoke entering the aircraft from the wing fire (see also Figures 4.22 and 4.23). Figure 4.32 also shows that the smoke spreads through the doorways in the separating wall from the first zone on the right-hand side of the plane into the second zone in just after 26 seconds into the simulation. However, the first zone is almost fully covered in smoke until the smoke enters the second zone. A passenger would be able to see the smoke coming through the doorway as it shoots across the zone before it starts spreading and filling to the rest of the second zone.



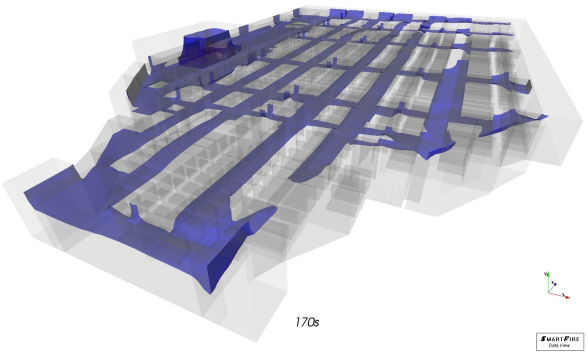
(a) VR view: Smoke progression at 1 minute 6.7 seconds into the simulation



(b) CFD view: Smoke progression at 66 second timestep of the simulation



(c) VR view: Smoke progression at 2 minutes 49.3 seconds into the simulation

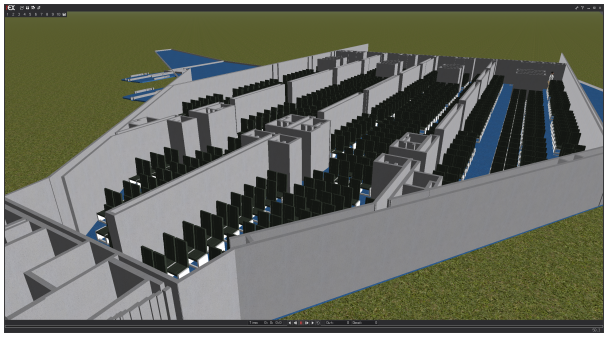


(d) CFD view: Smoke progression at 170 second timestep of the simulation

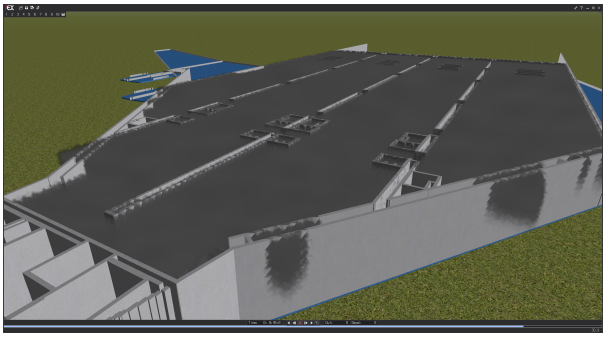
Figure 4.33: NACRE aircraft: Top down angle smoke progression visualisation in vrEXODUS

It is interesting to see that the smoke continues to spread in similar fashion in Figure 4.33 as it did in Figure 4.32 by shooting through the door way across the ceiling.

Figure 4.33 shows how the smoke fills the front of the plane in the third zone before it fills the rest of that zone and all exits are covered in smoke in under 3 minutes.



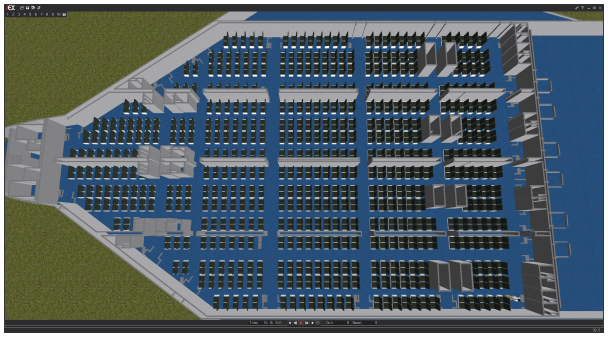
(a) Beginning of the simulation, before smoke starts filling the geometry



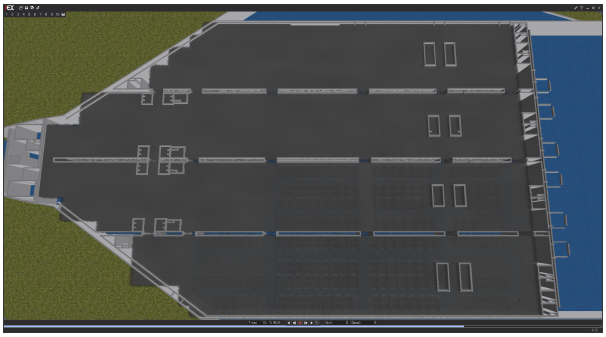
(b) Smoke has entirely filled the geometry

Figure 4.34: NACRE aircraft: Side angle smoke visualisation in vrEXODUS as empty and full

Figures 4.34a and 4.34b show the aircraft without ceiling from the front: without smoke and with smoke respectively. It is possible to see the smoke seeping through the walls. This is an output file limitation of the 2-layer and 10-layer smoke from SMARTFIRE as it is only capable of exporting smoke averaged over rectangle areas.



(a) Beginning of the simulation, before smoke starts filling the geometry



(b) Smoke has entirely filled the geometry

Figure 4.35: NACRE aircraft: Top down angle smoke visualisation in vrEXODUS as empty and full

Figures 4.35a and 4.35b show the aircraft without ceiling from above: without smoke and with smoke respectively. Due to the existing file format limitation

This demonstration case has shown:

- The importance of fine granularity of smoke when it is used for visibility representation. This is evident by comparing Figures 4.26 and 4.27 to see how the visibility in smoke

is gradually improved when the user is getting closer to the floor instead of the smoke suddenly clearing away.

- The possibility of using real-time calibrated and validated visibility in smoke rendering to effectively hinder the navigational capabilities in a virtual reality. Figure 4.27 demonstrates the visibility being reduced when the user is immersed in the smoke. It is difficult for the user to navigate in Figure 4.27 unless the user is crawling.

- Using accurate smoke representation to provide visual cues to the users so that they can take actions in response to the smoke. Figure 4.29 and other figures in general show how the smoke progression can be observed within the aircraft.

4.9 Smoke rendering performance tests

This section provides benchmark results for the real-time smoke rendering algorithm detailed in this chapter. The smoke was implemented in a C++ application and rendering is done using OpenGL. The smoke is loaded onto the graphics card using display lists and then rendered in multiple passes (see Section 4.4). The smoke is coloured using simplex noise algorithm as described in Section 4.6. The time to render the smoke as part of the rendering loop is recorded in microseconds using the C++11 chrono standard library (the time includes displaying it to the screen).

Table 4.2 lists the performance results when rendering a varying number of particles, ranging from 100,000 to 1,000,000. The columns list the rendering times in microseconds (μs) and in frames per second (FPS) for each particle count. PC 1 is an Intel Core i7 5930K processor, 64GB of 2132MHz DDR4 memory, NVIDIA GeForce GTX TITAN X graphics card with driver version 431.36. PC 2 is an Intel Core i9 9900X processor, 128GB of 2132MHz DDR4 memory, NVIDIA GeForce RTX 2080 Ti graphics card with driver version 431.36. Both PCs run at stock speeds (i.e. no overclocking). PC 1 ran Windows 7 Professional with Service Pack 1. PC 2 ran Windows 10 Enterprise version 1709. The test computers are used in software development and are not clean installations. The tests were performed after a reboot and without opening any user programs to run in parallel. However, the installed

programs running the background were likely affecting the results negatively.

Table 4.2: Smoke rendering performance results on 2 different PCs

Number of particles	μs (PC 1)	FPS (PC 1)	μs (PC 2)	FPS (PC 2)
100,000	12233	81.7	9737	102.7
200,000	31091	32.2	13811	72.4
300,000	50590	19.8	23888	41.9
400,000	70631	14.2	33215	30.1
500,000	85027	11.8	43402	23.0
600,000	108507	9.2	54029	18.5
700,000	126972	7.9	63145	15.8
800,000	145462	6.9	73554	13.6
900,000	163213	6.1	85666	11.7
1,000,000	180122	5.6	92720	10.8

In comparison to previous real-time smoke rendering performance tests done by Xu et al. [6] the smoke rendering proposed here is not only more accurate, but also over an order of magnitude more efficient (see Section 2.2.1 Chapter 2). It is also important to note that these tests were done using a now old graphics library, OpenGL, and a newer graphics library such as Vulkan is likely going to improve the results further.

4.10 Summary

This chapter has shown how graphics fidelity can be improved significantly by using higher detailed models (see Section 4.2) and accurate smoke rendering. The new efficient and accurate real-time rasterising smoke rendering technique was developed from an existing ray tracing technique and detailed in this chapter. The smoke rendering technique features a level of detail where graphical fidelity can be scaled up or down during runtime. It was calibrated and its accuracy was validated with multiple external users in Section 4.7.

The smoke rendering technique proposed in this chapter can also be used immediately in game engines such as Godot[9], Unity3D[10] and Unreal Engine[11]. This enables realistic and immersive real-time smoke visibility dependent applications beyond the proposed IRIS system.

The smoke validation case in this chapter provided a novel way of validating a rasterised smoke visibility rendering in a real-time virtual environment against the industry standard smoke visibility research. This smoke visibility validation aimed to replicate the original smoke visibility conditions of Jin's smoke visibility research [18], which is the current state of the art smoke visibility research as identified in Chapter 2.

The accurate visibility in smoke representation demonstrated in the aircraft demonstration case does provide key visual feedback to any user of the IRIS system, through either hindering the visibility of the user or providing visual cues of the current state of the situation. When training and practising with correct visual feedback, the trainee will be able to translate the learned expertise from the virtual world to real world scenarios.

The next chapter will focus on user interaction capabilities and how they are handled in this research.

[9]Godot - Spatial Material `https://docs.godotengine.org/en/3.1/tutorials/3d/spatial_material.html#blend-mode`

[10]Unity3D - ShaderLab: Blending `https://docs.unity3d.com/Manual/SL-Blend.html`

[11]Unreal Engine - Material Blend Modes `https://docs.unrealengine.com/en-US/Engine/Rendering/Materials/MaterialProperties/BlendModes`

Chapter 5

USER INTERACTION

Chapter 2 identified some key behaviours on how people react to gestures and commands especially when given by a person perceived as an authority (see Section 2.1.3). Chapter 2 also discussed the current understanding of crowd behaviour and how it has improved over the years (see Section 2.1.2). Some key findings include the understanding of the individual behaviour within the crowd that becomes the collective behaviour of the whole crowd, instead of individuals losing the sense of individuality and behaving like an ant colony (see Section 2.1.2) or a set of magnets (see Section 2.3.5). The literature review also highlighted past research into interfacing humans with computers to provide an interface with virtual objects and characters (see Section 2.3.7) and the importance of providing natural information and abstaining from artificial information (see Section 2.3.8).

To enable the interaction between the users-controlled agents and computer-controlled agents (i.e. simulated pedestrian model) for the IRIS system, it is necessary to provide the user with a set of plausible interaction capabilities as identified in Chapter 2. This chapter proposes how the user can use hand gestures and voice commands within a VR environment to interact with computer-controlled agents, and how those agents should react to the user's hand gestures and voice commands. This is the third research contribution in this thesis as described in Section 1.3.3, Chapter 1 and is further detailed in this chapter.

A set of verification cases that aim to verify the expected functionality of the capabilities proposed in this chapter are performed in Sections 5.2 to 5.9. The sections aim to verify the expected agent behaviour to user commands and gestures based on the verification definition by the ISO (International Organization for Standardization). The ISO definition of verifi-

cation is *"the process of determining that a calculation method implementation accurately represents the developer's conceptual description of the calculation method and the solution to the calculation method"* [176, 177].

The user interaction capabilities with a simulated crowd are tested first. It is important to verify the behavioural changes introduced to the computational pedestrian model, before they are used in a more complex environment. This assures the user that the expected behaviours introduced to the simulation will perform as expected, when they are first tested in an isolated environment.

The first verification test in Section 5.2 will verify the implemented line of sight test on a horizontal and a vertical axes. This is then expanded to include a set of crowd tests. However, the base line must be formed for the crowd test case scenario, before the individual user interaction commands can be verified to function correctly by altering the base case scenario. Each user interaction command is verified in its own isolated environment.

5.1 User interaction capabilities

To enable user interaction with simulated computer-controlled agents in the IRIS system, it is necessary to update the EXODUS software to allow user-controlled agents to issue commands to the general circulating civilian population (i.e. the computer-controlled agents), to influence their behaviour. This section proposes 8 different commands for the user-controlled agent to use in managing individuals and crowds of computer-controlled agents. The 8 commands were identified as part of the AUGGMED project (see Section 1.6) from interviews with the armed police officers in West Yorkshire Police. The commands identified by the officers will provide a representative set of commands for both crowd management and individual agent instructions. Each command can be issued by user-controlled agents as voice commands, hand gestures or both. The 8 implemented commands can be seen in Table 5.1.

Table 5.1: Crowd control commands

Command	**Description**
Stop	The agents should stop.
Go	The agents should carry on what they were previously doing. If it was stationary, they should start evacuating.
Get down	The agents should crouch.
Get up	The agents should start standing.
Start evacuating	The agents should start evacuating.
Evacuate in direction	The user looks into the direction where the agents should be evacuating.
Get out of the way	The agents should make room for the user, by moving to the sides.
Move over there	The user points at a location on the floor to indicate a point where the agents should go.

When the user-controlled agent issues any of the 8 commands in Table 5.1, the user is informed of this by a message which appears on the screen indicating the command issued. In this research the commands are given by using a keyboard and mouse. However, there is no reason why the commands cannot be given in any other way using any other device or even captured from a video or audio. Exploration for the most suitable way of issuing commands and gestures is left as future research and for simplicity the keyboard and mouse are used as the controllers in this research. Commands such as *"Evacuate that way"* or *"Move over there"* and *"Get out of the way"* depend on the direction and location that the user is pointing with their mouse which is the direction the character of the user looks.



(a) Stop command



(b) Go command



(c) Get down command



(d) Get up command

Figure 5.1: Stop, Go, Get down and Get up commands shown in vrEXODUS

Figures 5.1 and 5.2 show the influence areas of the voice commands in red and the influence areas of the hand gestures in blue. The influence areas are shown in the screengrabs in this thesis to help visualise the effective range of the issued commands. The immersed user (i.e. the participant) does not see the influence circles in a real use case to avoid artificial behaviour (see Chapter 2) and the areas are only visible to the coordinator. The red line is a circle around the user and the blue line is a semi-circle around the user (the shapes can be seen in Figure 5.3).



(a) Start evacuating command



(b) Evacuate in direction command



(c) Get out of the way command



(d) Move over there command

Figure 5.2: Start evacuating, Evacuate that way, Get out of the way and Move over there commands shown in vrEXODUS

The 8 commands can be issued to either individual agents (i.e. the currently selected agent within the VR), or if no agent was currently selected, to everyone within the area of influence.

In order to accurately represent the ability for user-controlled agents to issue commands, two main issues need to be addressed, these include:

- The representation of the area over which the commands can be heard/comprehended (i.e. the calculation of control areas).
- The likelihood of agents who do hear/comprehend the command obeying it.

The published literature reviewed in Chapter 2 established that for a person to obey the instructions from an authority, the person must be within the range of being able to comprehend the authority and needs to be able to see the authority. The reviews also established that for gesture-based communication, the target audience should be in front of the person communicating using gestures to understand that he is instructing them. The literature (see Chapter 2) also shows that in emergency scenarios the instructions from authority are almost always followed, even though they may not be followed immediately, but are followed eventually.

It is also worth reiterating that for commands to be issued as part of an immersive virtual environment, it is a requirement for all the necessary calculations to be performed in real-time. If the simulation would halt when any of the commands were issued due to the calculations being performed, it would negatively affect the realism and the immersion of the IRIS system, thereby deeming the system unfit for purpose and hence unusable. Therefore, it is important to note that the algorithms outlined below defining the manner in which commands are issued to agents are faster than real time.

5.1.1 Control areas

As stated in the previous section, any of the 8 commands implemented can be issued either verbally (i.e. a voice command) shown in red in Figure 5.3, visually (i.e. a hand gesture) shown in blue or both. Figure 5.3 demonstrates the shapes and directions of the hand gesture and voice command areas and how they overlap each other. Since the sound of an issued command would in reality travel in all directions, this is modelled as a circle around the user-controlled agent but is dependent on the line of sight between the agent issuing the command and the computer-controlled agents receiving the command

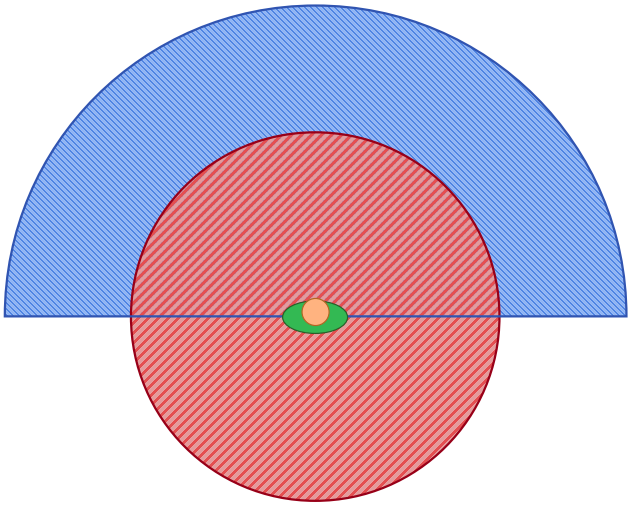


Figure 5.3: Hand gesture (blue) and voice command (red) influence areas

(see Figure 5.3). The computer-controlled agents located behind the user-controlled agent issuing the command are assumed be able to hear the command, and potentially obey it as long as they can see the agent issuing the command. In a real-world situation, the range over which a voice command could be heard and comprehended would be dependent on how loud the command was issued (i.e. shouted), the loudness of the ambient background noise [185], the facing direction of the person issuing the command and whether the hearing was affected by incidents prior to the issuing of the command (i.e. an explosion in close proximity causing partial or complete loss of hearing). However, without capturing the voice level from the user using the IRIS system and mapping it against real-world data and modelling the hearing levels in the computational pedestrian model on user-controlled agents, it is difficult to know how far away a user-controlled agent's voice could be comprehended. When reviewing the published literature such data was not found and therefore the range for the voice command was set to a fixed distance for this research. The fixed distance can be defined by the user, but a default value of 8 metres is used as it provided the desired level of response and influence similar to the reviewed literature (i.e. the agents can comprehend the issuer in his near vicinity, see Section 2.1.2 in Chapter 2) when tested. A 3D view of the control areas can be seen in bird's-eye view in Figures 6.9 and 6.20 in a later chapter (Chapter6).

As was discussed in Chapter 2, Dugdale et al. [12] concluded based on field studies that *"it is important that the user is able to drive the virtual person in such a way as to produce the appearance of orientating to other virtual persons. This could, for example, signal to another virtual character that they are looking at each other. Looking at another may be a device to judge whether a gesture is necessary."* [12]. Dugdale et al. [12] suggest that the facing direction of the user-controlled agent should be used to indicate whether or not the user-controlled agent is communicating non-verbally (i.e. using a hand gesture) with other agents. Therefore, the zone of influence (the catchment area) for a hand gesture is modelled as a semi-circle and centred at the agent issuing the command. The semi-circle is always facing the same direction as the agent issuing the command. As a result, only computer-controlled agents located in front of the agent issuing the command can be influenced using the hand gesture.

In a real-life situation, there are a number of factors that would influence the comprehension of the hand gesture, such as the population density in front of the person issuing

the hand gesture as well as the visual acuity of the person interpreting the hand gesture (as identified in Chapter 2). For this research the hand gesture is dependent on the line of sight between the agent issuing the hand gesture and the agents interpreting the hand gesture in a similar manner to the voice commands. However, the line of sight is limited in both the hand gestures and the voice commands to geometry objects, such as walls and pillars and does not take into account the possibility of other agents obscuring the vision. Using the line of sight calculation with the geometry objects is already computationally expensive (as discussed in Chapter 6) and therefore adding line of sight calculations for the surrounding and moving agents would require more performance optimisations to be feasible in real-time.

The line of sight calculations are performed in EXODUS and it does not represent agents in enough detail to make reasonable assumptions on how much an individual agent obstructs a given object. To make a better estimate, all agents would need to be rendered in 3D as seen by other agents to determine how much those rendered agents are obstructing visibility. The rendered image would then need to be analysed in real-time to determine the obstruction levels of the rendered agents and objects. This could be computationally problematic and has been left as future work.

Similarly, to the voice commands discussed earlier, due to the lack of published data on the hand gesture influence areas and range, the maximum hand gesture influence range is set to a fixed value in this research. The fixed distance can be defined by the user, but a default value of 15 metres is used as it provided the desired level of response and influence similar to the reviewed literature (i.e. the agents can comprehend the issuer in his near vicinity, see Chapter 2) when tested.

The hand gesture and voice command ranges represent the maximum distances at which agents can clearly hear and comprehend the commands issued. However, the line of sight calculations of the influence and catchment areas of a command take into account the obstacles and walls that are present within the geometry. The line of sight calculation also takes into account the height of the obstacles and walls, which means that computer-controlled agents that are within the range of the agent issuing commands may not be able to comprehend the command if the obstacle or wall is too high. As was established when reviewing published literature in Chapter 2, it is necessary for a person to see the authority to be able to be

influenced by them. Therefore, even if an agent could hear the agent issuing commands, but cannot see them, the agent is not being influenced by the authority. As a current limitation, agents do not also investigate the source of the command when they cannot see it. The user can set the heights of individual walls and obstacles in EXODUS when building the geometry and define the minimum height of walls and obstacles that will obscure the vision between agents. If a wall or obstacle is below the minimum obscuration height, defined by the user, then the agents are assumed to be able to see each other over it (i.e. the wall is not taken into account in the line of sight calculations). When the height of the wall or obstacle exceeds the minimum obscuration height set by the user, the wall or obstacle is assumed to be able to block the vision between agents and is taken into account in the line of sight calculations. Currently the line of sight calculation is performed on a 2D horizontal plane for increased performance in comparison to a 3D line of sight calculation. A 3D line of sight calculation would be necessary if the heights of the compared agents are taken into account with position and height of the wall. The 3D line of sight calculation would therefore produce more accurate results, but for demonstration purposes of the IRIS system and its capabilities, the 2D line of sight calculations are used in this research. The default obscuration height of obstacles and walls is set to 1.5 metres.

5.1.2 Likelihood of agents obeying a command

The effectiveness of a command (i.e. the probability of agents obeying the command within its catchment area) is determined with a compliance probability value. The literature suggests (see Chapter 2) that the in-emergency scenarios, the evacuees are compliant to the authority to the point that they almost always (with rare exceptions) follow the instructions given by the authority. However, the compliance is not always immediate and is influenced by the surrounding people, such as the observed actions taken by the other people. The literature also suggests (see Chapter 2) that for a person to be influenced by the authority, the person has to be within the authority's vicinity, be able to comprehend the instructions and when the person is beyond a certain distance, he no longer follows the authority. Therefore, the probability for an agent to obey a command when being within the influence area of the issued command is summarised into the following two factors:

- Their distance from the agent issuing the command.
- The number of other agents currently obeying the command.

To achieve the effects described in this chapter and in Chapter 2 a theoretical model was developed for EXODUS to represent these behaviours. The model allows the calculation of the probability of a computer-controlled agent to obey a command, in a time dependent manner (probability/second) for both hand gestures and voice commands. The user can define the probability of computer-controlled agents to obey the command at 0 metres and at the maximum radius for each command for both hand gestures and voice commands. The probability of computer-controlled agents obeying the command is adjusted linearly between the probabilities at maximum distance and 0-metre distance from the agent issuing the command. The user can also define the maximum radius for each command. If the user-controlled agent issues both the hand gesture and voice command at the same time, both commands are calculated and modelled independently and therefore compliment each other by increasing the total probability value of computer-controlled agents to obey the issued command. This is based on the principles outlined in the research by Cassell et al. [126].

As mentioned previously, the probability of an agent to obey a command is also dependent on the number of other surrounding agents currently obeying the command. The influence of the other surrounding computer-controlled agents on the individual agent's decision to obey the issued command is represented by a *Compliance Factor* which effectively determines how much more likely an agent is to obey the command. The more agents obey the command (as determined by the distance-based probabilities), the more the *Compliance Factor* increases until it reaches a maximum value when all computer-controlled agents within the influence zone obey the command. This allows the model to factor in a social influence factor, which has been identified by other researchers [186–188].

To find suitable default values for the probabilities of computer-controlled agents to obey commands at 0 metres and the maximum metres and for the compliance factor, a number of tests were performed to reach the *near full compliance* of following instructions from an authority in emergency scenarios as suggested by the reviewed literature in Chapter 2. It is also important to verify that not all computer-controlled agents obey the commands

immediately, and more agents will gradually start to obey the command (as influenced by the surrounding agents) until all agents obey the command if the user-controlled agent keeps issuing the command. The compliance, behaviour and response to commands are verified in Sections 5.3 to 5.9.

When no agents within the influence zone are obeying the command, the *Compliance Factor* is set to 1, and therefore, the agents in this case do not have an increased probability to obey the command. However, when all agents in the influence zone are obeying the command the *Compliance Factor* is increased to the maximum compliance factor value defined by the user (the default set to 3), in which case the agents are three times more likely to obey the command.

From the VR client, the user-controlled agents can issue a combined hand gesture and a voice command at any given time within the simulation. For this research the hand gestures and voice commands are both issued at the same time when user wants to issue a specific command. This helps to avoid over complicating the controls for the user by limiting the set of separate options to initiate individual types of command (i.e. hand gesture or voice command). In real-life emergency situations hand gestures and voice commands are also often issued simultaneously as was discovered in Chapter 2.

Once a given command is issued by a user-controlled agent a message is sent from vrEXODUS (the client) to EXODUS (the server) (see Section 3.3.15) that includes information regarding which agent issued the command (as defined by their unique *Agent Id*), their location (i.e. X, Y and Z coordinates), the direction in which the agent is facing, the time duration (in seconds) that the command is issued for and the type of command being issued (i.e. get down, get up, stop, go etc.). The message also includes information defining how the command is to be issued, such as the maximum radii for both the hand gesture and the voice command, the probabilities/sec that an agent will obey the command at both 0 metres and at the maximum radii and also the *Compliance Factor* value when all agents within the influence areas are obeying the command. See Section 3.3.15 for the full list of parameters.

Table 5.2: Command attribute definitions

Attribute	**Description**
Duration	How long the command was issued for (in seconds).
Min Agent Delay	Minimum amount of time (in seconds) an agent will continue to obey the command.
Max Agent Delay	Maximum amount of time (in seconds) an agent will continue to obey the command.
Radius	The maximum reach of the command (in metres).
Close Probability	The number of agents / second obeying a command at 0 metre distance from the issuer.
Far Probability	The number of agents / second obeying a command at the maximum radius from the issuer.
Compliance Factor	Coefficient of how many more times likely an agent is to obey a command when others around them are already obeying the command.

The command (i.e. hand gesture and voice command) specific attributes are listed in Table 5.2. These are the command attributes used throughout this thesis.

Once the command message information has been received by EXODUS (the server), it then calculates the corresponding influence areas for both the hand gesture and the voice command, before attempting to issue the command to the corresponding agents within the influence areas. It is important to note that the probability of a computer-controlled agent obeying the command is based on their distance from the agent issuing the command and the number of other computer-controlled agents within the influence zone that are already obeying the command. Hence, agents are not guaranteed to obey any issued command immediately, especially if they are located a large distance from the agent issuing the command and no other agents have currently obeyed it. Agents who do obey a command may take different amounts of time to comply, due to the probabilistic manner in which their compliance is determined.

vrEXODUS (the client) will also send regular updates to EXODUS (the server) to inform the simulation about the facing direction of the user-controlled agent (see Section 3.3.16). As the user-controlled agent changes the facing direction in vrEXODUS (the client), the facing direction is sent to EXODUS (the server), which also results in the change in direction of the influence areas in the simulation. The user-controlled agent will continue to issue the command in the simulation for the specified duration (the default is 3 seconds but can be changed if more data becomes available or a distribution could be applied). Once the duration expires the user-controlled agent stops issuing the command and will not issue any further commands until instructed to do so by the user through vrEXODUS.

5.2 Verification of catchment area and line of sight

Before more complicated interactivity capabilities can be tested and verified, it is paramount to establish that the computer-controlled agents can successfully react to the user-controlled agent's issued commands, and that this happens within the catchment areas of the commands while taking into account the line of sight. The aim of this verification test is to verify that the computer-controlled agents will respond to the commands given by the user-controlled agent if they pass within a line of sight. All obstacles in buildingEXODUS are represented with walls that have user defined heights. Therefore, when using buildingEXODUS as the evacuation model in the IRIS system only walls can be tested as obstacles.

This verification test verifies the catchment area for hand gestures and voice commands. This test features two long corridors A and B, with an agent on each corridor, a joint exit and a user area.

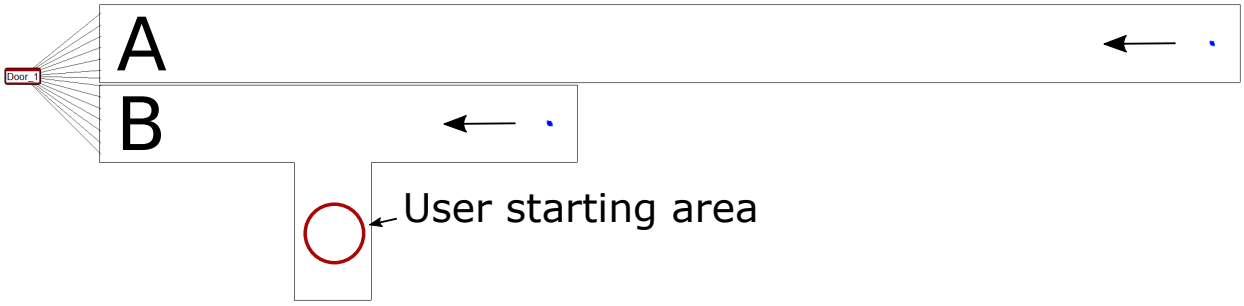


Figure 5.4: Starting point, where agents A and B are preparing to move towards exit Door_1

For the user to have time to join the simulation, get oriented and still have enough time to start issuing commands: it is necessary to make the walking distances for the computer-controlled agents long enough so that they do not reach the visible area in front of the user, until the user is ready. The second corridor is significantly longer than the other to make sure that both computer-controlled agents do not pass the user's location at the same time. This will also enable highlighting the fact that stopping the computer-controlled agent with the shorter exit route may cause the computer-controlled agent to exit the structure slower than the computer-controlled agent that has significantly a longer exit route, depending on whether the user-controlled agent successfully interacts with the targeted agent.

Corridor A is 50 metres long and is connected to an exit in one end. Corridor B is 21 metres long and is also connected to an exit in one end. The two corridors are separated by a 2 metres high wall, which is taller than the required 1.5 metres to block the gesture catchment area. Both corridors are 3.5 metres wide to allow the computer-controlled agent to move sideways if deemed necessary by the simulation, which should not happen as part of this verification test. Each corridor features a single agent, where both agents will start evacuating immediately at the start of the simulation. The user is placed in corridor B in its designated area (see Figure 5.4). The user will issue Stop command via a hand gesture and a voice command simultaneously.

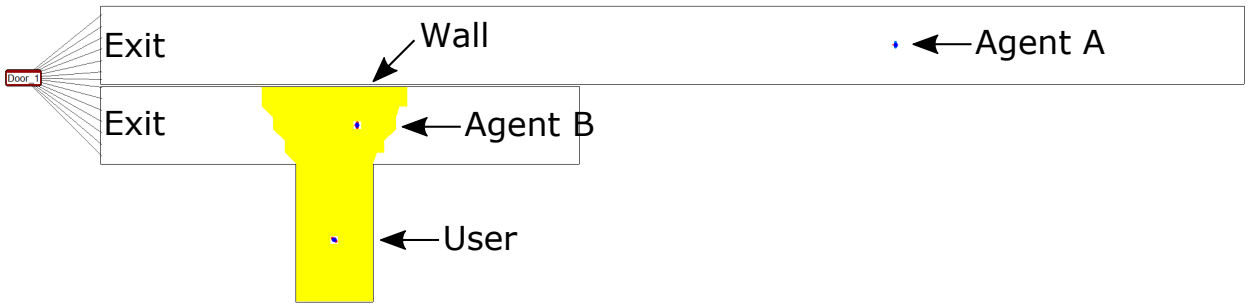


Figure 5.5: EXODUS view: line of sight verification case. Agent B has stopped in the gesture influence area. The yellow area shows the combined area of both the hand gesture and voice command.

Figure 5.5 shows the combined catchment area of both the hand gesture and the voice command. The user-controlled agent issues the command before the simulated agents enter the visible catchment area of the commands. The agent in corridor B reacted to this command by stopping. The second agent in corridor A does not reach the catchment area of the user's

command and will continue to evacuate until it reaches the exit. However, as the agent in corridor B reacted to the issued command, it will continue to follow the command as long as the user continues to issue the command. Once the user stops issuing the command or issues another command, the agent will start moving after the specified delay on the issued command (see Table 5.3). This test was repeated at least 10 times and the results did not change (i.e. the agent in corridor B always stopped, but the agent in corridor A never stopped).



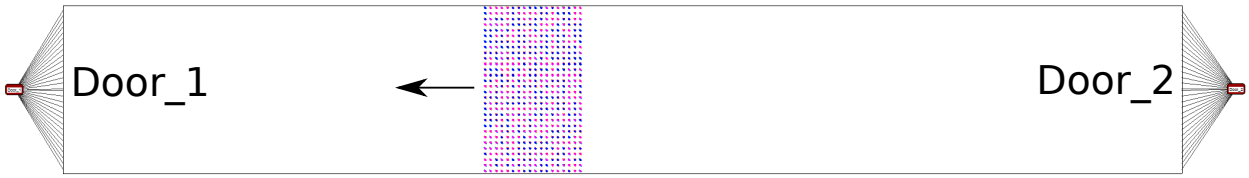
Figure 5.6: 3D view: User can see the agent in corridor B, but cannot see the agent in corridor A.

The view from the point of view of the user can be seen in Figure 5.6. The user cannot see the agent in corridor A and therefore the agent in corridor A is oblivious of the commands the user is issuing in corridor B. However, the agent in corridor B clearly stopped in front of the user.

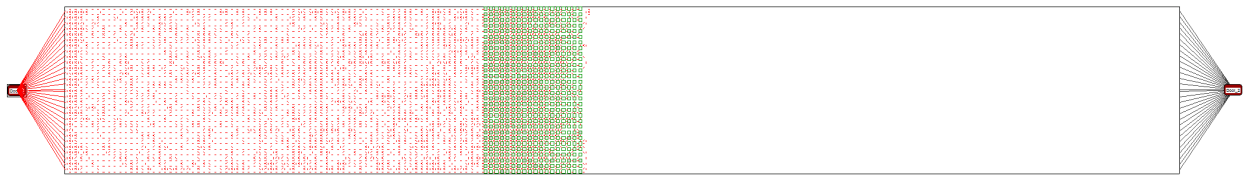
5.3 Base case for hand gesture and voice command verifications

This verification case aims to test redirection of a crowd as a whole as well as an individual agent. The pedestrian behaviour model uses the default attribute distribution set in buildingEXODUS. This is a discrete model with a default array of attributes set for a crowd. These are randomly generated by buildingEXODUS from its default table. The aim of this test is not to test all the intricacies of the model itself, but to verify whether or not it reacts to the given gesture commands.

It is important to define a base case that will be used to compare the results of each of the hand gestures and voice commands in the following sections. The base case sets the precedence for the agent behaviour. The purpose of each hand gesture and voice command verification case is to influence the crowd behaviour of the base case. By successfully altering the outcome of the simulation, either by influencing the paths of agents or the behaviour of the agents, in the intended way, the command is considered functional.



(a) EXODUS view: Crowd starting point. Door_1 is closer than Door_2.



(b) EXODUS view: Whole crowd exited through Door_1.

Figure 5.7: The default behaviour of the base case.

The base case is shown in Figure 5.7. The geometry is made of a corridor with 15 metres by 100 metres in dimension. Both ends of the corridor are connected to individual exits. This means that any agent can evacuate either through an exit called *Door_1*, or an exit called *Door_2*.

The corridor is populated with a crowd of 540 agents in 15 metres by 9 metres area. Pink coloured dots represent females and blue coloured dots represent males in the population. The crowd is located near the centre, but closer to *Door_1* than to *Door_2*. This exploits the default navigation system in EXODUS, the potential map. A potential map works by assigning a cost value to each available movement space incrementally from all exits (i.e. the further the movement space is from an exit the higher the cost). When evacuating without an itinerary, all agents follow the potential map, always preferring a lower cost movement space. There are some exceptions to this with advanced movement behaviours, such as shifting to a similar cost (within a set tolerance) movement space if it provides a route to overtake a queue, and swapping places with another agent that has a different level motivational characteristic. (These advanced behaviours produced the red movement paths behind the green starting locations in Figure 5.7b.)

The starting location of agents can be seen in Figure 5.7a with the exit names, and an arrow indicating the default evacuation direction for all agents.

Once the simulation starts, the alarm is also started. Once the alarm is started, a response time distribution derived from an actual evacuation of Marlowe Theatre that was purposely conducted and analysed [189] is used to randomly assign a response time to each agent. Based on the analysis of the Marlowe Theatre evacuation study, all agents will start evacuating between 0 and 180 seconds from the start of the alarm. The response times are distributed by a log-normal response distribution with a mean = 3.99 and standard deviation = 0.54. In a normal scenario the agents would continue to perform their tasks until their response time has been reached, at which point the agent starts to evacuate. However, in this test case each agent will continue to wait, by standing still, until their response time is up. Figure 5.7b shows the finished simulation with the starting location of each agent (in green) and the paths of the agents (in red).

Table 5.3: Shared command attributes

Attribute	**Value**	
Duration	3 seconds	
Min Agent Delay	20.0 seconds	
Max Agent Delay	30.0 seconds	
Radius	15 metres (hand)	8 metres (voice)
Close Probability	0.5 agents/second (hand)	0.8 agents/second (voice)
Far Probability	0.1 agents/second (hand)	0.2 agents/second (voice)
Compliance Factor	2.0 (hand)	3.0 (voice)

Table 5.3 shows the default values used in all the verification cases of different commands. The values are only postulated due to the lack of available data from the literature. However, the values produce reasonable and realistic looking results according to the feedback from the external participants as can be seen in Chapter 6. The probabilities should be replaced with calibrated values from experiments or trials when it becomes available. However, the above values have provided results which look reasonable (qualitatively) as can be seen later in the thesis in Sections 6.5.2, 6.6.2 and 6.7 in Chapter 6. The table follows the definitions laid out in Table 5.2.

5.4 Evacuate in direction command verification

This test will verify that simulated computer-controlled agents will follow instructions to evacuate in a given direction. These instructions are referred as *Evacuate in direction* in this thesis. The evacuation direction of the *Evacuate in direction* command is set by the facing direction network message of the user-controlled agent in Section 3.3.16. The voice command of the *Evacuate in direction* command is given as *Evacuate that way!*. When the hand gesture is given at the same time, the user-controlled agent is assumed to be pointing towards the direction the user is looking at. The implementation and the representation of the *Evacuate in direction* command is simplified from a real-world interaction, where the

authority giving the command would be more descriptive of the location and direction where people should go. A real-world example would be: “Follow the signs to exit B.” or “... turn left from the coffee shop.”

As soon as the simulation starts the user will start issuing the *Evacuate in direction* command in front of *Door_1*. Table 5.3 lists the attributes used for gestures in this verification case.

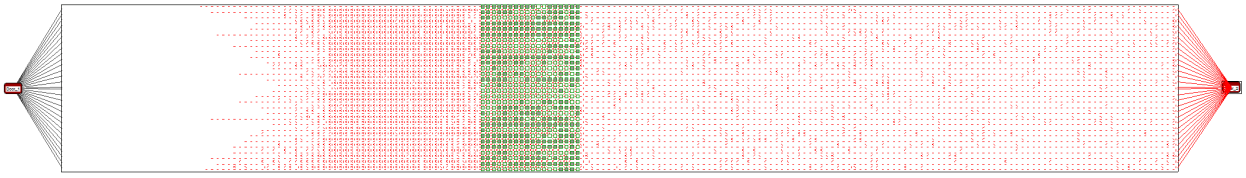


Figure 5.8: User standing in front of Door_1 redirected the crowd to Door_2.

The user is standing in front of *Door_1* and issuing the *Evacuate in direction* command to the approaching crowd. As the crowd approaches the user, the agents will enter the user’s command influence area and the EXODUS simulation will calculate the probability of each user’s probability to follow the given instruction. As can be seen in Figure 5.8: all agents did eventually follow the command the user was giving. Some of the stray red lines in Figure 5.8 show that not all agents followed the instruction immediately.



Figure 5.9: 3D view: Whole crowd is redirected to different exit.

Once the user has successfully redirected all agents to turn around and evacuate via *Door_2* instead of *Door_1*, the user will see a view show in Figure 5.9.

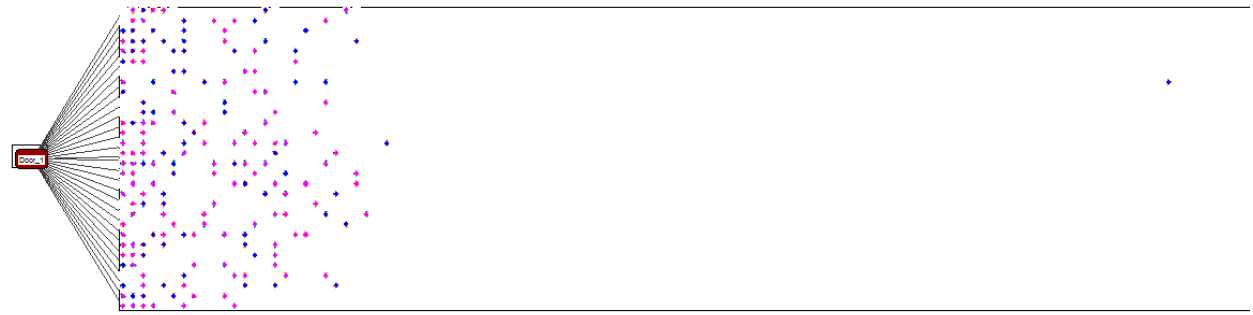
Individual agent command

The user may also issue the command to a selected computer-controlled agent. The selected computer-controlled agent knows that it is the only one the command applies to. However, the same probability calculations apply to the individual agent as it applies to a crowd. The agent will assess its vicinity for other agents obeying any command and determine whether it should follow the command given by the user (see Section 5.1.2).

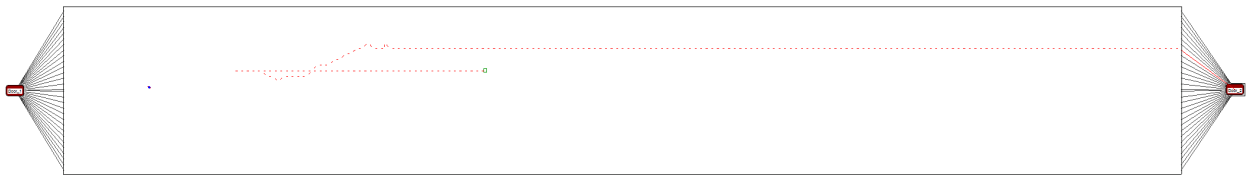


Figure 5.10: 3D view: Single user is selected and issued a command.

Figure 5.10 shows an agent that has been selected (blue arrow above the agent's head). In Figure 5.10 the agent has already reached the user's command influence area and has determined to follow the issued command and is thus already facing its back towards the user.



(a) Single user had been separated from the crowd.



(b) Redirection path.

Figure 5.11: Single user is redirected to Door_2.

Figure 5.11a shows the obedient agent already reaching past half way through the geometry towards *Door_2*. Figure 5.11b shows the entire length of the travelled path (in red) of the obedient agent, where the starting location is in green and the user's agent is still standing in front of *Door_1*.

5.5 Stop and Go commands verification

Some hand gesture and voice commands were developed to work in pairs (i.e. *Stop* and *Go* in this section and *Get down* and *Get up* in Section 5.6). The *Go* command instructs the targeted agent(s) to continue what they were previously doing, or if they did not have an existing itinerary and were just waiting around, they are informed to start evacuating. The *Start evacuating* command differs from the *Go* command by instructing the targeted agent(s) to start evacuating by abandoning their current task. *Stop* command instructs the targeted agent(s) to stop whatever they were doing and stop moving.

Table 5.3 defines the attributes used in this verification case. The user is standing in front of *Door_1* and will immediately start issuing the *Stop* command to all approaching agents from the beginning of the simulation.



Figure 5.12: 3D view: User issued a Stop command to all agents in the catchment area.

Figure 5.12 shows that some agents have already reached the command influence area and a subset of those have already stopped. As more agents start obeying the command, the likelihood of other agents in the influence area to obey the command will increase.



Figure 5.13: 3D view: User issued a Stop command to all agents in the catchment area.

Figure 5.13 shows that all the agents within the command influence area have obeyed the command as they are in the standing posture.

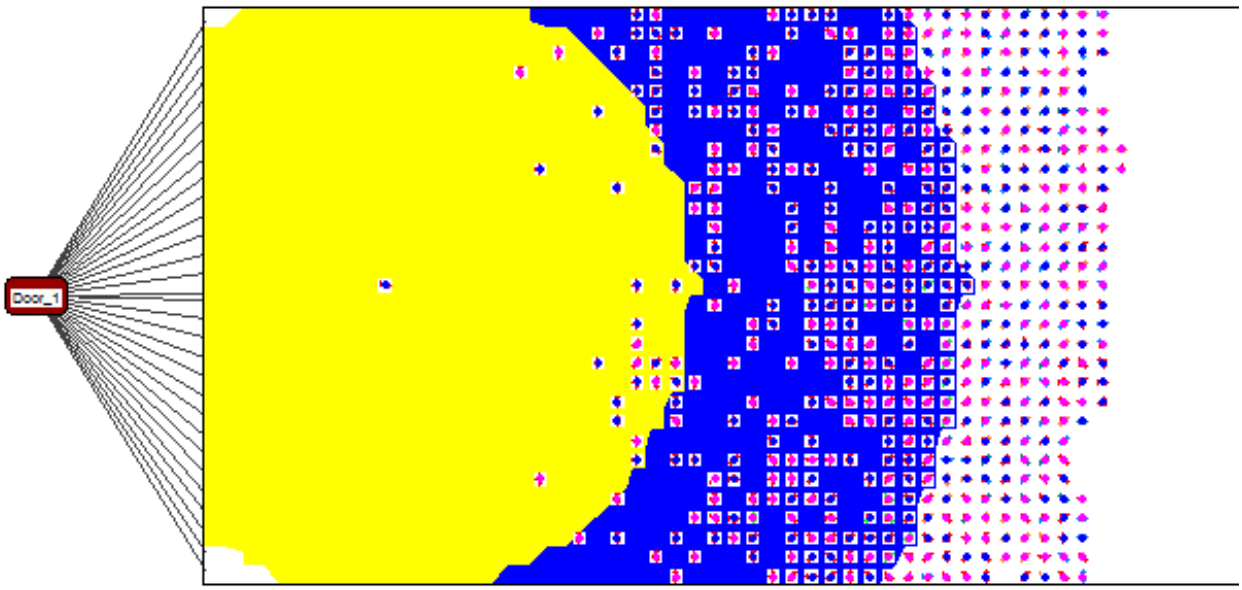


Figure 5.14: EXODUS view: User issued a Stop command to all agents in the catchment area.

Interestingly Figure 5.14 shows that not all agents have reached the influence area of the issued command. The obeying agents have effectively formed a wall and blocked the agents behind them thus preventing them from approaching *Door_1*. The user can only see what is happening in Figure 5.13 and does not know how large the crowd is behind these obedient agents that are trying to get through.

At this point in the test, when all agents are successfully prevented entering *Door_1*, the user starts issuing a *Go* command.

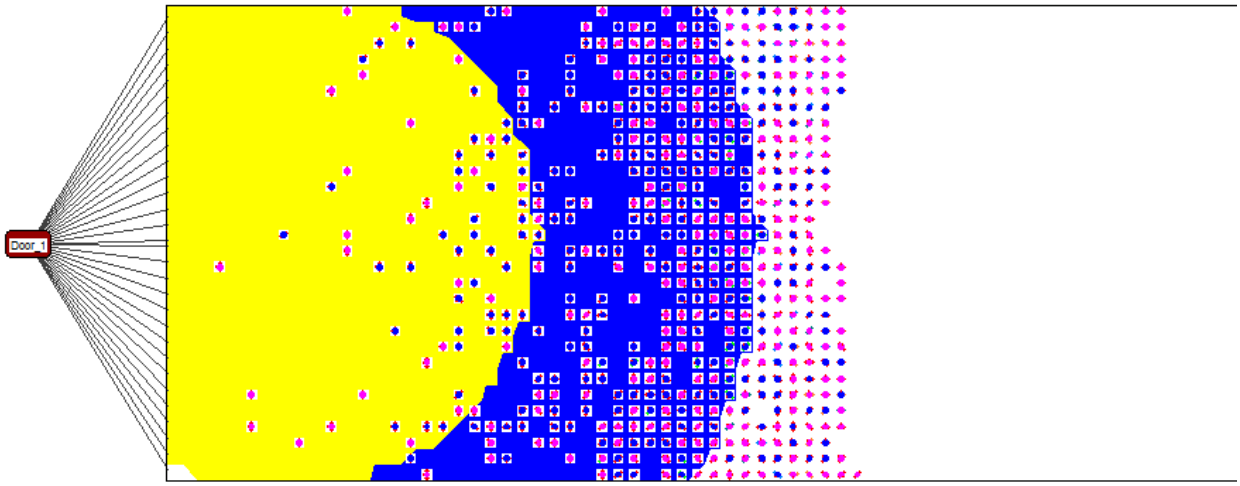


Figure 5.15: EXODUS view: User issued a Go command to all agents in the catchment area.

The zoomed in view in Figure 5.15 shows that some of the agents have immediately started moving, effectively removing the human wall and therefore allowing the blocked agents to navigate through.

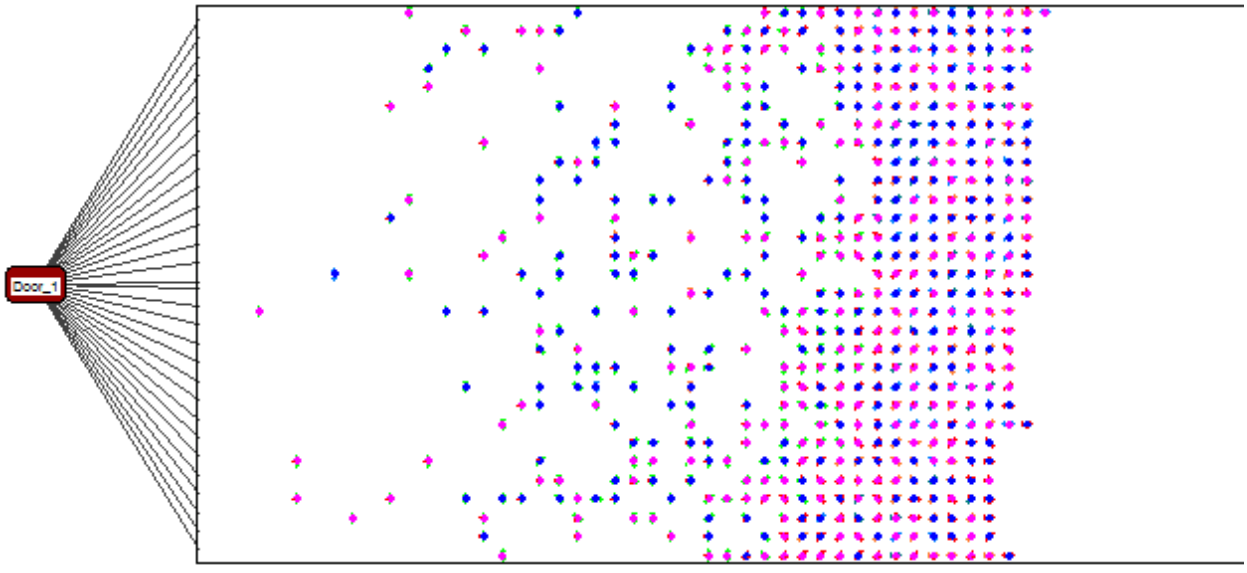


Figure 5.16: EXODUS view: User issued a Go command to all agents in the catchment area.

Figure 5.16 shows the same view as Figure 5.15, but without the overlaid catchment areas and the zoom.

Individual agent command

The user may also issue the command to a selected agent. The selected agent knows that it is the only one the command applies to. However, the same probability calculations apply to the individual agent as it applies to a crowd. The agent will assess its vicinity for other agents obeying any command and determine whether it should follow the command given by the user (See Section 5.1.2).

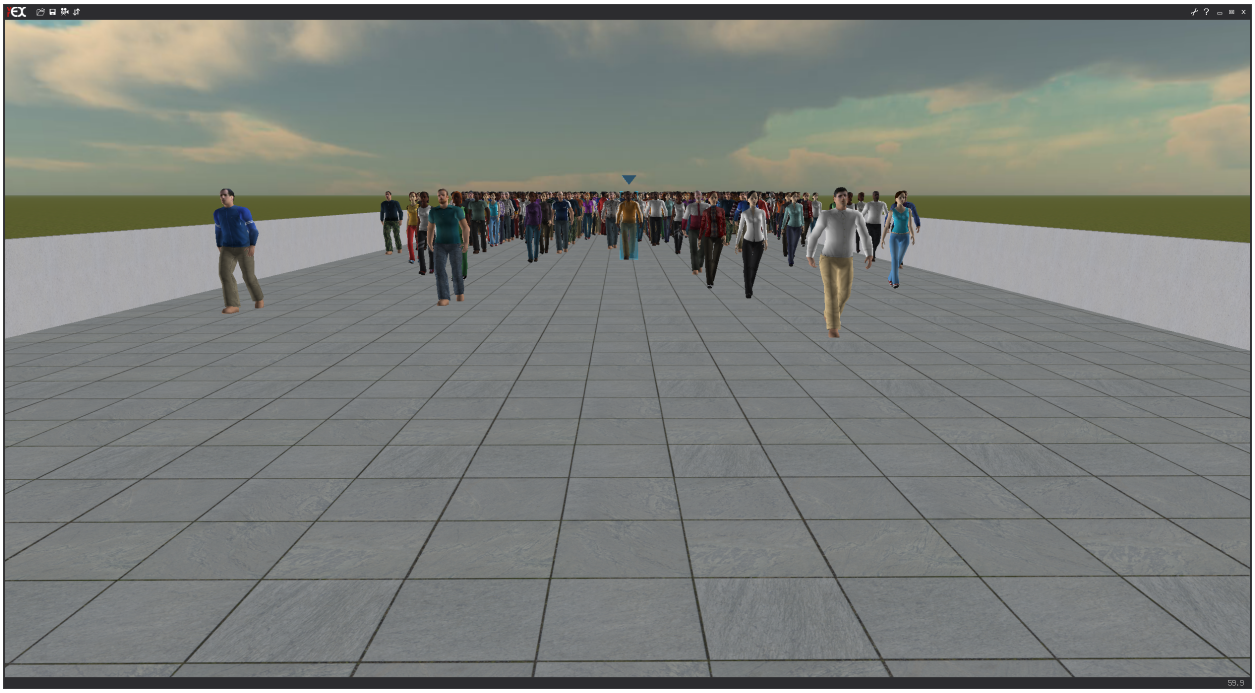


Figure 5.17: 3D view: User is selecting an agent before issuing a command.

Figure 5.17 shows that the user has selected the agent a head of time, ready to be given commands once it gets close enough to the user.



Figure 5.18: 3D view: User issued a Stop command to an individual agent.

Figure 5.18 shows that the user selected agent has reached the command influence area of the user-controlled agent and obeyed the *Stop* command as the selected agent has stopped within the catchment area of the hand gesture, indicated as a blue line.



Figure 5.19: 3D view: User issued a Stop command to an individual agent.

Notice the large crowd behind the selected agent in Figure 5.18 and how the simulation has progressed further in Figure 5.19 by having no crowd behind the selected agent as all agents have gone past him, as is illustrated by the fact that no agents are behind the computer-controlled agent. The user has repeatedly issued the *Stop* command to instruct the selected agent to stay still. The *Duration* attribute in Table 5.3 allows the obeying agent to start moving once the obeyed duration is up (see Section 5.1.2).

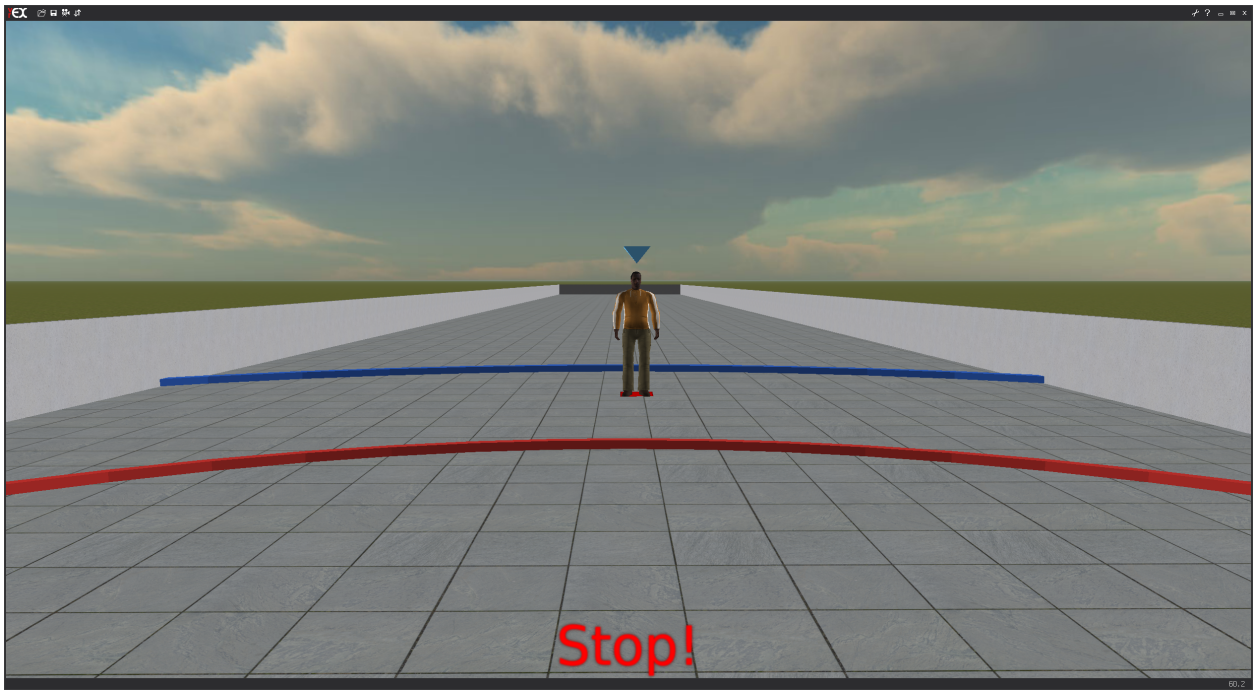


Figure 5.20: 3D view: User issued a Stop command to an individual agent.

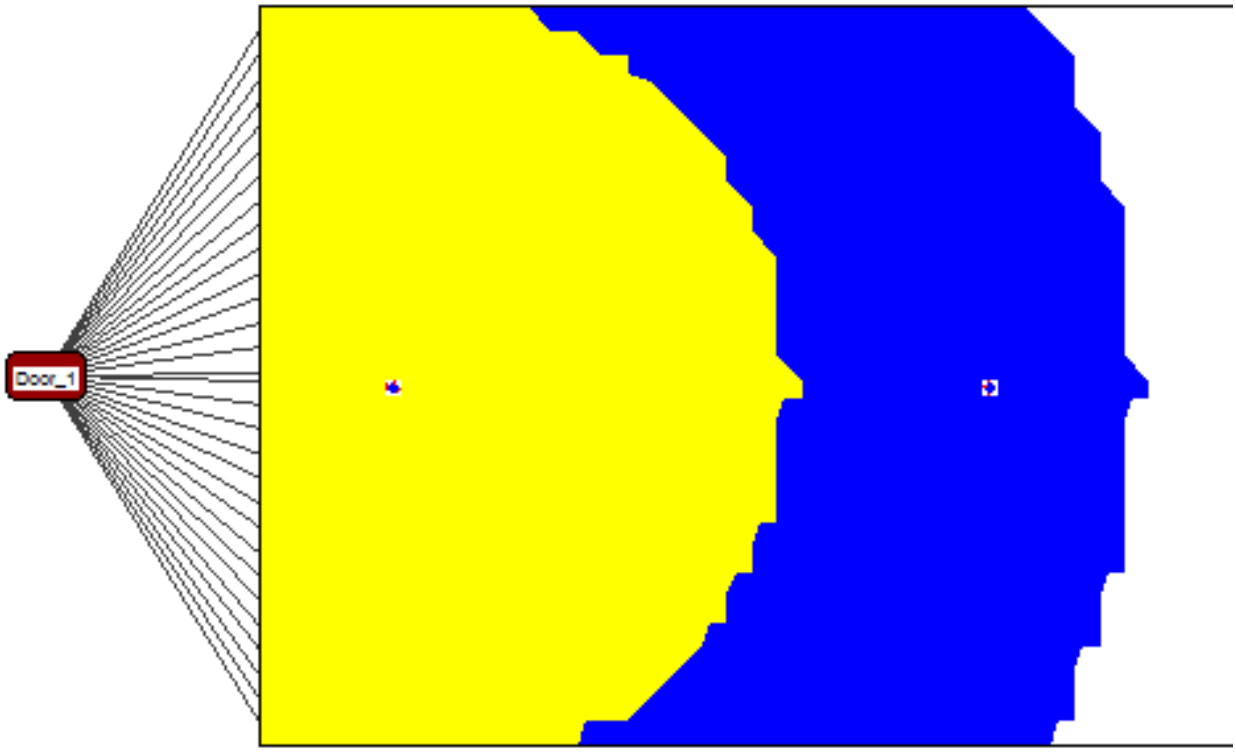


Figure 5.21: EXODUS view: User issued a Stop command to an individual agent.

Figures 5.20 and 5.21 show that the selected agent is the only one left in the simulation apart from the user and the selected agent is still successfully obeying the *Stop* command.

5.6 Get down and Get up commands verification

Some hand gesture and voice commands were developed to work in pairs (i.e. *Stop* and *Go* in Section 5.5 and *Get down* and *Get up* in this section). The *Get down* command instructs the targeted agent(s) to stop whatever they were doing and to crouch down. The *Get up* command instructs the agents to get up from the crouching position. If the agent is already standing the *Get up* command does nothing. If the agent is already crouching (either voluntarily or involuntarily due to smoke, heat or toxic gasses) the *Get down* command will instruct the agent to also stop. The behaviour of the *Get down* command was done to mimic the Metropolitan Police behaviour from the London Bridge incident in 2017 [74].

Table 5.3 defines the attributes used in this verification case. The user is standing in front of *Door_1* and will immediately start issuing the *Get down* command to all approaching

agents from the beginning of the simulation.



Figure 5.22: 3D view: User issued a Get down command to all agents in the catchment area.

Figure 5.22 shows some agents within the user influence area have already started to crouch. The progress of the influence of the command on the approaching crowd can be seen further in Figures 5.23, 5.24 and 5.25.



Figure 5.23: 3D view: User issued a Get down command to all agents in the catchment area.



Figure 5.24: 3D view: User issued a Get down command to all agents in the catchment area.



Figure 5.25: 3D view: User issued a Get down command to all agents in the catchment area.

Eventually all agents were crouching. Figure 5.25 shows the last few agents that are still standing, before also obeying the *Get down* command.

The user started to issue *Get up* command once the last agent crouched.



Figure 5.26: 3D view: User issued a Get up command to all agents in the catchment area.

Some agents immediately obeyed the *Get up* command as seen in Figure 5.26.



Figure 5.27: 3D view: User issued a Get up command to all agents in the catchment area.

Eventually a majority of the computer-controlled agents were standing as can be seen in Figure 5.27. However, as is shown in Figures 5.26 and 5.27, the agents are standing, but not moving. This is due to the agents having stopped previously, as instructed by the *Get down* command. The user can instruct the computer-controlled agents through the user-controlled agent by using the *Start evacuating* or *Evacuate in direction* commands at this point to instruct the agents to start evacuating. However, once the *Duration* period of the influence of the command has expired, the agents will start evacuating naturally.

Individual agent command

The user may also issue the command to a selected agent. The selected agent knows that it is the only one the command applies to. However, the same probability calculations apply to the individual agent as it applies to a crowd. The agent will assess its vicinity for other agents obeying any command and determine whether it should follow the command given by the user (see Section 5.1.2).

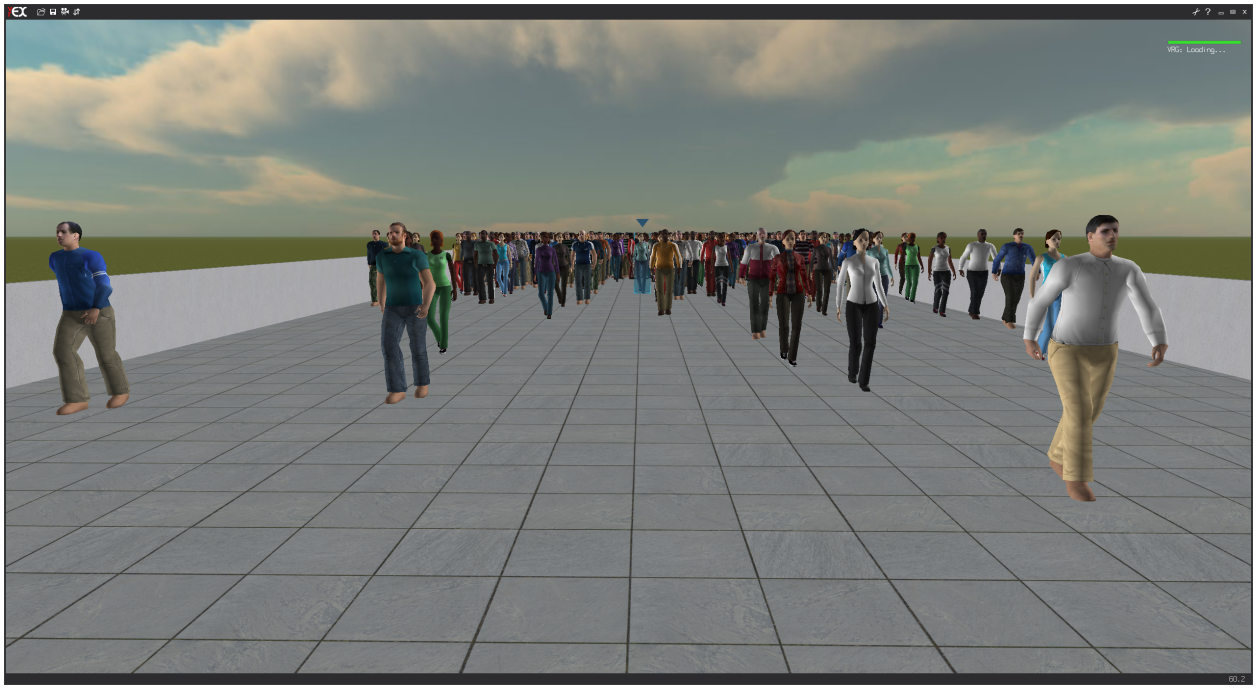


Figure 5.28: 3D view: User is selecting an agent before issuing a command.

Figure 5.28 shows that the user has selected the agent ahead of time, ready to be given commands once it gets close enough to the user.



Figure 5.29: 3D view: User issued a Get down command to an individual agent.

Figure 5.29 shows that the user selected agent has reached the command influence area

of the user and obeyed the *Get down* command.



Figure 5.30: 3D view: User issued a Get down command to an individual agent.

Notice the large crowd behind the selected agent in Figure 5.30 and how the simulation has progressed further in Figure 5.31 by having no crowd behind the selected agent as all agents have gone past her. The user has repeatedly issued the *Get down* command to instruct the selected agent to stay still. The *Duration* attribute in Table 5.3 allows the obeying agent to start moving once the obeyed duration is up.

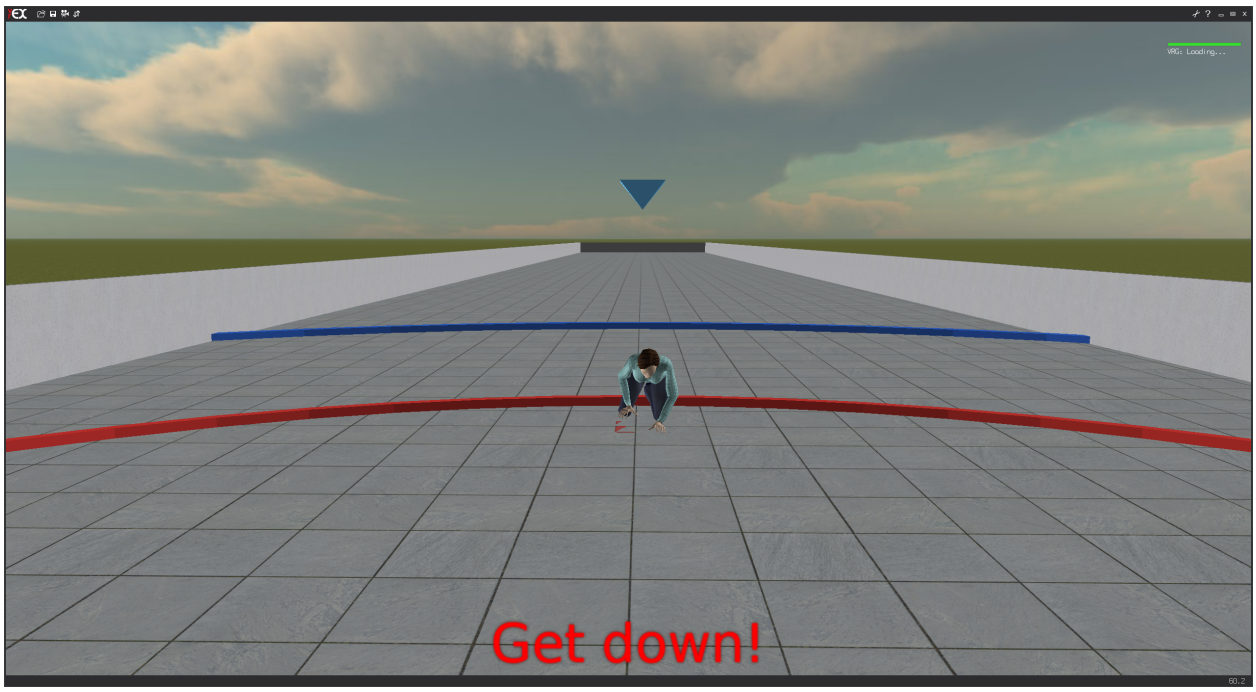


Figure 5.31: 3D view: User issued a Get down command to an individual agent.

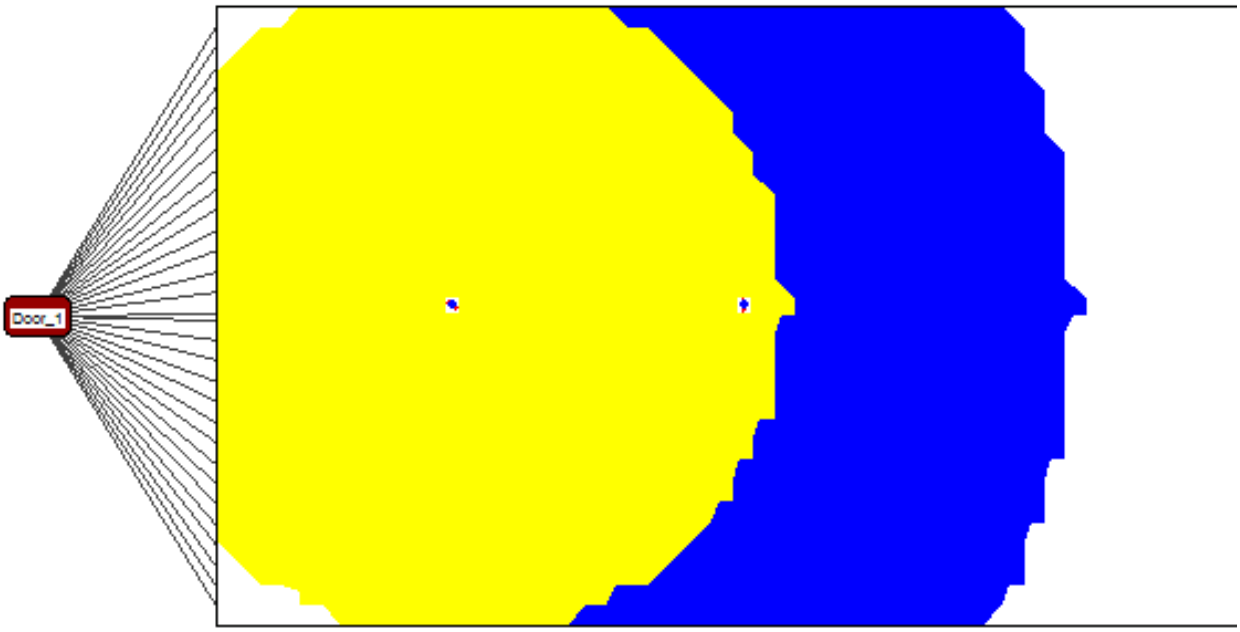


Figure 5.32: EXODUS view: User issued a Get down command to an individual agent.

Figures 5.31 and 5.32 show that the selected agent is the only one left in the simulation apart from the user and the selected agent is still successfully obeying the *Get down* command.

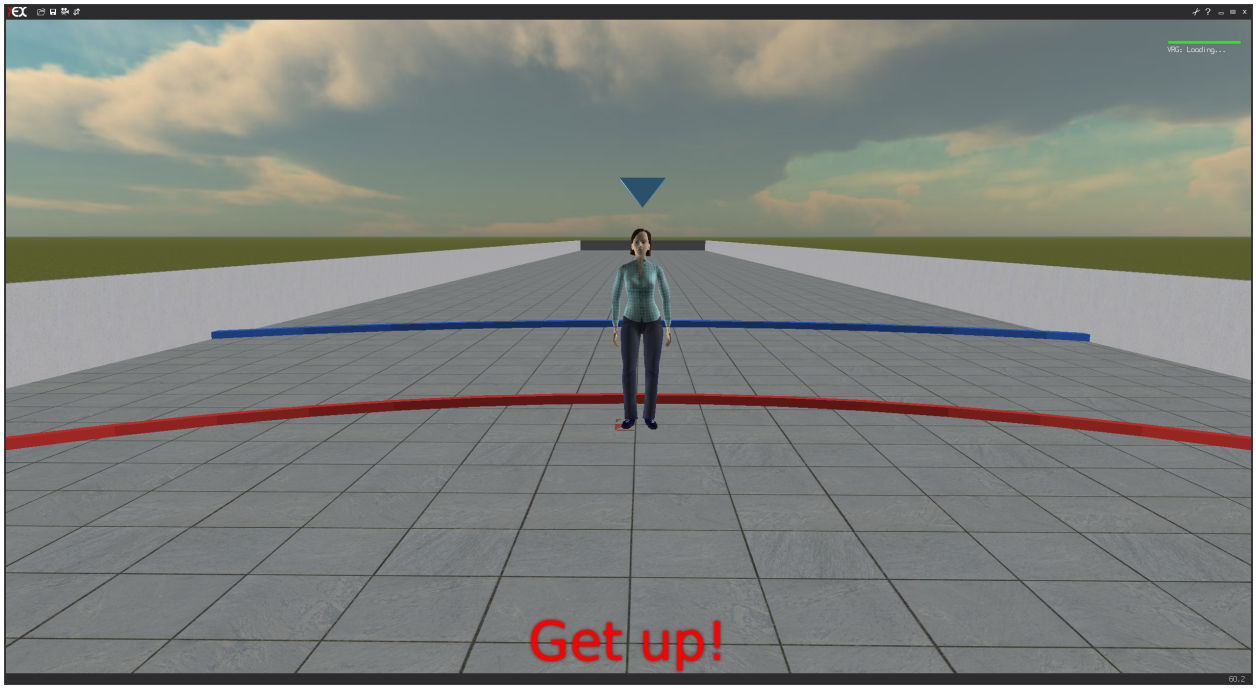


Figure 5.33: 3D view: User issued a Get up command to an individual agent.

The user eventually issued the *Get up* command, which the agent followed in Figure 5.33.

5.7 Start evacuating command verification

This verification case verifies the functionality of the *Start evacuating* command. The *Start evacuating* command instructs the agents to begin the response phase (see Section 5.3) if they have not already done so, and to abandon all other current and remaining tasks. The aim of the *Start evacuating* command is to improve the evacuation time, by having an official or other influential character hurrying the agents to respond.

Table 5.3 defines the attributes used in this verification case. The user is standing next to the agents in the centre of the geometry and will start issuing the *Start evacuating* command to all within his reach from the beginning of the simulation (see Figure 5.35).

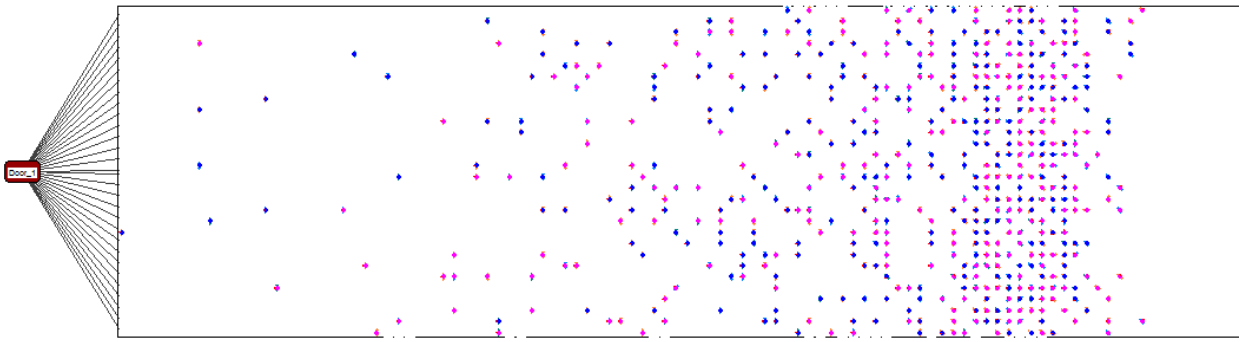


Figure 5.34: EXODUS view: Evacuation simulation at 27.8 seconds with a normal response and without user influence.

Figure 5.34 establishes the base line for the agent response without any user interaction. This is the same response time distribution as was set in the base case in Section 5.3. At 27.8 seconds into the simulation, not all agents have responded.

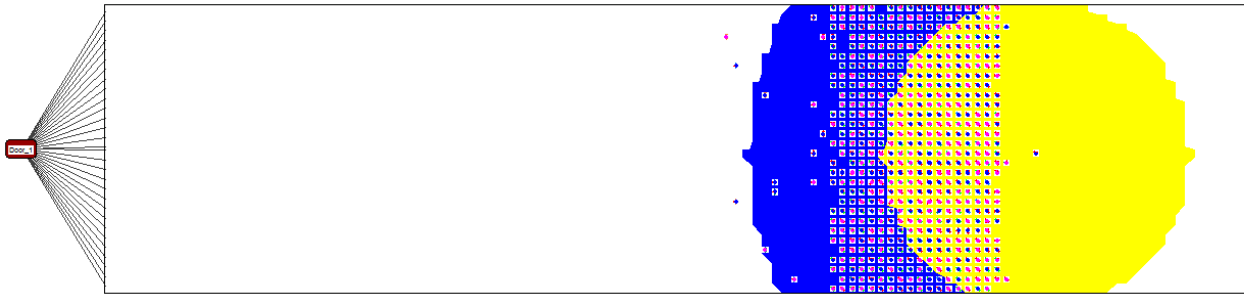


Figure 5.35: EXODUS view: User issued a Start evacuating command to all agents in the catchment area at 5.3 seconds into the simulation.

The user stands in the centre of the geometry, facing the crowd and starts issuing the *Start evacuating* command at 5.3 seconds into the simulation. Some agents have already started evacuating even before the user begins to issue the *Start evacuating* command as can be seen in Figure 5.35.

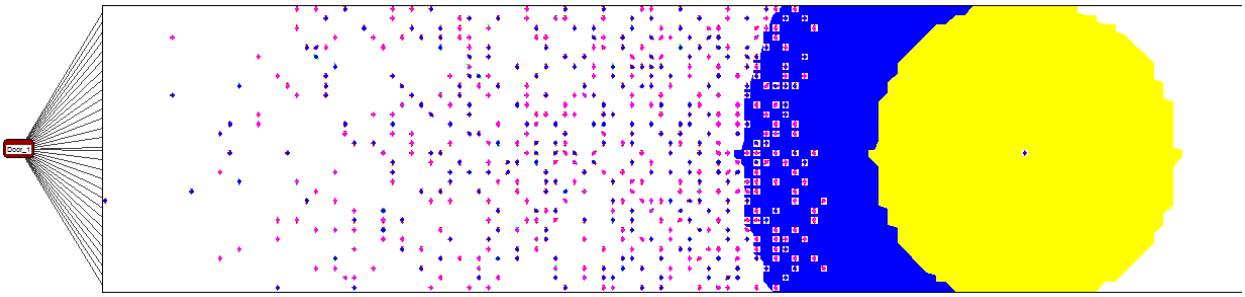


Figure 5.36: EXODUS view: Evacuation at 27.7 seconds when the user started issuing a Start evacuating command to all agents in the catchment area from 5.3 seconds into the simulation.

At 27.7 seconds into the simulation, all computer-controlled agents have responded and moved away from the starting area, as can be seen in Figure 5.36. At 27.8 seconds into the simulation in the base case the majority of the computer-controlled agents were still in the starting area as can be seen in Figure 5.34.



Figure 5.37: 3D view: User issued a Start evacuating command to all agents in the catchment area.

Figure 5.37 shows the user facing the agents at the beginning of the simulation and has started issuing the *Start evacuating* command. Currently, other agents blocking the visibility of hand gestures or hearing the vocal commands are not simulated in EXODUS. It would be

computationally expensive to accurately calculate the obscuration values of the surrounding agents from each agent's point of view. This is discussed further in Section 5.1.1. However, this should be something to research in a future project.



Figure 5.38: 3D view: User issued a Start evacuating command to all agents in the catchment area.

Figure 5.38 shows the agents responding to the *Start evacuating* command and have reached beyond the user's voice command area and the end of the user's hand gesture area.

5.8 Get out of the way command verification

This verification case verifies the functionality of the *Get out of the way* command. The *Get out of the way* command instructs the agents to move to the sides and out of the way of the user's agent. The facing direction of the user's agent is set by the facing direction network message in Section 3.3.16. The facing direction is used by the EXODUS simulation to move the agents away from the front of the user-controlled agent to allow the user-controlled agent to pass through the crowd of computer-controlled agents. The aim of *Get out of the way* command is to enable the user (e.g. as an official) to get through crowds of agents and therefore reach a point of interest sooner than merely quietly pushing through a dense crowd.

The effectiveness of this command is highly dependent on the available space that the crowd has for moving out of the away.

Table 5.3 defines the attributes used in this verification case. The user is standing in front of *Door_1* and will immediately start issuing the *Get out of the way* command to all approaching agents from the beginning of the simulation.

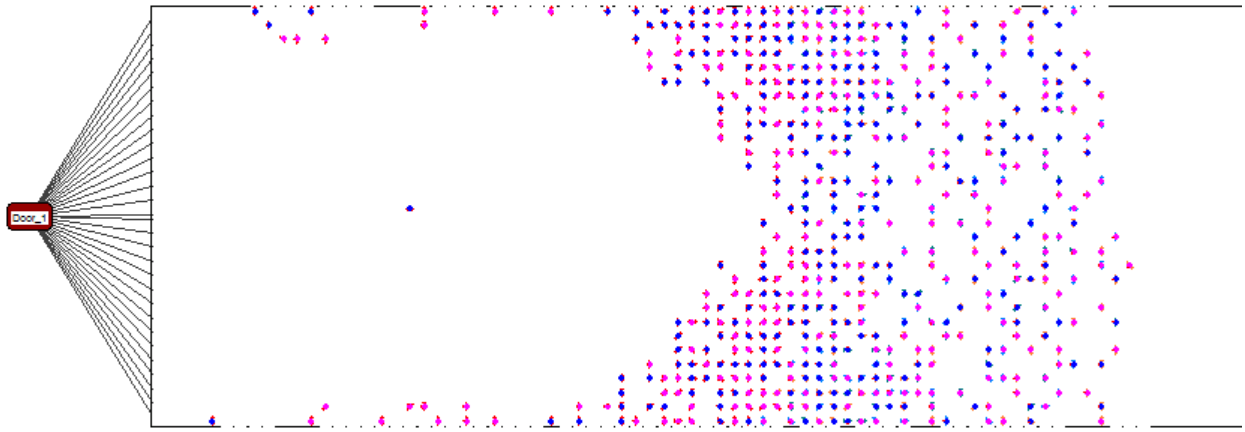


Figure 5.39: EXODUS view: User issued a Get out of the way command to all agents in the catchment area. (Without visible influence areas)

When the user issues the *Get out of the way* command it forms a pattern similar to an arrow that pierces through the crowd, making room for the agent issuing the command as seen in Figure 5.39.

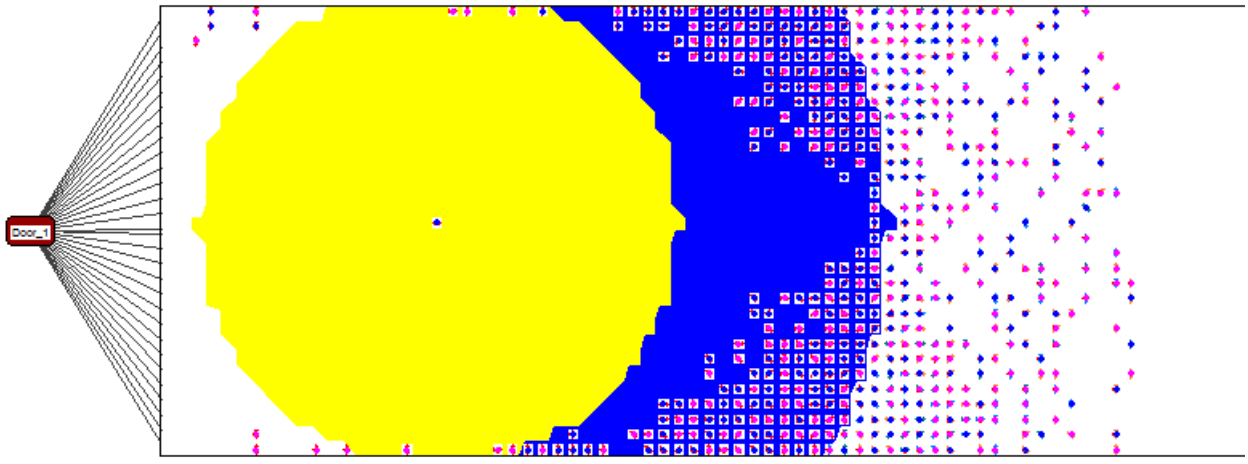


Figure 5.40: EXODUS view: User issued a Get out of the way command to all agents in the catchment area. (With visible influence areas)

Figure 5.40 shows that the beginning of the arrow pattern is formed at the edge of the command influence area. An obeying agent, who has started to evacuate, has two conflicting motivations: the agent wants to move closer to the exit and away from the centre line of the influence area. This directs agents moving towards the edges and around the user.



Figure 5.41: 3D view: User issued a Get out of the way command to all agents in the catchment area.

Figure 5.41 shows the facing directions of the crowd clearly. Figure 5.41 also shows some agents crouching. Any agent that does not have free space where to move to, will crouch and wait for the situation to be over. This behaviour happens near walls when an agent does not have a way of escaping the situation.

5.9 Move over there command verification

This test will verify that simulated computer-controlled agents will follow instructions to move to an indicated location. These instructions are referred to as *Move over there* in this thesis. The location of the *Move over there* command is set by the user pointing to a location on the ground with his mouse controller. When issuing the hand gesture, the user-controlled agent is assumed to be pointing towards the indicated location where the

computer-controlled agents should move to. The voice *Move over there* command is assumed to be giving precise instructions where the computer-controlled agents should move to and the computer simulation will simulate this by giving each computer-controlled agent a location near the point where the user-controlled agent was indicating.

Table 5.3 defines the attributes used in this verification case. The user is standing in front of *Door_1* and will immediately start issuing the *Move over there* command to all approaching agents from the beginning of the simulation.

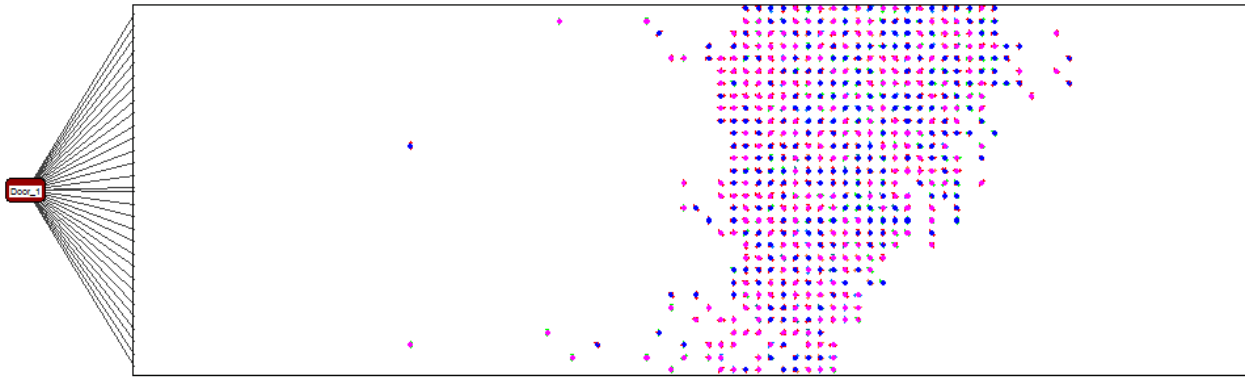


Figure 5.42: EXODUS view: User issued a move over there command to all agents in the catchment area.

Figure 5.42 shows that computer-controlled agents have obeyed the hand gestures and voice commands given by the user-controlled agent and have gathered to a single location. The gathering of the crowd can also be seen from the user-controlled agent's point of view in Figure 5.43 below.



Figure 5.43: 3D view: User issued a Move over there command to all agents in the catchment area.

To verify that the crowd can be instructed to change locations when necessary, the user-controlled agent issues a new move over there hand gesture and voice command to the computer-controlled agents as they start evacuating from the previously instructed location and move towards the user-controlled agent.

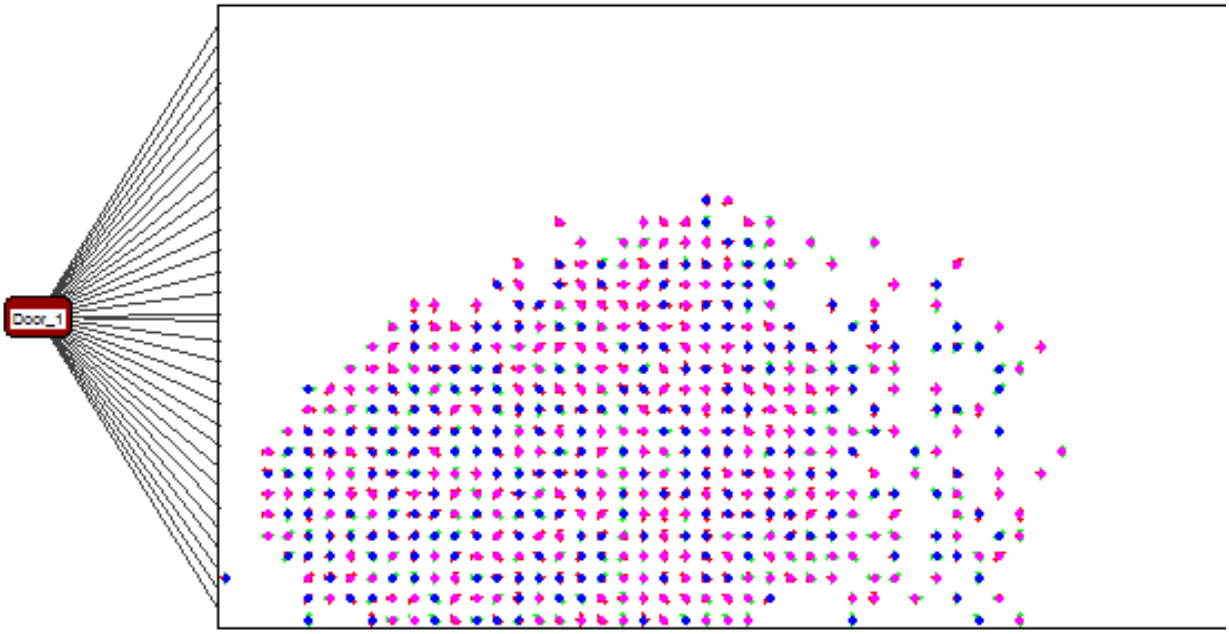


Figure 5.44: EXODUS view: User issued a move over there command to all agents in the catchment area to move to a new location.

The crowd moves to the new location instructed by the user-controlled agent as can be seen in Figure 5.44. The gathering of the crowd in the new location can also be seen from the user-controlled agent's point of view in Figure 5.45 below. As the new location is near the user-controlled agent, the user-controlled agent becomes surrounded by the crowd as can be seen in Figures 5.44 and 5.45.



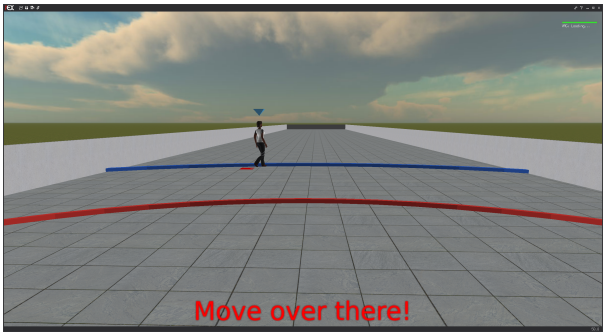
Figure 5.45: 3D view: User issued a Move over there command to all agents in the catchment area to move to a new location.

Individual agent command

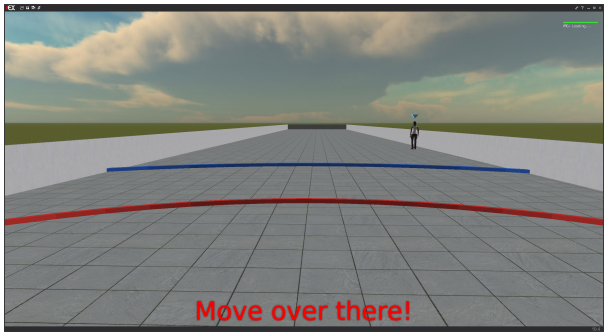
The user-controlled agent may also issue the commands to a selected computer-controlled agent. The selected computer-controlled agent knows that it is the only one the command applies to. However, the same probability calculations apply to the individual agent as it applies to a crowd. The agent will assess its vicinity for other agents obeying any command and determine whether it should follow the command given by the user (see Section 5.1.2).



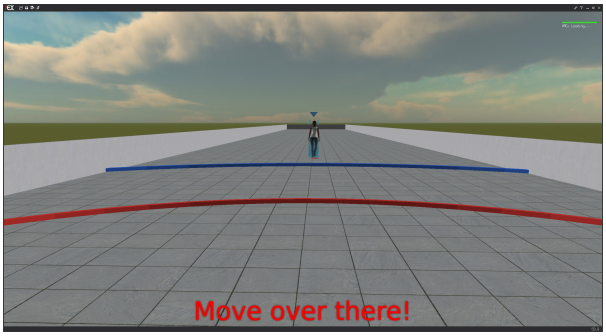
(a) 3D view: Selected computer-controlled agent moving towards the first indicated location by the Move over there command.



(b) 3D view: Selected computer-controlled agent moving towards the second indicated location by the Move over there command.



(c) 3D view: Selected computer-controlled agent has reached the second indicated location by the Move over there command.



(d) 3D view: Selected computer-controlled agent moving towards the third indicated location by the Move over there command.

Figure 5.46: 3D view: User issued a Move over there command repeatedly to an individual computer-controlled agent.

Figure 5.46 shows that the user has selected a computer-controlled agent and issued move over there commands in succession to the selected agent. The agent was given *Move over there* instructions when she reached the catchment area of the user-controlled agent. After waiting for the duration of the delay indicated in Table 5.3, the selected computer-controlled agent started to evacuate and approached the user-controlled agent as he was standing in front of the nearest exit to the computer-controlled agent. The computer-controlled agent was given instructions 3 times, which resulted in a path as shown in Figure 5.47 below.

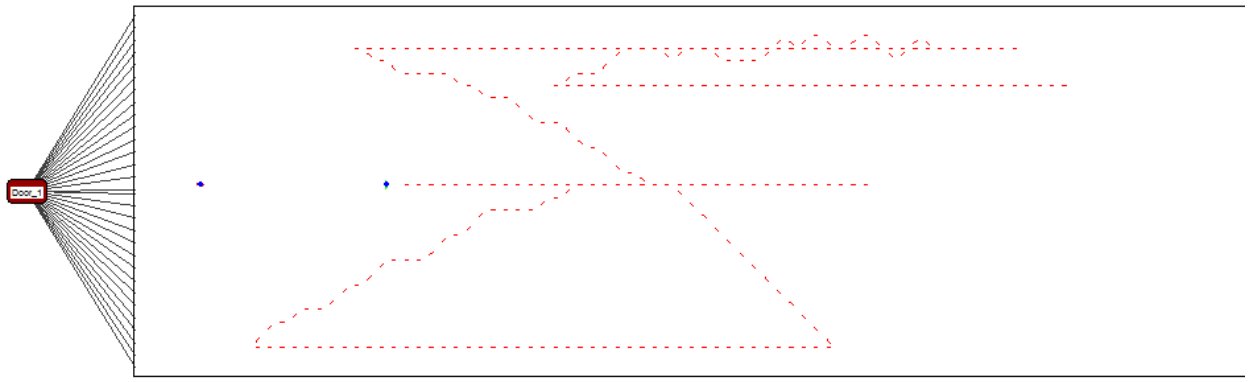


Figure 5.47: EXODUS view: User issued a Move over there command repeatedly to an individual computer-controlled agent.

5.10 Summary

This chapter detailed the research into the interaction capabilities between user-controlled agents and computer-controlled agents. Eight commands were proposed for interaction with computer-controlled agents in an immersed and simulated environment. Key attributes for the commands were identified with suitable control areas to aim to represent people interaction found in the literature. Each of the eight commands were tested and verified in functionality.

The new and novel features introduced in this chapter to the IRIS system are:

1. Interaction between users and simulated agents (i.e. both crowds and individuals) using hand gestures, voice commands or both. The interaction capabilities enable the users to work alone or in cooperation to manage simulated crowds in training or in testing of procedures, with plausible reactions and outcomes.

2. Non-deterministic simulations that are stochastic and reactive to commands issued by user-controlled agents in pedestrian behaviour models. The non-deterministic simulations allow participants to experience unpredictable, but plausible scenario outcomes reflecting real-world situations.

Each voice and hand command verification case tested and verified the individual and

crowd level response of the simulated computer-controlled agents to the instructions given by the user-controlled agent. These tests were run in an isolated environment, which reduce any experimentation variability for testing and verification. More complex scenarios are demonstrated with external users in the next chapter (Chapter 6). Such as when issuing a command that involves the agent to start or stop moving, as the agent would have a set of itineraries in a more complex scenario and the agent may have started moving as a result of following its itinerary rather than the issued command from the user.

The next chapter will provide three case studies with participants to show the capabilities that were verified in this Chapter.

Chapter 6

CASE STUDIES

In this chapter the Immersive Real-time Interactive Simulation (IRIS) system and its single and multi-user capabilities are demonstrated with example cases. Having demonstrated the key stages of component development and the construction of the complete system, and validated and verified the new key capabilities in Chapters 4 and 5, this chapter aims to demonstrate the IRIS system in action. There are many use cases where the IRIS system can be applied to (see Section 1.4), such as testing and designing procedures, enabling new training capabilities or enhancing the agent behaviour capabilities of pedestrian simulations by injecting human input.

Airports in particular are difficult places to utilise for procedure verification, testing, live trials or training. Airports are usually in constant use. Passengers can book their flights days, weeks or even months in advance. The airlines have staff in the premises serving passengers and flight crew, and servicing planes. The airplanes that are not flying are the ones not generating income. Shutting all these operations down to use the premises for testing of procedures or training would be expensive. Additionally, the use cases would be limited to non-destructive and non-harmful practises only. Such as a fire cannot be reasonably started, including filling the airport with smoke, or the usage of explosives or firing of firearms. The IRIS system can be used for training and testing of procedures in virtual airports.

To choose the right type of demonstration cases, it is important to base the cases on actual probable emergencies. The Emergency Guidebook for General Aviation, published by the University of Minnesota [190], identified a number of emergencies. In a summary of a non-comprehensive list, some of the key emergencies are: Aircraft incidents/accidents, Landside

emergencies (Medical response, Vehicle accidents, Fires), Natural disasters (Tornado, High winds, Lightning strikes), Fires (Aircraft, Structural, Vehicle and Fuel), Electrical power failures, Hazardous materials, Water/Ice rescue, Wildlife management, Security/Criminal activities (Terrorism, Bomb threats: buildings, Bomb threats: aircraft, Hijacking, sabotage or other interference with operations, Vandalism or theft, Confrontational situations and crowd control), or Overdue aircraft.

The following demonstration emergency cases have been chosen from the list as identified in the previous paragraph: landside emergencies (assisted evacuation) without smoke and in smoke, and security/criminal activities (suspicious package/bomb). An aircraft fire case has also already been demonstrated in Chapter 4. The three scenarios covered in this chapter have been performed as case studies with human participants.

This chapter aims to demonstrate the novel contributions as stated in Section 1.3, Chapter 1, of the thesis by the use of efficient and accurate real-time smoke (Chapter 4) and enabling multiple users (Chapter 3) to simultaneously interact with an existing evacuation simulation by issuing virtual voice commands and hand gestures (Chapter 5). The evacuation model provides the default behaviours to agents, including the navigation, movement and general behaviour such as tasks that the agents are performing. The evacuation model also makes the agents respond to fire hazards (such as smoke) and alarms.

The work presented in this thesis enables the user to enter the simulation as one of the agents and override some of the behaviour of the user-controlled agent, such as decisions on where the agent would like to head. The use of an existing evacuation simulation aims to substantially improve the interaction capabilities currently found in VTEs (see Section2.3, Chapter 2). How the partial overriding of the behaviour capabilities of the evacuation simulation may be affecting the validity of the evacuation model is discussed in Section 6.8.

6.1 Geometry description

There is no hard limit developed in the proposed IRIS system on the number of users or on the size of the geometry, therefore it would be possible to conduct the experiment in a very

large multi-terminal airport with tens of users, but it would be unnecessarily complex for the purpose of demonstrating the capabilities of the system. Some actual real-life use cases may require such complex set ups, but for this Chapter to be effective for its purpose on the demonstrating the proposed IRIS system, it is necessary to focus on a much smaller scale. The hypothetical scenario must still be plausible and not too minuscule in size.

A small hypothetical virtual airport was used for this purpose. It is approximately 155 metres by 55 metres in size. It is modelled as an international airport, featuring both land side and air side divided by security checks and passport control. This is not too dissimilar to that of a real rural airport. There are also much smaller real international airports, such as those found in Finland (e.g. Tampere-Pirkkala in Figure 6.1). Due to security reasons the airport covered in this chapter is not based on any real airport in particular. However, it still features most of the key airport functionalities, to give the appearance of an actual functioning airport. The layout of the airport can be seen in Figure 6.2.



Figure 6.1: Aerial view of Pirkkala airport in December 2014, from Finavia website[1]

[1]Tampere-Pirkkala Airport aerial view: `https://www.finavia.fi/sites/default/files/styles/header_image_large/public/images/2017-12/Tampere_Pirkkala_Airport_Aerial_view_2014_December_1_0.jpg`

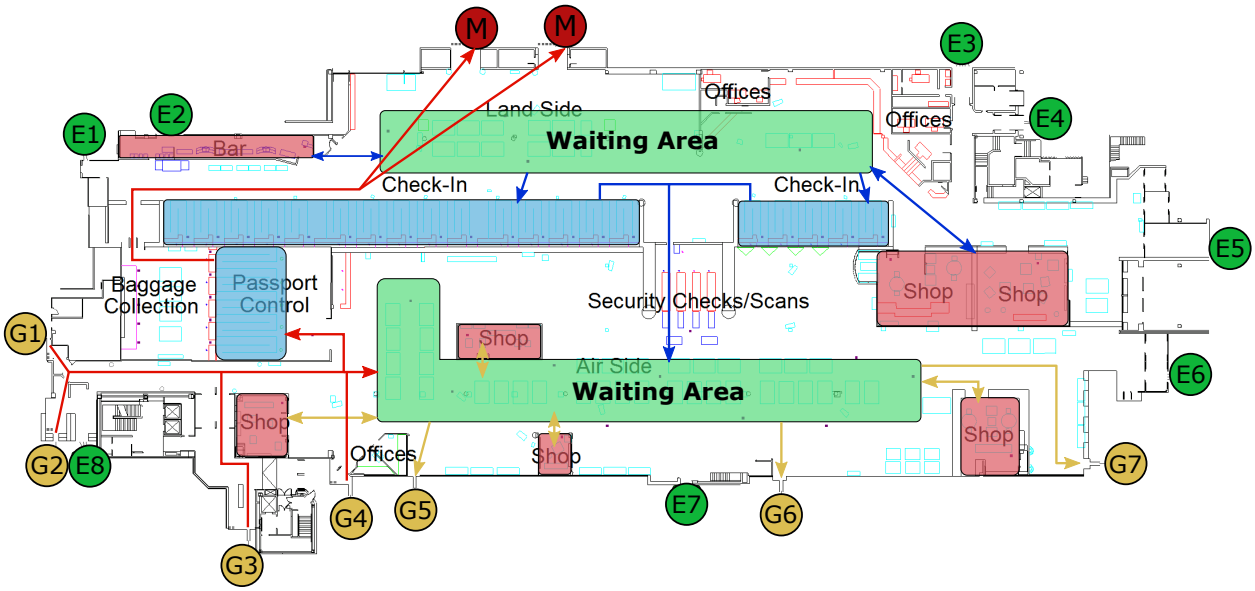


Figure 6.2: The airport geometry in EXODUS with partitioned areas

The structure is partitioned into two main areas (see Figure 6.2): land and air side. These are comprised of a main entrance (shown in red), 8 emergency exits (shown in green) and 7 departure/arrival gates (shown in yellow). Gates 1-4 are for arrivals, while gates 5-7 are for departures. Land side comprises 20 separate check-in desks, 2 shops, a bar and offices, while air side comprises 4 shops, offices and the passport control and baggage collection areas.

6.2 Population circulation

It is worth noting that all agents that take part in the circulation, are computer-controlled agents. As discussed in Chapter 2, circulation is an integral part (i.e. the part between ingress and egress) of modelling structures as people movement systems. A basic circulation scenario was performed within both scenarios (see Sections 6.5 and 6.6). Agents enter land side via the main entrance, before they either go to the landside shops/bar (red) or alternatively wait in front of the check-in desks (green). After a random amount of time, the agents randomly move to one of the check-in desks and queue (blue). When agents reach the front of the queue, they wait a random amount of time in order to simulate the time it takes to undertake the check-in procedure, before then heading off to the security checks/scans desk in order to gain entrance to air side. When agents reach air side, they will either wait or visit the shops

(shown in red). Once they have spent a random amount of time in the waiting area or in the shops, they will then move to one of the random departure gates (Gates 5-7) and queue. Once a plane arrives (i.e. the gate opens), the queued agents will board the plane, and hence exit the structure. Similarly, agents enter air side via the 4 arrival gates (Gates 1-4), before either they move to the passport control area (shown in blue) or alternatively transfer to another plane. Agents who transfer to another plane will wait for a random amount of time in the main air side area (shown in green), before they head towards one of the departure gates and queue before boarding the plane (i.e. exiting the structure). Agents move to passport control and randomly select one of the queues, and when they reach the front of the queue, the agents wait a random amount of time to simulate the time it takes to undertake the passport check procedure. Agents will then head to the baggage collection area, where they will wait for a random amount of time to simulate the time it takes to collect their bags, before they move to land side and exit via the main entrance.

Some assumptions of the circulation include: the population is assumed to be business people and each passenger is in the airport by themselves. This means there are no families (children, elderly, or disabled people) in the geometry or other complex group behaviour, such as a group of people travelling together and aiming to get to the same security queues, shops or departure gates. Also, everyone is calm and walks everywhere, which means nobody appears to be late for a plane. There is no separation of check-ins for different airlines in a sense that once an agent has randomly checked in, he will randomly select a departure gate. Each departure queue will get a random distribution of passengers and once the gate is open, the entire queue will empty, and the gate will close after a certain period of time. There are some help desks around the office areas, but these do not feature staff and passengers do not go there, as there are no problems with any security checks of check-in procedures. While the agent behaviours in the airport are missing some smaller tasks, the aim of the circulation behaviour in the two scenario cases is to simulate the main behaviours found in a typical airport. A part of distinguishing the simplifications of the circulation model proposed in this demonstration case is to highlight the complexity of a real-world scenario in comparison to a modelled behaviour simulation, and therefore shows the need for a more complex simulation design where an IRIS system can bring advanced human behaviour through real human participants.

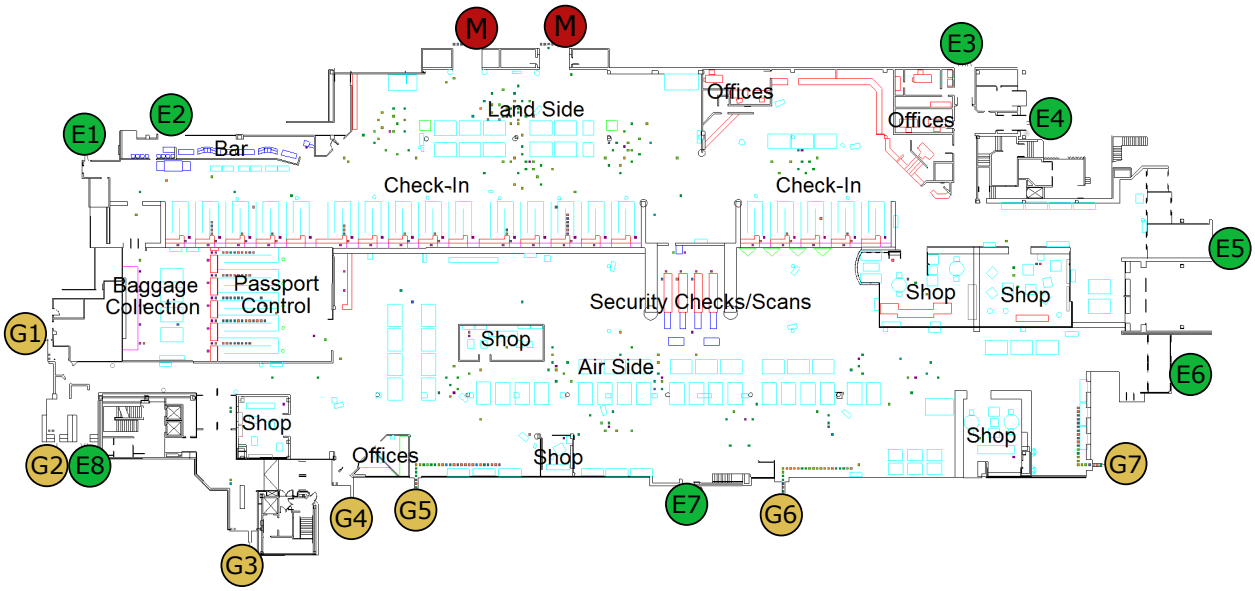


Figure 6.3: The airport geometry in EXODUS with population

In the Figure 6.3 can be seen people milling around on both land side and air side as the simulation is about 20 minutes in. The land side features many check-in areas where people can queue. Some of the areas have queues and some are empty. In the air-side people have queued in front of the exits and are waiting to be let on planes.

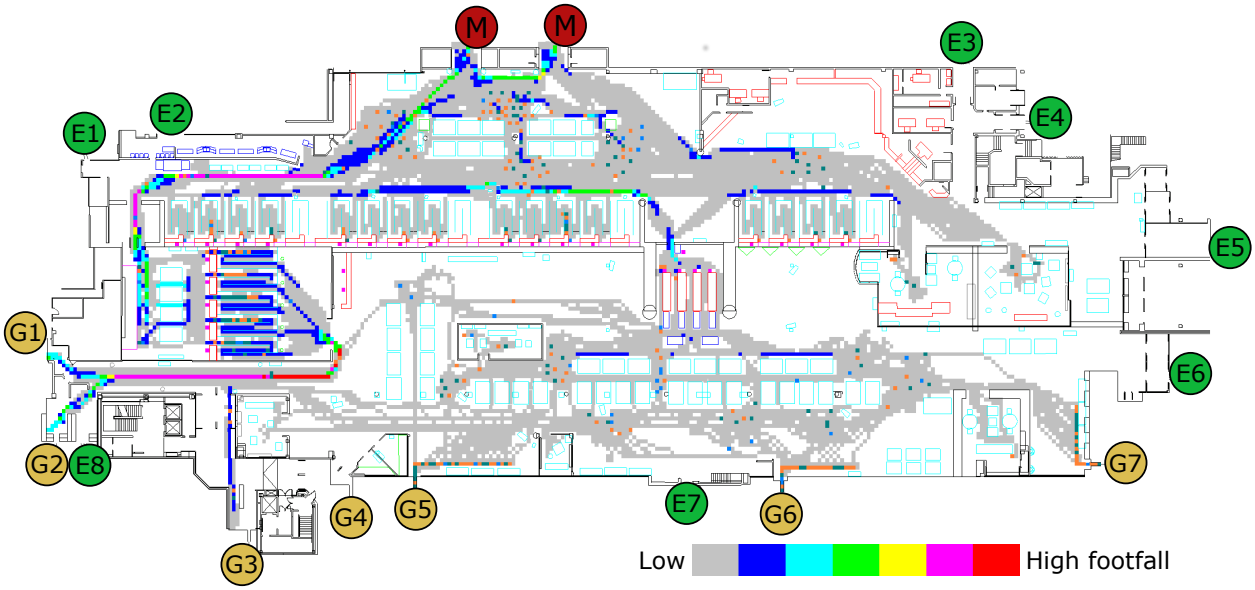


Figure 6.4: Footfall heat map of most used areas at 23 minutes into the simulation

The agents perform the ingress, circulation and egress (as identified in Chapter 2 phases in the airport geometry. During circulation the agents move between different areas in a prob-

abilistic manner (i.e. randomly choosing different routes as they move through the structure).

The geometry is initially empty, except for the 28 agents who represent staff and are positioned behind desks at the security scans, passport control and check-in areas. The 28 pre-positioned agents remain still until an alarm is sounded (see alarm behaviour in Chapter 5). When the simulation is run the ingress flow rate is set higher than the egress flow rate, which are controlled at the exits and departure gates. As a result, the total number of agents within the structure will increase as the simulation progresses. On both scenarios, the simulation was run until the total number of agents in the structure had reached 400. Once the structure reached 400 agents, the alarm was sounded, and the user-controlled agents were added to the simulation in both scenarios. The same alarm was used in the demonstration cases as was used in the verification cases in Chapter 5. Until a computer-controlled agent begins his evacuation phase (i.e. the response time has elapsed), the agent will continue to circulate within the structure. However, at the sound of the alarm, the departure gates will close, and computer-controlled agents are no longer allowed to exit via departure gates but are instead permitted to use both the main exits and emergency exits. Once the computer-controlled agents begin to evacuate, they will choose their nearest available exit. Also, the flow of new computer-controlled agents into the structure stops at the sound of the alarm (i.e. agents will not enter the structure when the alarm is ringing).

6.3 Performance measurements

6.3.1 Preliminary software performance tests

As has been described in the earlier section: the airport geometry and the population feature complex design and behaviours. The airport features many obstructions for visibility calculations, such as varying height objects, and varying size and shape rooms and corridors. The agents are also performing tasks from queuing, visiting shops and waiting areas, milling, entering and exiting the building. It was therefore an interesting and more realistic test case to measure the performance of the IRIS system. It is difficult to test the exact performance of the simulation as it varies depending on what is currently being simulated (i.e. the interac-

tions and behaviours of agents) and as the simulation is slowed down to real-time. Therefore, it is important to find the noticeable performance drop below real-time from the numerous timesteps of the simulation, where a single timestep takes longer than the maximum allowed time for the simulation to stay in real-time.

Table 6.1: Airport performance test

Number of agents	Number of simultaneous commands	Remains in real-time
200	2	Yes
200	4	Yes
200	6	Yes
200	8	Yes
400	2	Yes
400	4	Yes
400	6	Yes
400	8	Yes
600	2	Yes
600	4	Yes
600	6	Yes
600	8	No

The results for Table 6.1 were performed in a local area network (LAN) environment to ensure low latency between clients and the server. The server was run on an Intel Core i7-5930K running at 3.5GHz with 64GB of memory (i.e. to eliminate any potential memory problems). In the case where 600 agents were being simulated in the geometry and 8 commands were issued at the same time, the simulation was only occasionally falling below real-time speed. The computational performance of the command, whether it is a hand gesture or a voice command, is highly dependent on the surrounding elements for the line of sight calculations (i.e. the number of walls, objects and agents) as well as the size of the influence area of the command. Some optimisations were done to the implementation such as using a quadtree to determine which objects and agents are within the catchment area's

reach. Further optimisations may be necessary to use the IRIS system in a more complex scenarios than the airport scenario described in this chapter.

6.3.2 Base case performances

For the first two scenarios (assisted airport evacuation) in smoke and without smoke, a good base case needs to be established to compare against user performances later in Section 6.5.2. Like in the previous test cases a fixed random number seed cannot be used for both the base case and the user case, because as soon as the user interacts with the simulation (even by merely standing still) the pseudo random number generator will immediately go out of sync due to the sophisticated decision making done by EXODUS agents. Therefore, the base cases are established by running the simulation 1000 times for both clear and smoke versions of the airport evacuation (i.e. a total of 2000 runs combined). The mean curves of the 1000 runs can be seen in Figure 6.5. Table 6.2 lists summary statistics for comparing total evacuation time (TET) based on the 1000 simulation runs. The starting conditions for each of the runs are the same, which is determined by the same ingress, circulation and egress rules, the same population distribution (age, gender, mobility etc.), and the same evacuation response distribution. The simulations are run until there are 400 agents in the building at which point an alarm is sounded.

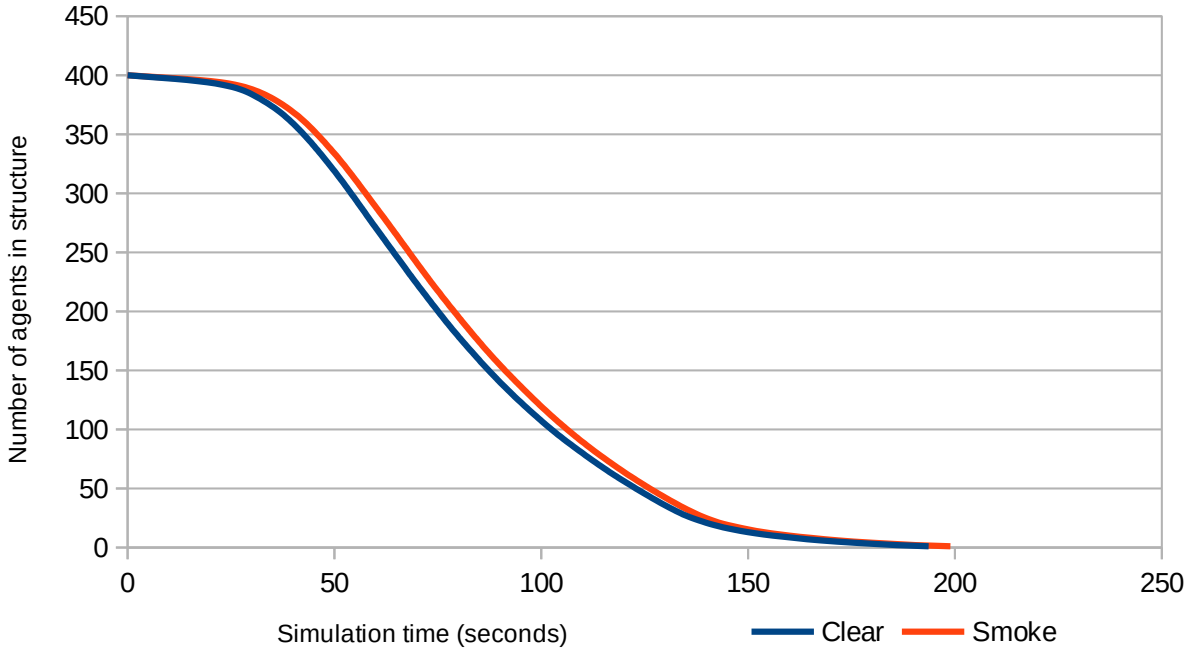


Figure 6.5: The mean evacuation time for each agent in the simulation from 1000 runs with and without smoke.

Table 6.2: Summary statistics for comparing total evacuation time (TET) between the base case with smoke (smoke column) and without smoke (clear column) based on the 1000 randomised simulations.

Total evacuation time (seconds)	**Clear**	**Smoke**
Mean	193.65	198.90
Median	194.33	199.47
Mode	194.14	191.82
Minimum	143.45	160.32
1^{st} percentile	164.94	172.60
5^{th} percentile	175.18	181.19
95^{th} percentile	209.69	218.49
99^{th} percentile	216.67	224.02
Maximum	223.25	230.14
Standard deviation	10.73	11.28
Standard error	0.339	0.357

The smoke in the airport is calibrated to be uniform to all agents (i.e. computer-controlled and user-controlled) with a visibility of 8 metres (see Chapter 4 for more information on how the visibility is calculated). The 8-metre visibility in non-irritant smoke reduces the movement speed in EXODUS simulation by about $\approx 3\%$ [191], which accounts to the cause of the reduced mean TET of $\approx 2.7\%$ as seen in Figure 6.5 and in Table 6.2. However, a recent publication [192] has argued that visibility beyond 3 metres does not reduce movement speed at all. This is something that should be further investigated [193] as the publication is partially based on artificial smoke and there is a difference in visibility in smoke between artificial and real smoke, which may be irritant, from fire [194].

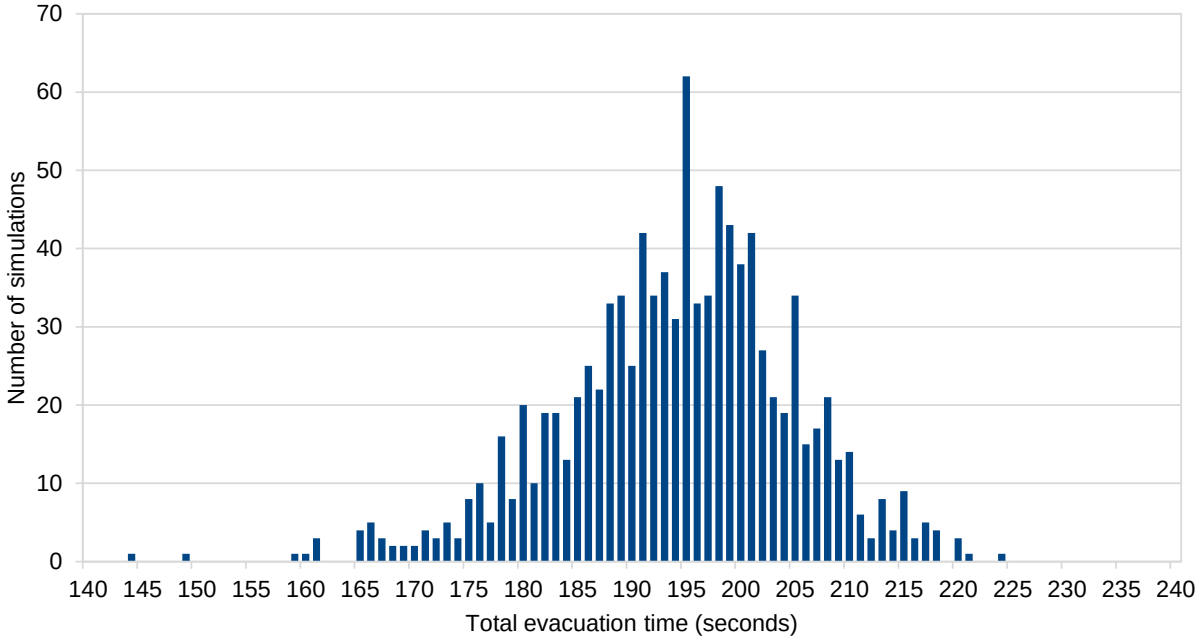


Figure 6.6: Histogram of total evacuation times (TET) of the airport geometry without smoke based on 1000 simulations.

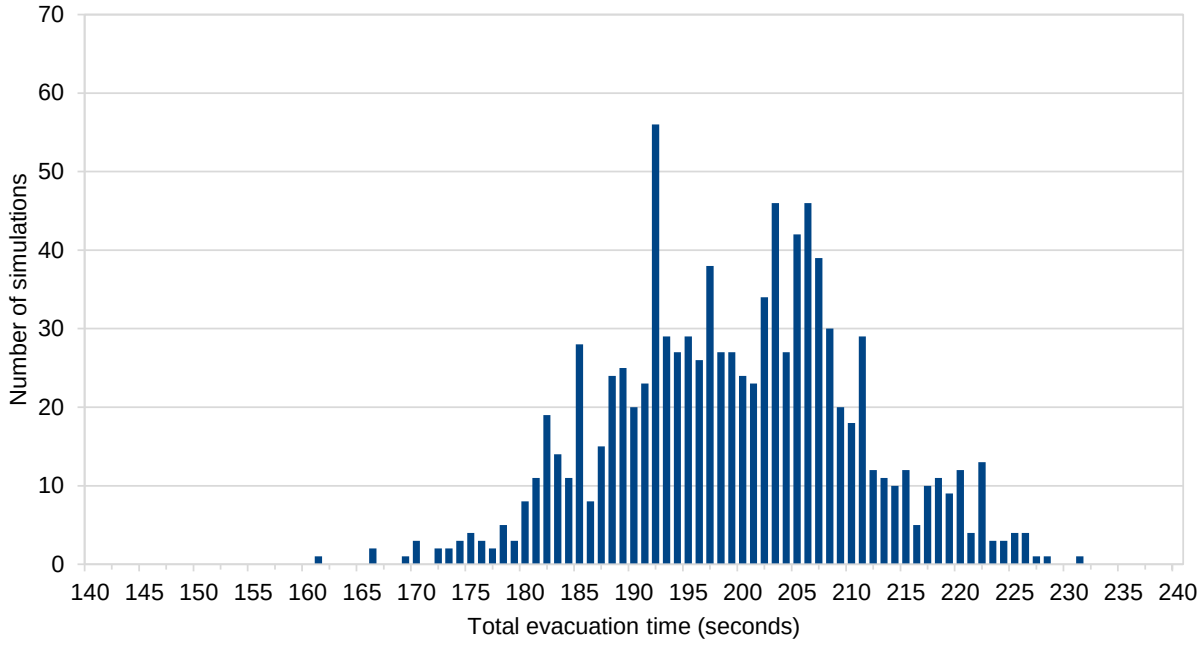


Figure 6.7: Histogram of total evacuation times (TET) of the airport geometry with smoke based on 1000 simulations.

Figures 6.6 and 6.7 show the TET distribution from the 1000 simulation runs of the base case. The randomised simulations with and without smoke produce a standard normal distribution for TET.

Each scenario also utilised existing data output generation capabilities from EXODUS to enable the outcome of each simulation to be analysed after completion. This therefore enabled the effect of the user actions on the population to be directly quantitatively and qualitatively assessed, in order to determine how successful (or not) users were in achieving their desired aims.

6.4 Participants

The case studies in this chapter and the questionnaire (in Appendix C) have been approved by University of Greenwich Research Ethics Committee (see Appendix B for the approval letter). The collected data included the questionnaire and simulation summary files. Names of the participants were only collected in the participation consent form and are not linked

to the questionnaires or the collected data. Each participant received a unique participation number, which was marked into both the questionnaire of the participant and the questionnaire of another accompanying participant (i.e. participation partner), to record which participant performed with which partner. In total 10 participants were involved with the case studies. The participants worked in pairs, which resulted in 5 pairs in total. The same participants also took the visibility in smoke tests in Section 4.7, Chapter 4.

The criteria for selecting the participants was that they had no prior experience or knowledge with the IRIS system and a participant had to be able to read text (12-point font) on a computer screen about 80cm away. All the participants are adults who are comfortable with using a keyboard and mouse.

The questionnaire revealed that both participants in Group 1 play video games regularly (weekly/monthly), including first-person video games. All participants in Groups 2 and 3 used to play video games (years ago / used to), including first-person video games. All participants in Groups 4 and 5 had either never played video games or played rarely, including first-person video games.

Only the Group 1 had been previously trained using VR prior to these case studies. Group 1 also said they were very comfortable (strongly agree) with navigating in 3D environments, including using keyboard and mouse. Groups 2 and 3 agreed that they were able to easily navigate in 3D environments, including using keyboard and mouse. One participant in Group 5 said to be comfortable in navigating in 3D environments, including using keyboard and mouse. However, Group 4 and one participant in Group 5 said they could not easily navigate in 3D environments, including using keyboard and mouse.

The participants had about 15 minutes to navigate through the airport and get familiar with it before commencing the trials. The practice period was only performed prior to the start of the trial. Once the participants felt comfortable and ready to begin or the 15 minutes were up, the practise simulation was stopped, and the participants were moved on to the first scenario.

6.5 Scenario: Assisted Airport Evacuation

6.5.1 Scenario Description

Due to airports being large open spaces, it is difficult to build a bottleneck situation apart from the typical queueing on counters and security check points. An automated rule-based computer-controlled evacuation in EXODUS simulation is likely to give similar results on each run, with a small variation based on the deviations in the random number generator. Whereas a user influenced evacuation would not be deterministic and will vary as much as the user intends to interact with the simulation. This allows the user impact on evacuation to be measured against a base case, which produces similar results over multiple runs. The performance of the base case for this exercise can be found in Section 6.3.2.

Figure 6.8 shows the route an airport official travelled, under user's control, while diverting the crowds from their typical exit route. These kinds of sudden higher-level decisions are difficult to automate in a simulation in a meaningful manner without pre-determining the requirement for it and carefully planning this interactivity using a coordinated system. The control possibility like this plays a crucial role in Scenario 2.

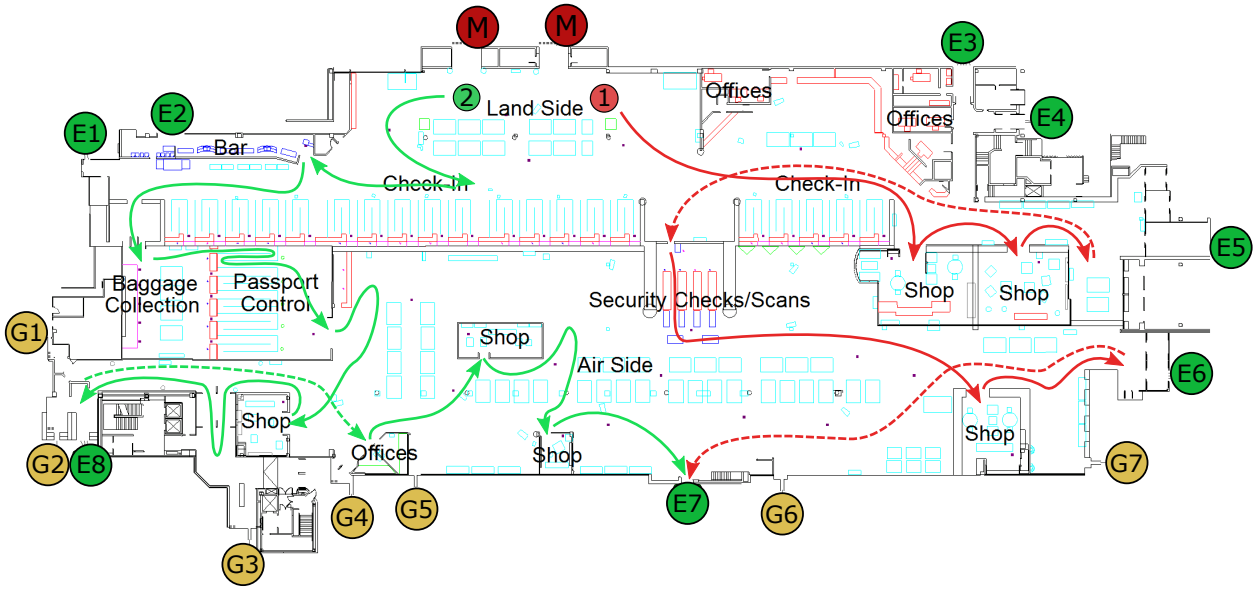


Figure 6.8: Example routes adopted by two users during the evacuation

This scenario involved using the IRIS system to proactively reduce the time taken to evacuate the airport. As outlined previously, within the *base case* the alarm is assumed to sound as soon as the current population within the structure reaches 400. At this point agents will be randomly assigned a response time, and hence will all respond within a maximum of 3 minutes, with the overall response distribution following a log normal distribution (the alarm is discussed further in Chapter 5). To reduce the overall evacuation time the *user case* was run involving the addition of two user-controlled agents.

These user-controlled agents would move through the structure issuing *Start Evacuating* commands to the other agents. The computer-controlled agents within the corresponding control areas who choose to obey the command will then immediately abandon their tasks and commence evacuation via their nearest available exit. As a result, these computer-controlled agents will not wait until their designated response time to the alarm had elapsed before commencing evacuation, and hence these agents will commence evacuation sooner than in the *base case*.

Computer-controlled agents who have not been directly issued a command by a user-controlled agent (i.e. those on airside) continue their assigned tasks (i.e. queuing, waiting, circulating etc.). The full voice command and hand gesture control areas (i.e. ignoring the effect of walls/obstacles) can be seen in Figure 6.9. In addition, the actual control areas, taking into account walls/obstacles, can also be seen in Figure 6.9.

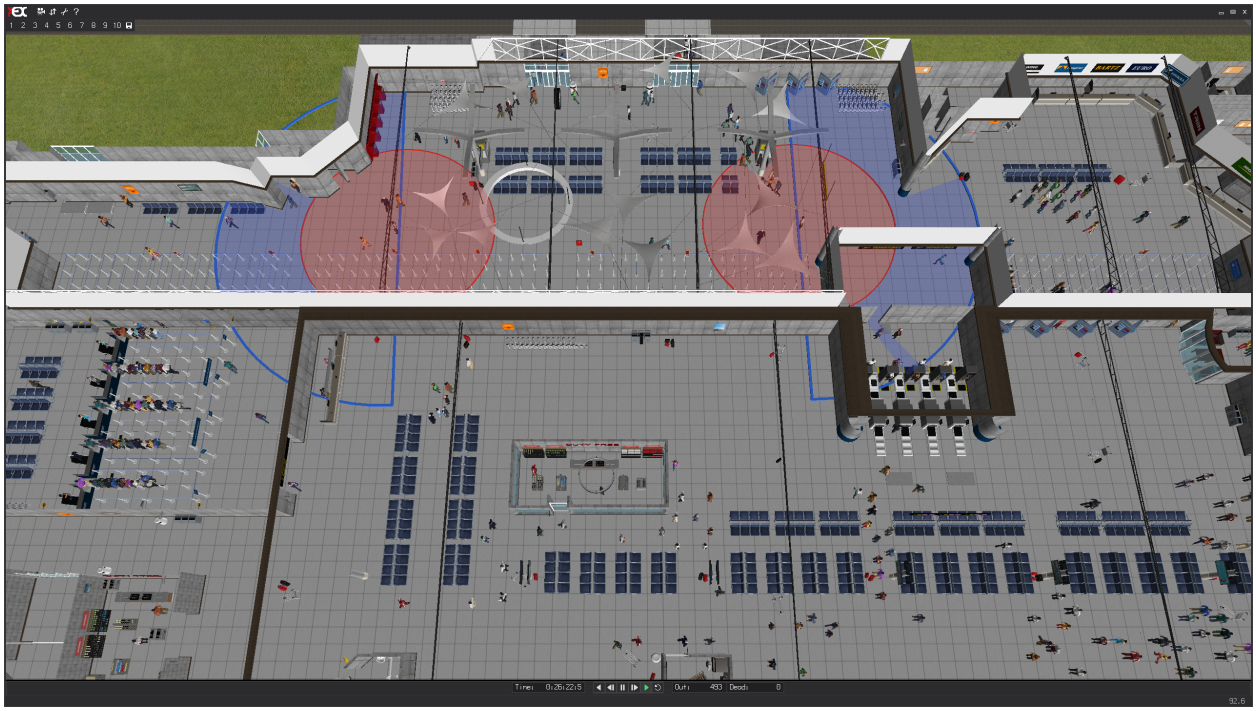


Figure 6.9: 3D view of the example users issuing Start Evacuating command. Red area indicates the voice command influence zone. Blue area indicates the hand gesture influence zone. The hand gesture and voice command areas overlap in front of the user.

Figure 6.9 shows the corresponding control areas of two user-controlled agents (as displayed in bird's-eye view in vrEXODUS) approximately 20s after the alarm has been sounded. Here it can be seen that as a result of the user agents issuing *Start Evacuating* commands, agents initially located landside around the central check-in desks and waiting area are commencing evacuation and heading towards the main entrance. Figure 6.10 shows the first-person view of a user issuing *Start Evacuating* commands.



Figure 6.10: 3D view of the example user 2 issuing Start Evacuating command. Red circle-area indicates the voice command influence zone. Blue semicircle-area indicates the hand gesture influence zone. The hand gesture and voice command areas overlap in front of the user.

The users may choose to start from any location within the structure and may exit through any of the available exits. The user-controlled agent has to be moved to an exit before the user is allowed to finish the exercise and exit the simulation. Both users do not have to exit at the same time or through the same exit. Users can talk to each other as the computers where the exercise takes place are near each other, but the users cannot see each other outside VR or each other's screens (see Figure 6.11). A tall bookcase is located between the desks of the users to prevent line of sight outside VR (see Figure 6.11). This is to simulate communication either by voice or other radio communication, such as communication devices that the staff may have in a real situation.

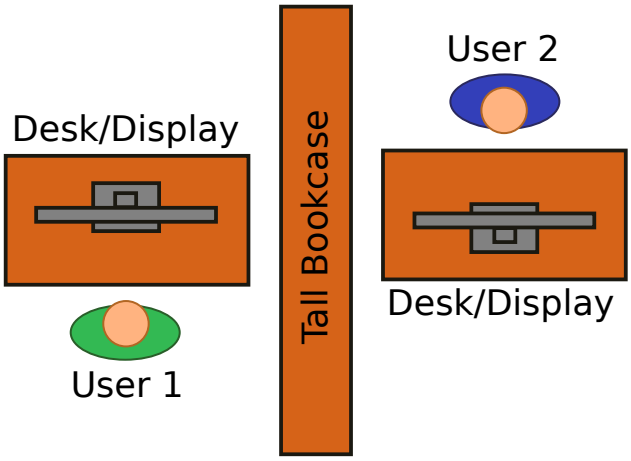


Figure 6.11: Seating positions for two participants



Figure 6.12: 3D view of a user in smoke with 8 metres of visibility (i.e. the light extinction coefficient is 0.375)

The users are not restricted to any fixed path and can go anywhere within the structure and in any order. This allows the users to plan the routes beforehand and to test different procedures over three runs to find a more optimal outcome. The three runs allow the participants to have two attempts to improve on their initial result. Once the pair of users have assisted in completing three evacuations of the airport without any hazards, a smoke hazard is introduced (see Figure 6.12) and the same users have to assist in the evacuation of the

same airport three more times in smoke. The user-controlled agents are subjected to the same effects from the smoke as the computer-controlled agents described in Section 6.3.2. The visibility in smoke is rendered for the users as outlined in Chapter 4. The smoke is set to 8 metres of visibility (see Figure 6.12) and therefore the maximum influence distance for the hand gestures and the voice commands are also limited to 8 metres. The 8 metres of visibility in smoke reduces the maximum influence distance from the hand gestures allowing the measurement of the impact of the reduced visibility. The smoke is simulated as cool non-irritant smoke to remove any effects from toxicity or heat on the agents. This is to remove any variation in the test caused by non-related factors being tested and focus the tests on the impact of the reduced visibility in smoke.

It is important to highlight that the users are unfamiliar with the fictional airport before taking part in these exercises. The only familiarisation the users have had is about 15 minutes of practise in getting used to the controls and navigating in the geometry. Therefore, the users are becoming more familiar with the geometry as they progress through the exercises. The main reason why the smoke is introduced after the clear (non-smoke) runs is to reduce the chances of them becoming disorientated by the smoke.

6.5.2 Subject performance

The performance results from the assisted airport evacuation exercises from 5 groups of users with 2 users in each group are discussed in this section. The evacuation performance graphs for each group for both with and without smoke can be found in Appendix D.

Table 6.3: Assisted evacuation without smoke: total evacuation time (TET) results in seconds

TET	**Group 1**	**Group 2**	**Group 3**	**Group 4**	**Group 5**
Run 1	121.45	167.49	162.42	194.30	194.07
Run 2	121.16	163.27	205.50	200.63	183.75
Run 3	136.25	152.71	187.31	165.38	189.18

From comparing the total evacuation time (TET) between the base case (Table 6.2)

and the user cases (Table 6.3) it is clear that Group 1 had significant impact in the total evacuation of the airport building. Group 1 reduced the TET by over 20 seconds compared to the minimum TET from 1000 runs and 72 seconds compared to the mean TET.

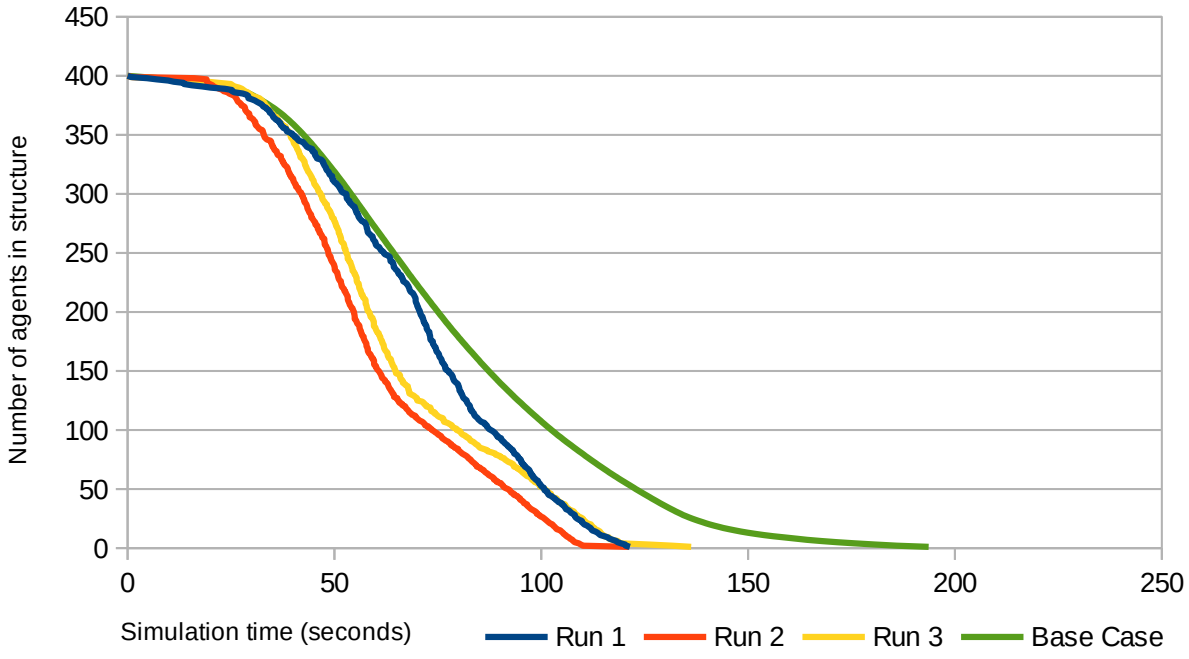


Figure 6.13: Assisted evacuation without smoke: Group 1 performance

The evacuation performance of Group 1 in comparison to the mean evacuation curve of the base case can be seen in Figure 6.13. Group 1 had similar impact on the two consecutive runs after the first run as shown by the similar shape curves in Figure 6.13.

Group 2 and 3 managed to reduce the TET (Table 6.3) below 1^{st} percentile TET of the base case (Table 6.2). Group 4 managed to reduce the TET below 5^{th} percentile of the base case. However, Group 5 was not able to reduce TET significantly below the mean TET time of the base case. In the total of 15 runs 11 were below the mean TET of the base case and 13 were below the median TET of the base case. If this is treated as a set of Bernoulli trials, the probability of 13 runs out of 15 runs (Table 6.3) to be below the median (Table 6.2) is only 0.37%. Seven (about half) of the TET results were below 5^{th} percentile and none of the TET results were above 95^{th} percentile of the base case. Therefore, the results show that in general the users had a significant impact on the evacuation performance of the computer simulated agents. Group 1 demonstrated that having familiarity and confidence in navigating

a 3D environment can significantly increase the user-impact on the simulation.

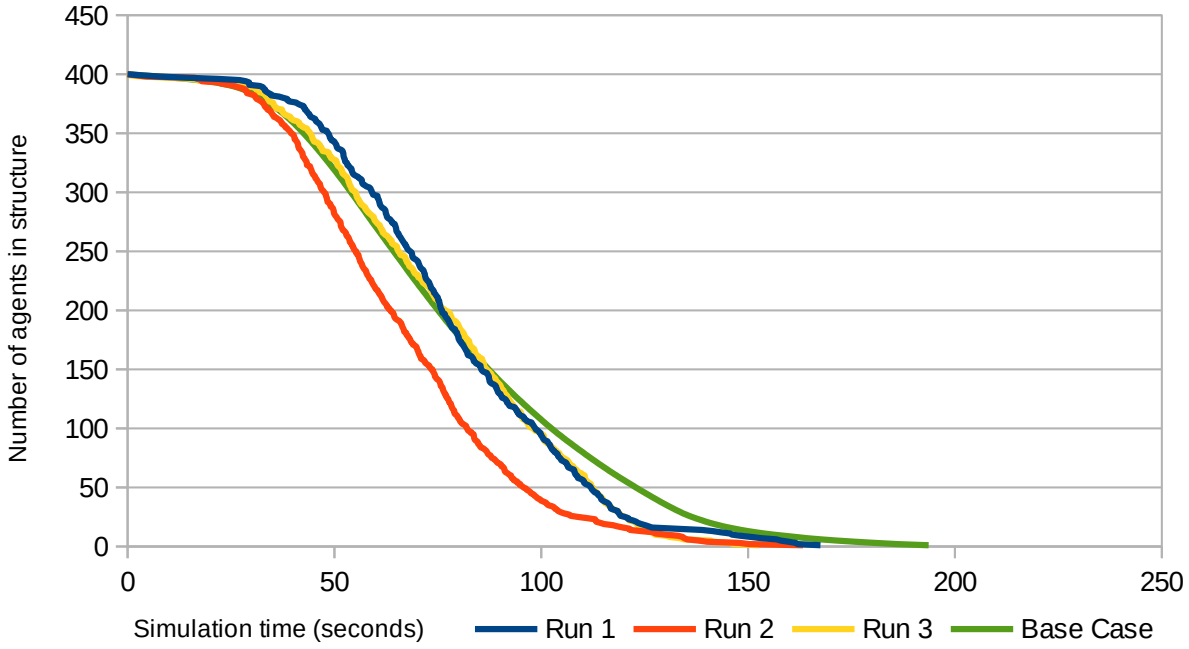


Figure 6.14: Assisted evacuation without smoke: Group 2 performance

Only Groups 1 and 2 did not alter the routes taken by the user-controlled agents from the first run and repeated the same planned procedure. Groups 3 and 5 altered the routes taken by the user-controlled agents between each run. One of the users in Group 4 altered their route for each run and the other user only altered their route for the third run.

Table 6.4: Assisted evacuation in smoke: total evacuation time (TET) results in seconds

TET	Group 1	Group 2	Group 3	Group 4	Group 5
Run 1	182.97	211.94	203.35	198.61	215.75
Run 2	188.72	174.40	195.20	196.73	177.06
Run 3	148.16	186.59	190.06	200.75	200.08

Introducing smoke into the assisted evacuation exercises made all participants perform worse in relation to the mean base case than without smoke. In the smoke free exercise Group 1 consistently managed to reduce the total evacuation time (TET, Table 6.3) in all three runs below the minimum recorded TET from the base case (Table 6.2). In contrast to the

smoke free exercises, Group 1 managed to reduce the TET (Table 6.4) only once below the minimum TET from the base case (Table 6.2). However, the Run 3 TET by Group 1 is over 10 seconds below the minimum TET of the base case which is an indication of significant impact on the evacuation performance.

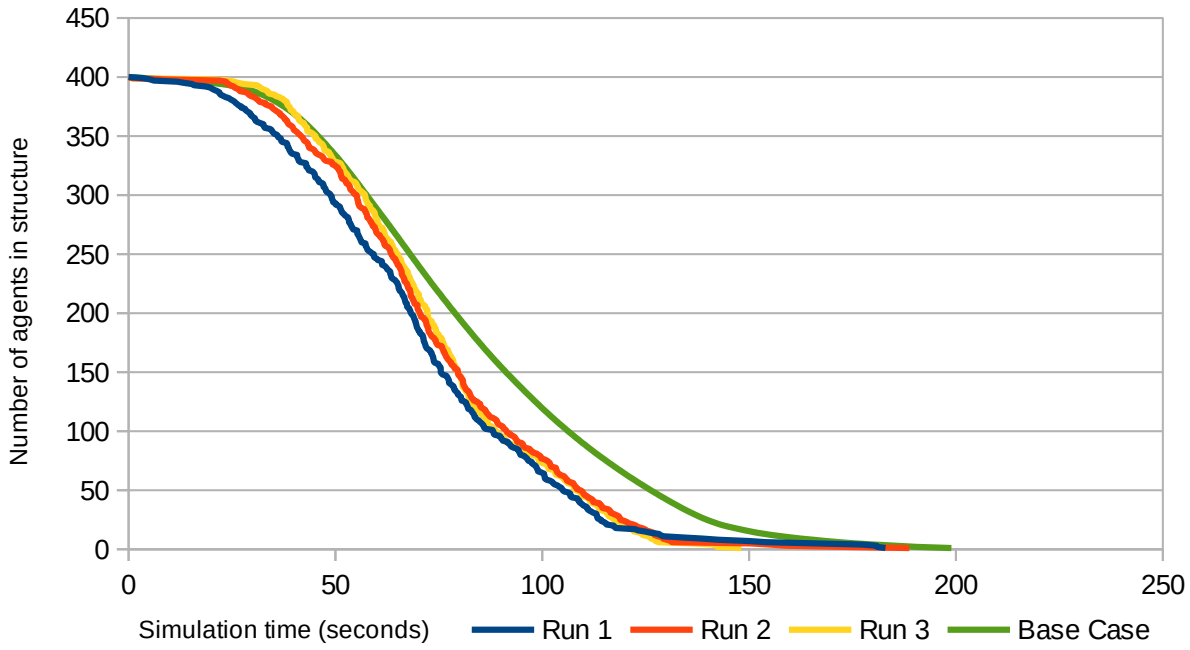


Figure 6.15: Assisted evacuation in smoke: Group 1 performance

Figure 6.15 shows the evacuation curves of the computer-controller population as assisted by Group 1 in three runs. Run 3 has about 40 seconds ($\approx$ 21%) shorter tail (TET) to the curve than Run 2 and about 35 seconds ($\approx$ 19%) shorter tail than Run 1. The recorded simulation results (available from buildingEXODUS) can provide further insight into the long TET values for both Run 1 and Run 2. Both participants exited the airport structure about 29 seconds before the last computer-controlled agent exited the structure in Run 2, and about 11 seconds before the last computer-controlled agent exited the structure in Run 1. In Run 1 there were four agents left in the structure when the participants exited and in Run 2 there were two agents left in the structure. However, in Run 3, which resulted in average $\approx$ 20% better TET value (Table 6.4), the participants exited the structure about 20 seconds after the last computer-controlled agent had exited. From the feedback of the exercises both participants struggled to locate the last remaining agents in the structure due to the reduced visibility in smoke. Also, both participants had exited the structure in all cases

once the airport was assumed to be fully vacated. The airport is approximately 155 by 55 metres in size (see Section 6.1) and the smoke reduced the visibility to 8 metres. In absence of smoke the participants could potentially see the entire length of the structure. Furthermore, Group 1 exited the structure about 10 seconds after the last computer-controlled agent had exited in all three cases without smoke, and the three cases had significantly reduced TET time (Table 6.3) in comparison to the minimum TET time of the base case (Table 6.2).

Groups 2 and Group 5 managed in one of their runs to reduce the TET (Table 6.4) below 5^{th} percentile of the base case TET (Table 6.2). None of the results were significantly worse than the mean base case. From the 15 runs a total of 12 runs were close to the mean and therefore, it is difficult to determine if they showed any significant impact on the evacuation. In comparison to the same evacuation scenario without smoke, the number of user runs that reduced the TET time below 5^{th} percentile of the base case TET dropped from 7 counts to only 3 counts by only introducing smoke into the scenario. This is a clear drop in user performance when operating in a smoke-filled environment. It demonstrates the impact of smoke in evacuation when the evacuation is assisted by the use of hand gestures and voice commands. It is important to note that the users developed increased familiarity with the building for the consecutive runs. Therefore, the users were more familiar with the geometry in the smoke runs than in the smoke free runs.

Data from the questionnaire shows that only Groups 1, 4 and 5 altered their routes between the runs in the smoke environment. Only in Group 5 the participants did not plan the routes taken by each other for the smoke environment. However, these do not seem to provide any substantial insight into the results and require further work to determine if this metric is worth recording in future exercises.

Each user reported that on each run the computer-controlled agents responded to the commands and the agents responded as expected, with an exception of a single user in Group 4 on Run 2 in the non-smoke scenario. The user reported that the agents responded, but the user did not seem to have as much influence as was desired and the computer-controlled agents did not always respond immediately. The user mentioned that sometimes more than one command was necessary to get someone to move and start evacuating. This only confirms that the behaviour manifested as intended. The user behaviour for the computer-controlled

agents were set to not always respond immediately.

Table 6.5: Assisted evacuation use case participant questions

	How realistic / realistically ...
1	... you thought the simulation was?
2	... did the agents evacuate the building?
3	... did the agents respond to your gestures?
4	... did the agents move / navigate?
	How credible was the agent behaviour...
5	... in response to your commands?
6	... in the non-smoke environment?
7	... in the smoke environment?

The users were also asked the questions in Table 6.5 after completing all the runs in this scenario. This was done to collect qualitative feedback to aid future development.

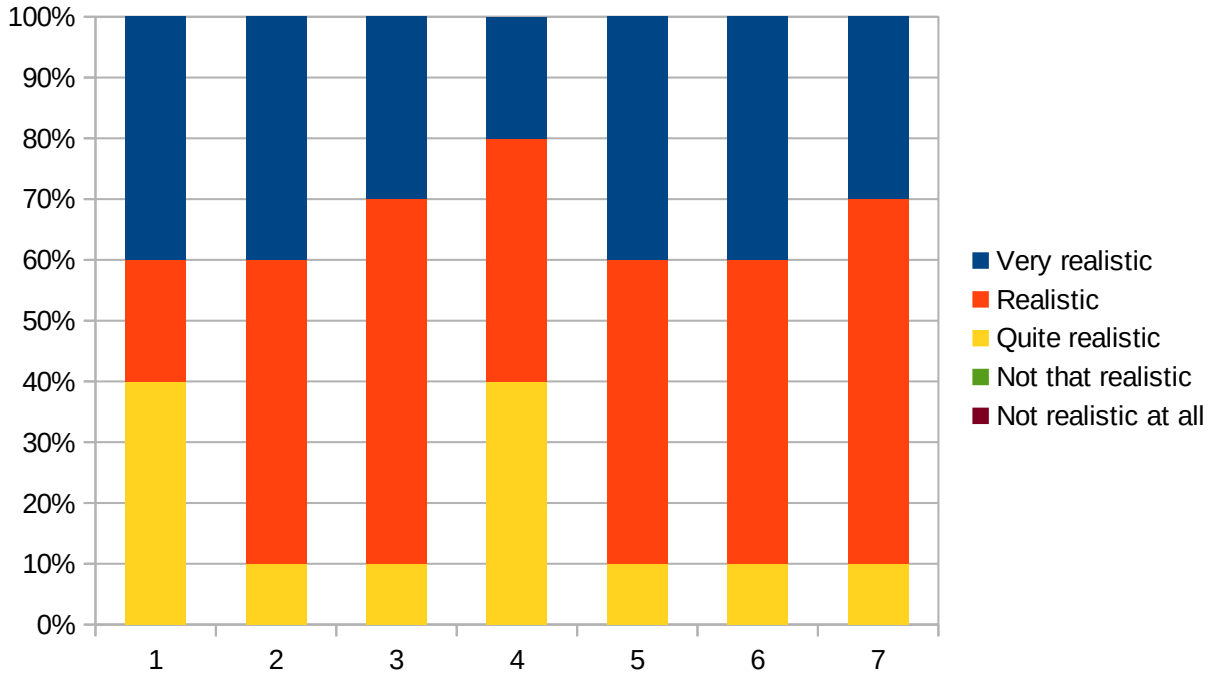


Figure 6.16: Assisted evacuation use case participant responses to questions in Table 6.5

Figure 6.16 shows participant responses to the questions in Table 6.5. All the participants thought the simulation was at least quite realistic, while 60% thought the simulation was real-

istic. The movement and the navigation of the computer-controlled agents were also thought to be quite realistic by all the participants and 60% thought the movement and navigation appeared realistic. Two of the issues reducing the score on the realism scale were related to the animation quality and the grid like (i.e. node-based) movement in buildingEXODUS software. This feedback was given verbally by some of the participants.

The evacuation of the agents and the response to the gestures and commands in both smoke and non-smoke environments were thought to be realistic by 90% of the participants. This indicates that the newly developed and tested features in this research were qualitatively thought as realistic. It is important to remember that the sample size of the participants in this study was only 10 people and therefore, these results should be interpreted only as indicative and not conclusive. However, they are consistent in their appreciation of the conditions faced.

6.6 Scenario: Secure an area

6.6.1 Scenario description

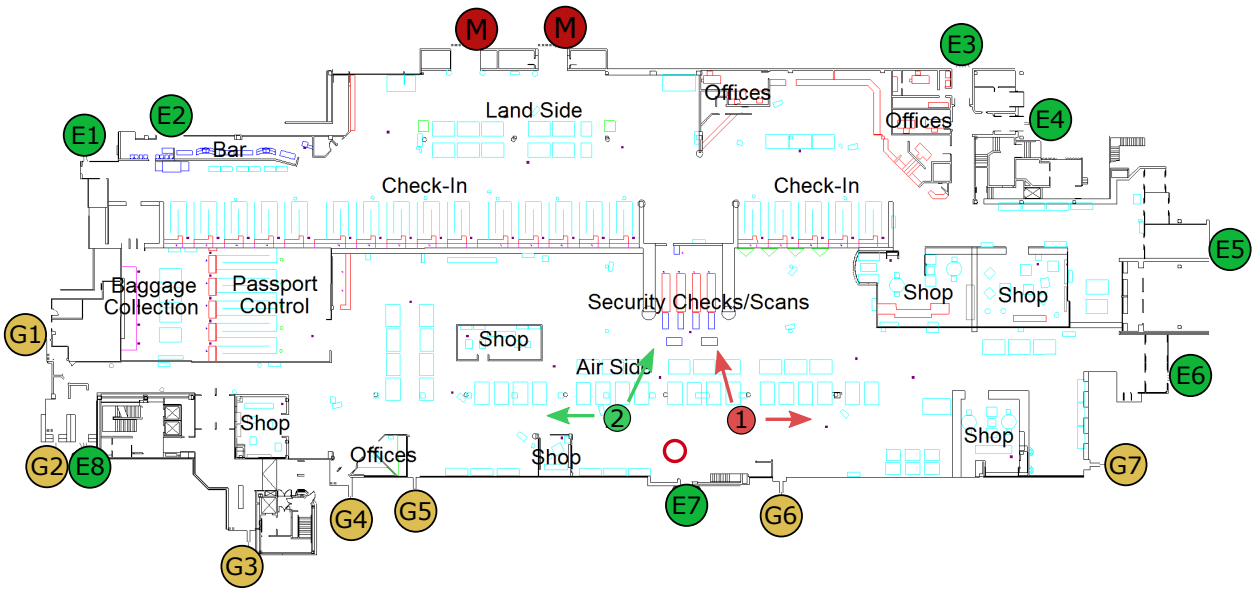


Figure 6.17: The location of a suspicious package (red circle above exit E7) and example locations of two user-controlled agents directing other agents away from it

This scenario involved using the IRIS system to proactively redirect computer-controlled agents away from an area with a suspicious package (shown in red, see Figure 6.17) towards given exits.

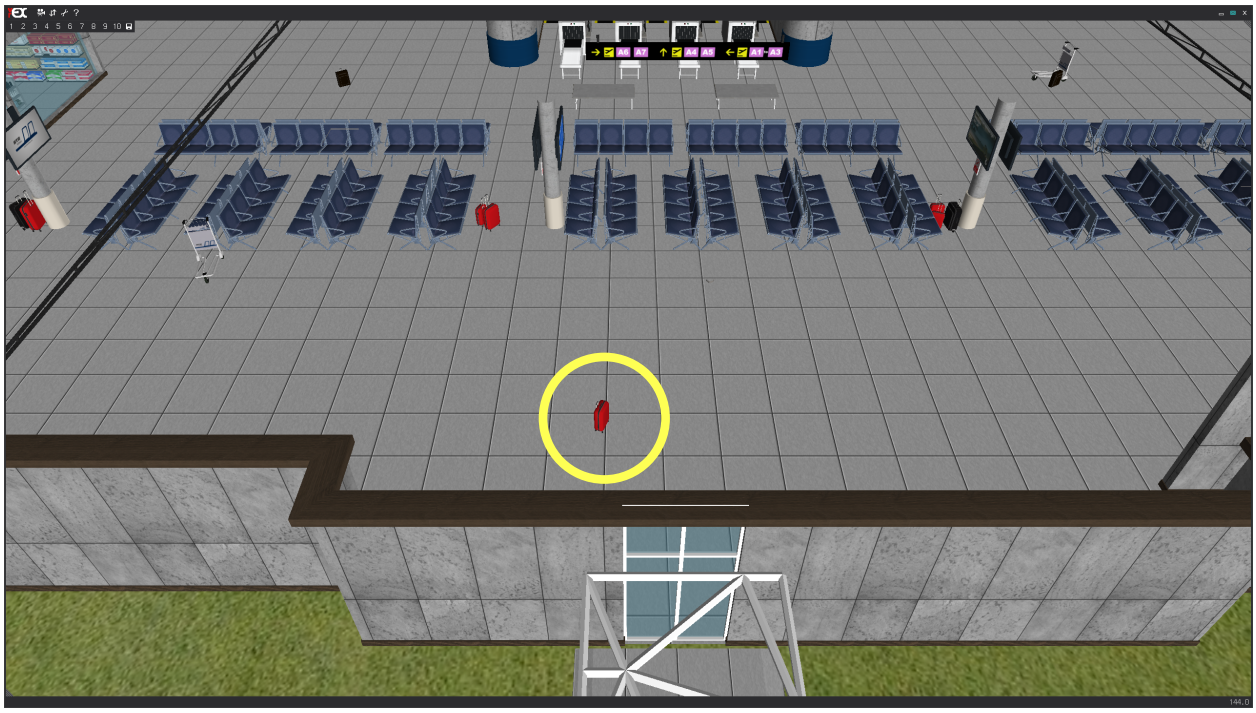


Figure 6.18: The suspicious (red) package in the 3D view highlighted by a yellow circle

The suspicious package is set to be located near the entrance to emergency exit 7. The suspicious package is a red suitcase, which looks like any of the other red suitcases around the building (see Figure 6.18). Similar to the first scenario, two user-controlled agents were added to the structure, at the time the alarm was sounded. These user-controlled agents were located in close proximity to the suspicious package, and they stayed in fixed positions in order to guard the given area (i.e. the agents will not move around within the structure). Throughout the simulation the user-controlled agents issued *Evacuate That Way* commands, in order to make nearby agents immediately commence evacuation via exits in the direction that the user-controlled agent was facing when the command was issued. In this manner, all agents approaching the suspicious package would immediately be directed away from it towards available exits.

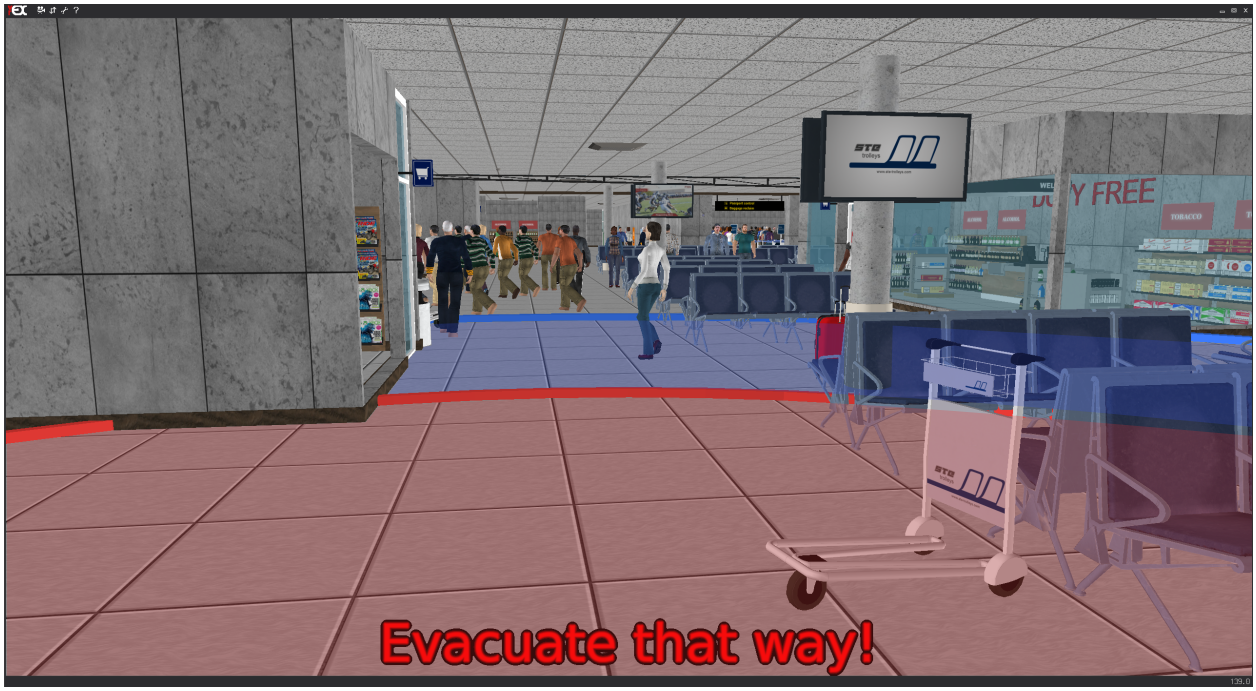


Figure 6.19: 3D view of the example user 2 issuing Evacuate That Way command. Red area indicates the voice command influence zone. Blue area indicates the hand gesture influence zone. The hand gesture and voice command areas overlap in front of the user.

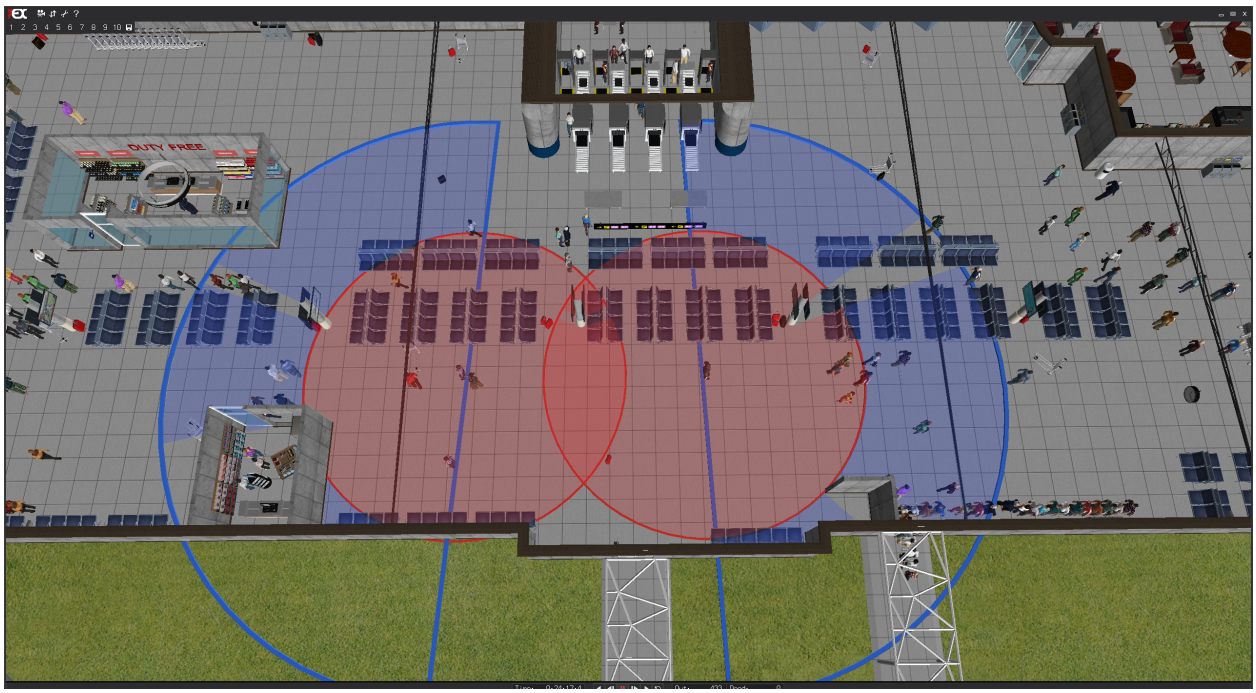
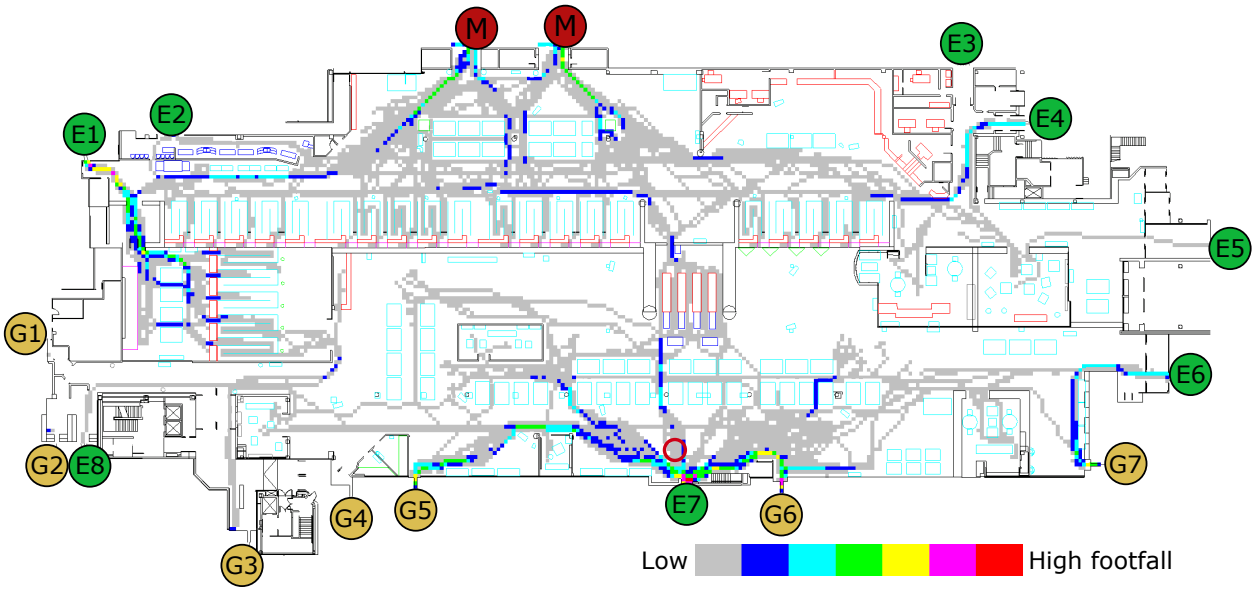


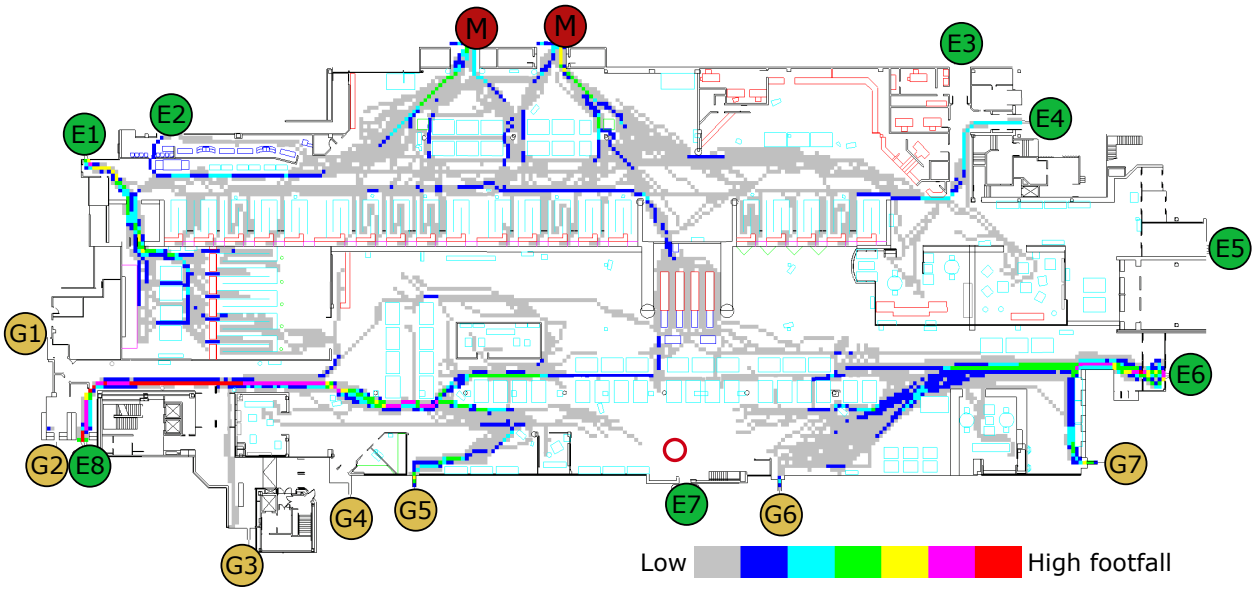
Figure 6.20: 3D view of the example users in bird's-eye view. Red circle-area indicates the voice command influence zone. Blue semicircle-area indicates the hand gesture influence zone. The hand gesture and voice command areas overlap in front of the user.

Figure 6.20 shows the corresponding control areas of the two example user-controlled

agents (as displayed in bird's-eye view in vrEXODUS) approximately 20 seconds after the alarm has been sounded. As a result of the user-controlled agents issuing *Evacuate That Way* commands, computer-controlled agents initially located around emergency exit 7 (E7) (i.e. the location of the suspicious bag) have commenced evacuation in their assigned direction. Hence, agents within the control area of user-controlled agent 1 can clearly be seen heading east towards emergency exit 6 (E6), while those in the control area of user-controlled agent 2 are in contrast heading west towards emergency exit 8 (E8). The full voice command and hand gesture control areas (i.e. ignoring the effect of walls/obstacles) can clearly be seen, as can the actual control areas.



(a) Base Case



(b) User Case

Figure 6.21: Footfall contours for the Base Case, and User Case. The location of the suspicious package is indicated by a red circle.

Figure 6.21 shows the paths adopted by computer-controlled agents after the alarm was sounded via the EXODUS footfall contours. The colour of the footfall contour depicts the level of use by agents during egress, with the colour of the contour being darker the more it was used. Figure 6.21a shows the footfall contours within the *base case*, where agents respond to the alarm and exit via their nearest exit. The heavy use of the main exit and emergency exit 7 (E7) can be seen, along with reduced levels of usage by emergency exits

1 (E1), 2 (E2), 4 (E4) and 6 (E6). In comparison, Figure 6.21b shows example footfall contours within the *user case* and demonstrates the effects on the adopted paths by agents as a result of a successful user intervention. Here it can clearly be seen that emergency exit 7 (E7), which had been significantly used in the *base case*, is no longer used, as a result of the suspicious package in front of it, and users directing agents away from it. As a result, the usage of the emergency exits 6 (E6) and 8 (E8) (i.e. the exits to which agents were redirected) have significantly increased. All other exits, particularly those located land side, show almost identical levels of use. The main exits also show almost identical levels of use, even though the computer-controlled agents that were approaching the suspicious package via the security scans would have been redirected. The lack of any significant change in the footfall contours for the main entrance, therefore, indicates that not many agents were redirected back through to land side during the simulation.

The participants of the case studies will aim to produce a footfall contour similar to the successful example case shown in Figure 6.21b. The participants may choose any location for their user-controlled agents and may discuss any plans including directions for the evacuation instructions. A total of three runs will be conducted for each group of participants. This aims to demonstrate learning and improvement over multiple runs assuming the participants are not successful in each run.

6.6.2 Subject performance

A trial to secure an area near an exit in an airport was performed using five groups with two users in each group. The aim of the exercise is for the two participants to prevent any computer-controlled agents approaching the suspicious bag. The participants used hand gestures and voice commands to guide occupants to different exits and away from the bag. The performance results from the securing of an area exercise from five groups of users with two users in each group are discussed in this section. A comprehensive list of all the footfall contours from all the participant runs are located in Appendix D.2.

With the exception of Group 1, all other groups failed to secure the area in at least one of the runs. This may be due to the Group 1 being more familiar with first-person games and

playing computer games more regularly than other groups and therefore being more familiar with the mouse and keyboard controls (see Section 6.4). The other groups (i.e. not Group 1) were able to show clear improvements in becoming more successful in their second and third run in comparison to their first run.

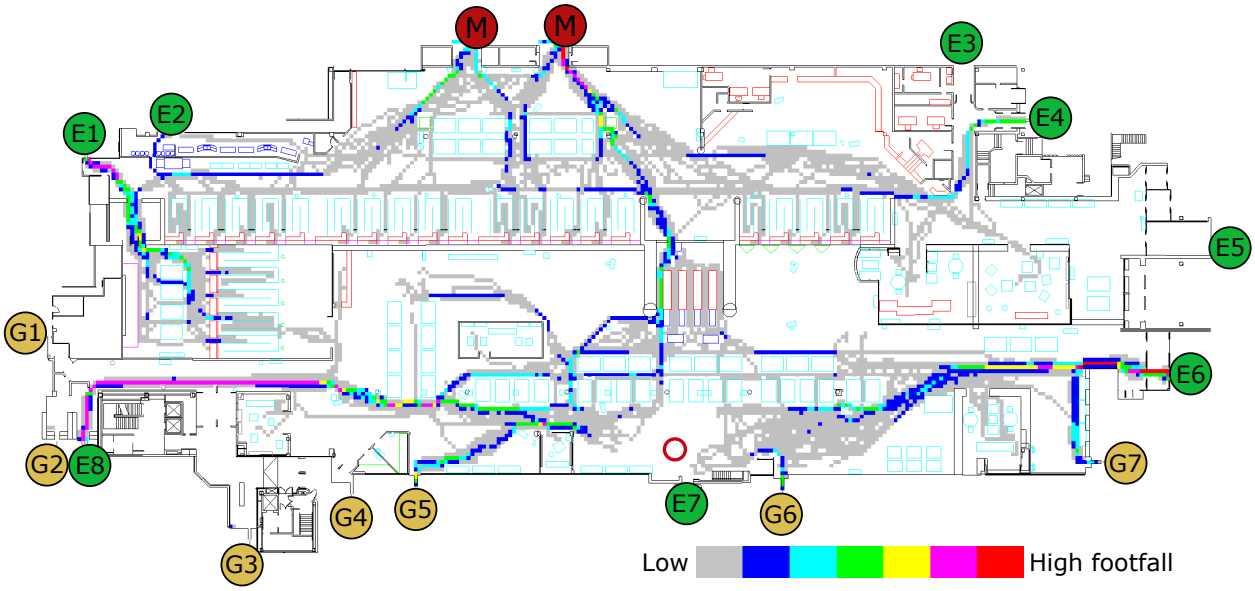


Figure 6.22: Secure an area: Group 4, Run 1, footfall contour

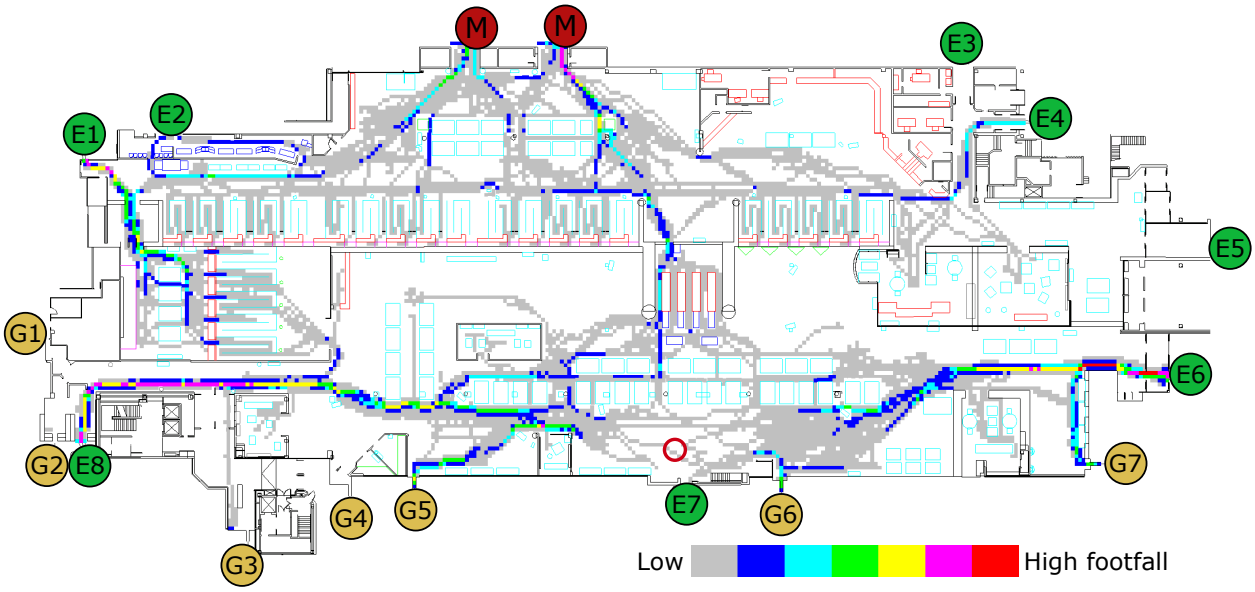


Figure 6.23: Secure an area: Group 4, Run 2, footfall contour

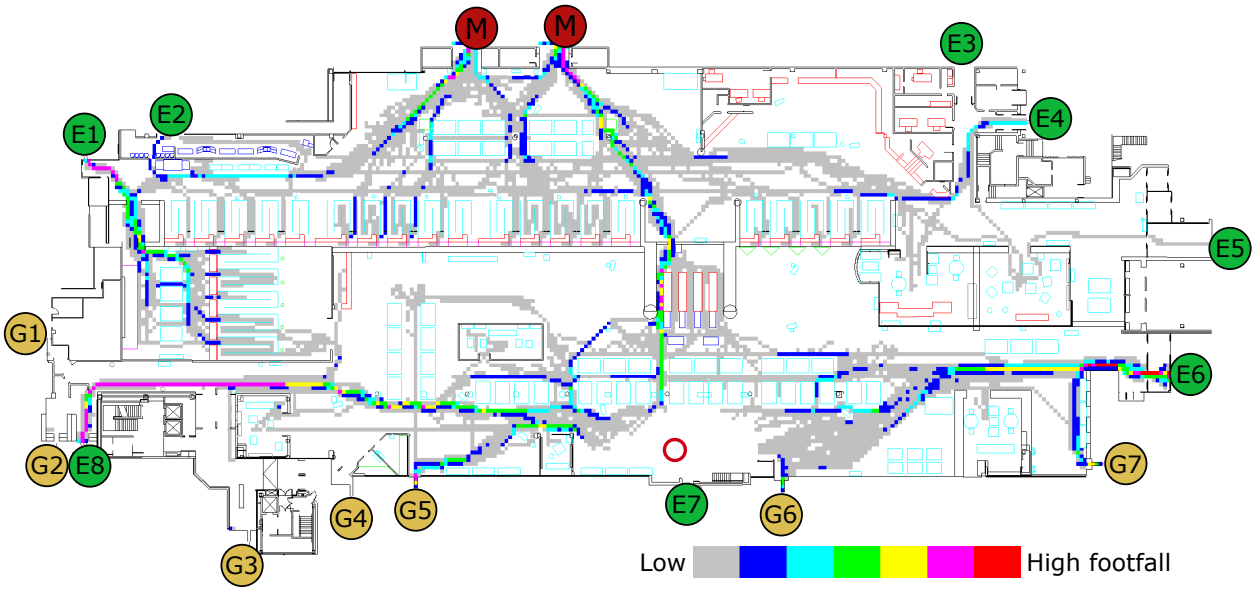


Figure 6.24: Secure an area: Group 4, Run 3, footfall contour

As an example, Group 4 managed to improve their results over multiple runs as can be seen in Figures 6.22, 6.23 and 6.24 by the fact that no footfall contours pass over (through) the red circle representing the location of the suspect bag. One of the users in Group 4 successfully secured their part in the exercise in the first run (see Figure 6.22) whereas the other user let some agents through to evacuate from the bottom exit. The first attempt by the participants to improve on the results actually made the results worse as can be seen in Figure 6.23 where the footfall contour passes over the red circle in comparison to only getting in its vicinity when agents exited through E7. However, once attempting to improve the results the second time the results were clearly better as the participants were able to prevent anyone from approaching the secured package (highlighted in red circle) and from using Exit E7. Thus, the group achieved their best footfall contour from all three runs on their final run (see Figure 6.24).

Table 6.6: Secure an area use case participant questions

	How realistic / realistically...
1	... you thought the simulation was?
2	... did the agents evacuate the building?
3	... did the agents respond to your gestures?
4	... did the agents move / navigate?

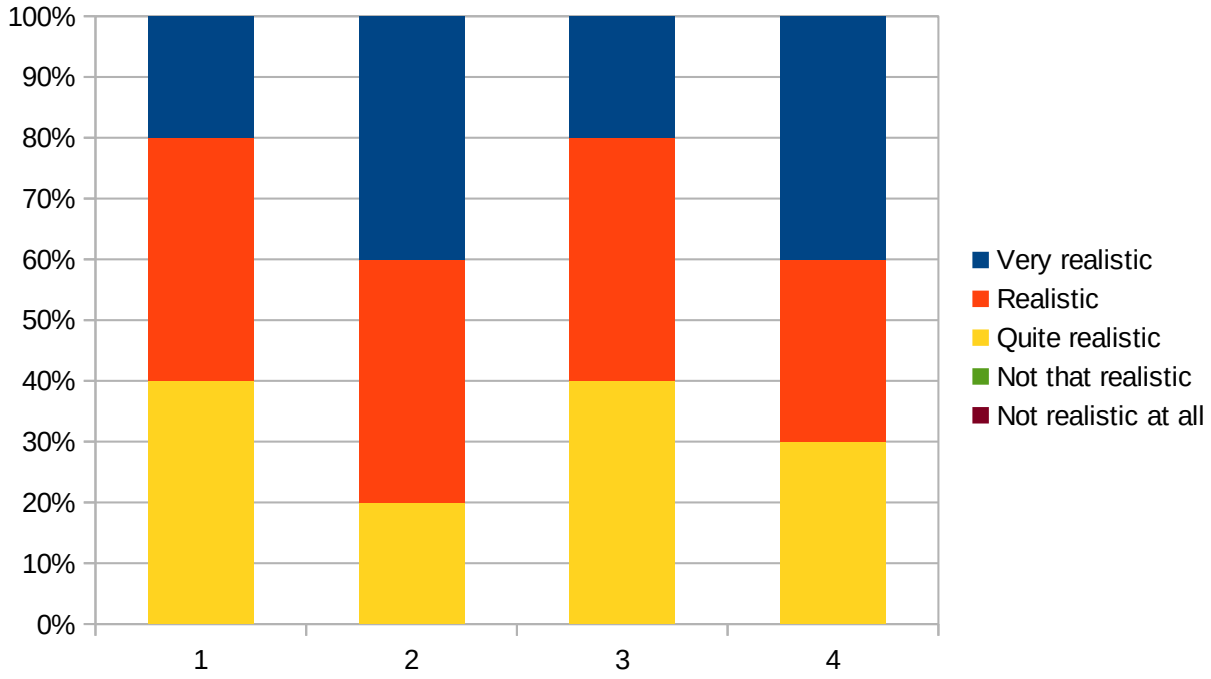


Figure 6.25: Secure an area use case participant responses to questions in Table 6.6

All participants felt that the simulation was at least *quite realistic* as seen from the feedback collected by the questionnaire in Appendix C (see Table 6.6 for the questions and Figure 6.25 for the answers). Agent behaviour, including evacuation from the building, movement and navigation as well as responding to gestures were thought to be at least *quite realistic* by all participants and at least 60% thought the behaviour was realistic.

The results in this section and in Section 6.5.2 show the ability to use the IRIS system to not only make computer-controlled agents respond and start evacuating sooner, but also to directly control the routes the agents adopt. This enables the IRIS system to be used by teams of individuals to undertake crowd management training in large areas, and with large numbers of agents, something that may not be possible via live training exercises. The results from a given scenario can be directly quantified, viewed and assessed, thereby enabling trainers to determine whether the desired training outcomes were achieved by the users.

6.7 General scenario participant feedback

General feedback was collected from the participants as part of the exercises by the questionnaire in Appendix C. The questions can be seen in Table 6.7 below and the answers in Figure 6.26 below. The questionnaire also provided optional sections for written comments for additional feedback which are also discussed in this section.

Table 6.7: General participant questions

	How accurately would you say the following statements describe you or your prior experience?
1	I feel that a system like this one, once it has been fully developed, could enhance current training methods. (In what way? Please also comment below)
2	I feel that a system like this one, once it has been fully developed, would be an effective tool to identify, develop and explore new tactics when the threat or trends change.
3	I feel that the statistics and assessment methods shown/described today will give sufficient performance feedback.
4	I think other organisations would benefit from using this kind of system.
5	I think a system like this one would enable you to exercise your training in decision making, analytical thinking and problem-solving skills.
6	I found the movement controls (keyboard and mouse) intuitive to use.
7	I found the gesture controls easy to use (keyboard).

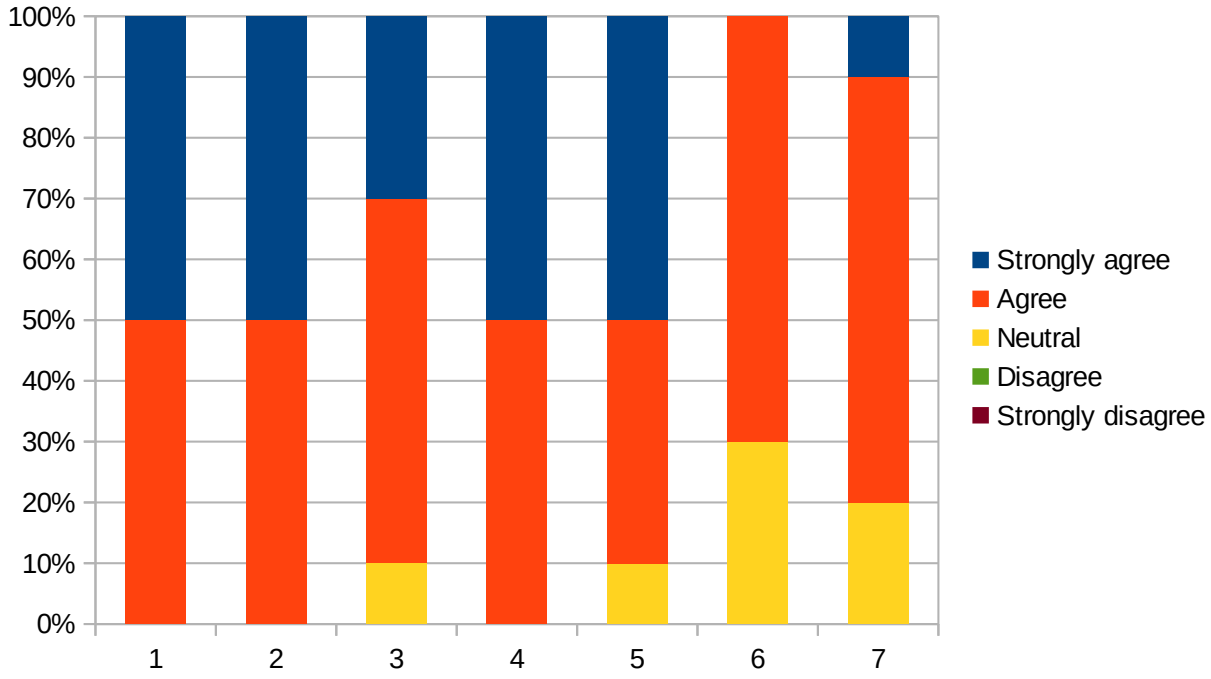


Figure 6.26: General participant responses to questions in Table 6.7

All participants unanimously agree that the IRIS system could enhance current training methods, once fully developed (see Figure 6.26). Equally the participants thought that the IRIS system would be an effective tool to identify, develop and explore new tactics when threat or trends change, once fully developed. The participants also thought other organisations would benefit from the IRIS system. 90% of the participants think the IRIS system as used by the participants already provides sufficient performance feedback and would enable training in decision making, analytical thinking and problem-solving skills. However, while 70% or more of the participants (see Figure 6.26) agree that the keyboard and mouse controls are intuitive and easy to use, this is also the area where the IRIS system achieved its lowest score amongst the participants.

From the written feedback it was identified that the controls need to be improved when developing the IRIS system further as they were breaking the immersion for some of the users. Another criticism was the collision detection, especially with other agents, this could be improved by the use of a physics engine. The IRIS system currently does not allow pushing of computer-controlled agents and therefore makes it difficult to navigate in dense crowds. The grid-based nature of the finite movement model in EXODUS makes it difficult

to simulate pushing in smaller increments than half a metre. This will need further research to implement a realistic collision behaviour with other agents. Some participants wished to be able to control the loudness of the voice commands as not everyone would shout equally loud. As sound and audio were not implemented as part of the IRIS system, this was reducing the immersion that the participants experienced. This is part of the further work. Some participants wrote that they could not see the fire (flames) in the smoke as there was a lot of smoke, but nothing was visibly burning. The flames were not rendered, and their visualisation was not the focus of this research but including the flames in the rendering would add realism in the final version.

6.8 The validity of the pedestrian simulation software

This section aims to discuss how the validation of buildingEXODUS is affected by the user interaction introduced in this thesis and how the existing validation affects the realism of the VR for different users, such as the first-person view and the birds-eye view.

buildingEXODUS has undergone multiple validations [82, 195–206] which include qualitative and quantitative validations [191]. buildingEXODUS also benefits from separate validations of airEXODUS and maritimeEXODUS as they share many of the same components. [191] However, this section focuses on the validation of buildingEXODUS as it is the version of EXODUS linked to the IRIS system in this thesis.

The discussion on validation in this section is based on the validation definition by the ISO (International Organization for Standardization). The ISO definition of validation is defined as *"process of determining the degree to which a calculation method is an accurate representation of the real world from the perspective of the intended uses of the calculation method"* [176, 177].

6.8.1 Validity of first-person view

buildingEXODUS has not featured a first-person view in any of the known validation cases. However, the qualitative validation cases [82,196,197,206] have shown that buildingEXODUS produces reasonable agreement with the experimental data in terms of locations, movement and paths for the agents within structures.

These published validation cases have been limited to 2D representations of computer simulated agents. 2D data does not contain all the necessary information to show visually accurate 3D representation of the same scenario. Person to person, person to geometry and person to hazard interaction [204] in 3D includes the movement (i.e. the animation) of different parts of the body, which are not present in the 2D representation in buildingEXODUS. A number of actions are represented in EXODUS, but these typically require an agent to visually stand still for the duration of the action before continuing their agenda. The visual fidelity of agents performing various actions had to be added to the 3D representation of the simulation.

The 2D qualitative validation does not fully translate to a valid 3D view in the first-person. This is due to the addition of the extra dimension requiring adding extra information not present in the 2D representation.

6.8.2 User interaction effect on validation

The user interaction capabilities introduced in this research have not been validated. Currently the literature lacks data to enable accurate virtual representation of the effect of hand gestures and voice commands on virtual crowds. However, the user can put the data into the model once the data becomes available. The IRIS system could be validated once a suitable validation case becomes available.

The IRIS system requires very little modification to any pedestrian model. This can be seen from the changes introduced to the EXODUS model as shown in Figure 6.27 and outlined later in this section. This allows the majority of the pedestrian models to stay

accurate to the previous validations, but any user interaction with the model will likely modify the outcome of a simulation as this chapter has shown.

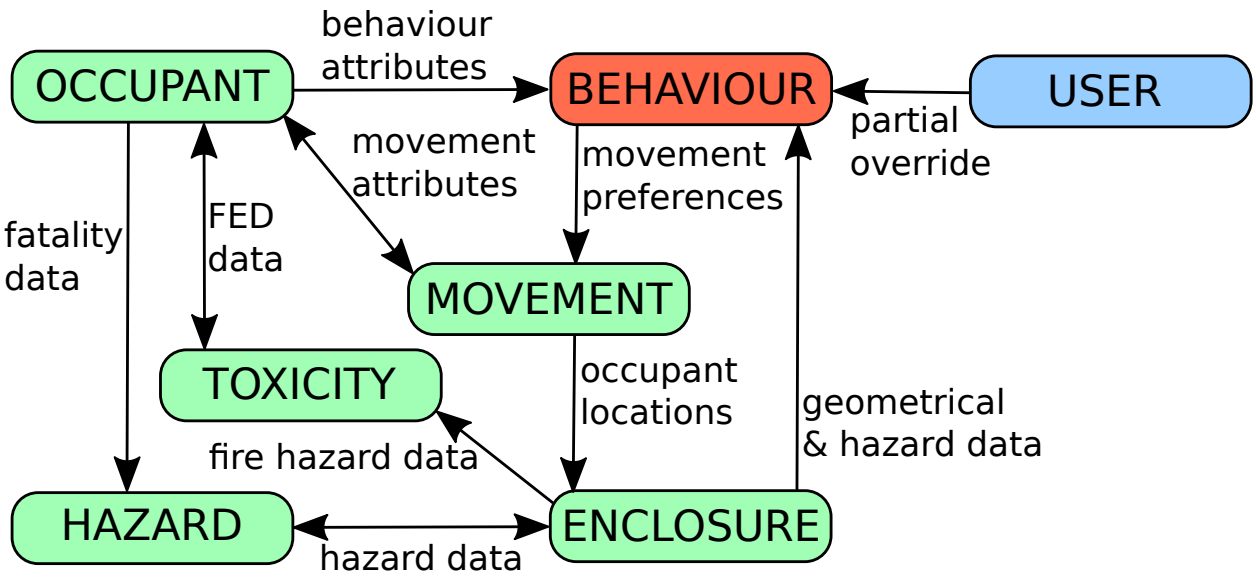


Figure 6.27: EXODUS sub-model user interaction

Figure 6.27 shows the original EXODUS sub-model structure (in green and red) and their interaction [191,201]. It also shows the addition of the user-component, which interacts directly with the *behaviour* sub-model. The *behaviour* sub-model is the only sub-model that required any modification in the EXODUS software to enable EXODUS to be linked to the IRIS system. All the original sub-models working together have been validated as a whole. The *behaviour* sub-model has been modified in two ways as detailed in the next two paragraphs.

The first modification was to enable the user to override part of the behaviour capabilities, such as the decision making of which way the agent wants to move or look. However, not all behaviour capabilities can be overridden by the user, such as an involuntary crawling in toxic smoke or in hot temperatures. The involuntary crawling could be made optional in the model and let it be overridden by the user, but this was not the case in this research. The *movement* sub-model still decides the capability and the actual action of the movement, such as preventing the agent to move when there is another agent or obstacle occupying and blocking the space, and how fast the agent can move.

The second modification for the *behaviour* sub-model was to introduce the new behaviour

capability to respond to hand gestures and voice commands given by another virtual agent. No other new behaviours were introduced, or other modifications made to the *behaviour* sub-model. This leaves a relatively original version of the model still intact with very little actual modification.

6.9 Summary

This chapter has detailed three case studies which demonstrate the contributions of the research carried out and presented in this thesis. Case study one, Assisted Airport Evacuation without smoke (Section 6.5), demonstrates significant impact from users on evacuation performance when immersed into the simulation and interacting with computer-controlled agents. Case study two, Assisted Airport Evacuation in smoke (Section 6.5), demonstrates significant impact from accurate smoke visibility reduction on users by significantly reducing their performance in comparison to the first case study. Case study three (Section 6.6), Securing an area, demonstrates clear learning and improvement through practising the same scenario over three consecutive runs.

This chapter has detailed three case studies for this research. The first two cases involved evacuating an airport with large crowds, with and without smoke. The last case involved securing an area where a suspicious bag had been reported. The case studies were performed on 10 participants, who formed 5 groups (pairs). Each group performed each of the exercises three times. The simulation results and a questionnaire feedback were collected from the participants after each run.

From the feedback of the participants, the IRIS system was generally agreed by the participants to be realistic. The simulation, the evacuation of agents, the movement and navigation of agents as well as the response to user gestures and commands were visually agreed by the participants to be realistic. Areas for improvement were also identified, such as the inclusion of sound, better collision behaviour and better user controls. All participants successfully completed all the exercises and were able to improve their performance, in terms of speeding up the evacuation, over multiple runs using the IRIS system.

This chapter has demonstrated that using virtual environments with simulated human behaviour with computational pedestrian evacuation models, it is possible to create advanced scenarios that can be quickly reiterated for testing procedures and training. The limitations of the computational models can be enhanced with complex human behaviour by the introduction of the human participants as part of the simulation. The enhancement provides the possibility to add key functionality to any computational human behaviour model through adding behaviours that are inherently missing from the model or difficult, time consuming, or impossible to otherwise implement.

Chapter 7

CONCLUSIONS AND FUTURE WORK

This thesis proposed a practical solution to enhance computational pedestrian models as well as virtual training environments (VTE) for emergency scenarios. Current VTEs used for training first responders (such as firefighters and law enforcement officers) lack credible human behaviour representation and realistic representation of fire and its products (i.e. smoke, heat and toxic substances) (identified in Chapter 2). The proposed solution linked a pedestrian behaviour model, a fire hazard (i.e. the modelling of the impact of smoke, heat and toxic substances on a simulated population) model, a CFD fire model and a virtual reality (VR) environment together to create an immersive, real-time interactive simulation (IRIS) system (see Chapters 1 and 2, 3). This is the first research contribution as described in Section 1.3.1 in Chapter 1.

The second research contribution as described in Section 1.3.2 in Chapter 1 was the introduction of a new efficient and accurate smoke visibility algorithm in Chapter 4. The smoke visibility is represented in the virtual reality environment and its effectiveness confirmed using current state of the art smoke visibility research to validate the accuracy of the smoke visibility in the virtual reality environment (see Chapter 4). A set of new validation tests (see Section 4.7) were introduced to compare the virtual smoke to real-world results. The tests with 10 external participants demonstrated the accuracy of the visibility reduction in smoke as observed from a computer monitor. The new proposed solutions outlined in Chapters 3, 4 and 5 does enhance the current state of the art virtual reality training capabilities through

immersive and reactive real-time computational pedestrian simulation. The stochastic and reactive outcomes of the pedestrian simulations can play a key role in the trainee's learning process to skilfully adapt to new unpredictable scenarios.

The proposed IRIS system enabled the representation of credible human behaviour in VTEs by providing real-time user interaction between user-controlled agents and computer controlled-agents in a pedestrian behaviour model simulation (see Chapters 5 and 6). The VTE capabilities are further enhanced by the representation of fire and its effects (i.e. smoke, heat and toxic substances) on the user-controlled and computer-controlled agents (see Chapters 4 and 6). This was achieved by using realistic fire data from a validated CFD Fire simulation and modelling its effects in the evacuation model on user-controlled agents as well as computer-controlled agents. The user can therefore see the impact of the fire and its products based on the best scientific understanding (i.e. smoke, heat and toxic substances) on the user-controlled agents as well as the computer-controlled agents. As demonstrated in the case studies with 10 external participants in Chapter 6, multiple human participants can take control of agents that are part of the simulation and interact with other agents using virtual hand gestures and voice commands. This is the third research contribution as described in Section 1.3.3 in Chapter 1. The real-time user interaction influences the outcome of the simulation, and therefore extending capabilities of the behaviour model. This allows the user to be able to make realistic, fire emergency induced, decisions on the user-controlled agents as the users are immersed in the VR environment with a realistic representation of fire hazards.

The IRIS system, as outlined in Chapters 1 and 3, has been shown to be a useful tool in planning, designing, testing and iterating on procedures, such as safety procedures (see Chapter 6). Using the IRIS system, the coordinator has a clearer overview of the scenario since they can effectively adopt a bird's-eye-view approach, fly around and get in close to what the participants are doing without getting in the way of the participant. The IRIS system can also be used to further record and analyse human behaviour in a virtual environment with accurate and detailed step-by-step recordings and simulation summaries. The interactivity model with crowds and individuals alike (IRIS system) as detailed in this thesis provides clear groundwork towards a wider adaptation of crowd management and crowd control features through user interactivity with computational pedestrian simulation models

(as discussed in Chapters 2 and 6). The proposed solution (the IRIS system) will enable a competent user to adapt an interactive system with any of the numerous pedestrian models currently in active use (as discussed in Chapter 2), with an accurate smoke visibility representation for better immersion.

The IRIS system and the real-time accurate fire smoke visibility visualisation have wide-reaching implications, not only providing a more realistic virtual environment where certain actual emergency procedures and training can be evaluated and performed in a holistic way, but also impacting on more general applications, such as the $100bn video game entertainment industry [27], where more immersive and realistic environments are in demand [28] (i.e. enabling more accurate representation of smoke in real-time).

It is important to note that while the research in this thesis used proprietary software as part of the IRIS system, all parts of the IRIS system can be replaced directly by alternative software packages that have source code available. The pedestrian behaviour model, buildingEXODUS, could be replaced by VADERE[1], JuPedSim[2] or Menge[3]. The CFD fire model, SMARTFIRE, could be replaced by FDS[4]. The VR environment, vrEXODUS, could be replaced by a game engine such as Godot[5], Unity3D[6] or Unreal Engine[7]. The AUGGMED project demonstrated that the VR environment in the IRIS system can be replaced by a VR solution developed on the Unity3D game engine (see Section 1.6, Chapter1).

7.1 Key research questions revisited

Chapter 1 identified key research questions for the research presented. This section revisits the questions and summarises the key findings from this thesis in relation to these questions.

The main question that highlights the research goals of this thesis is proposed as follows:

[1]VADERE is freely available from `https://www.vadere.org/`
[2]JuPedSim is freely available from `http://www.jupedsim.org/`
[3]Menge is freely available from `http://gamma.cs.unc.edu/Menge/docs/code/menge/html/index.html`
[4]FDS is freely available from `https://pages.nist.gov/fds-smv/`
[5]Godot is freely available from `https://godotengine.org/`
[6]Unity3D is available from `https://unity.com/`
[7]Unreal Engine is available from `https://www.unrealengine.com/en-US/`

How can virtual reality training and the evaluation of procedures be improved through the development and use of immersive, real-time interactive simulation (IRIS)?

The main question immediately raises the following sub-questions and associated response.

What technologies are available to support the development of an immersive, real-time, interactive simulation (IRIS)?

Chapter 2 highlighted existing technologies that can be utilised for the IRIS system. Real-time ray tracing technology is not yet feasible to use in high fidelity graphical settings and the IRIS system has to rely on higher performing rasterisation graphics pipelines. Computer game engines have increased in popularity since the start of this research and provide an attractive approach to faster prototyping of new research. However, due to the generalisation of these game engines to fit a wider audience, the performance impact is unavoidable and, therefore, a tailored solution was necessary to reach the required performance of the IRIS system.

If the commenced research is to last longer than a brief lifecycle of the Microsoft operating system, a careful consideration should be taken to select a suitable graphics library (i.e. OpenGL vs Direct3D) (see Chapter 4). Also, careful consideration should be used in the selection of a collision model (i.e. physics engine) for the virtual reality, especially if it is developed separately from the linked validated pedestrian model to make sure that virtual world represents accurately the modelled behaviour and does not deviate unrealistically from it.

In Chapter 2, it was identified that the IRIS system should have the following capabilities:

- A virtual user-controlled agent with interaction capabilities with individuals or crowds of computer-controlled and simulated agents.
- Plausible interaction between user-controlled agents and computer-controlled agents using natural human hand gestures and voice commands.

- Fine or continuous movement model capability from the behaviour model as well as the modelling of individual and crowd behaviours.
- Immersive (natural) representation of the virtual world in emergency scenarios, especially in terms of accurate representation of visibility in fire smoke, effects of toxic gases and heat on computer-controlled agents as well as the user-controlled agent.
- Accurate smoke visibility and fire hazard data from a sophisticated CFD fire simulation tool.
- Run in real-time.

The IRIS system needed to link the following software components:

- A validated computational pedestrian/evacuation behaviour model (i.e. plausible individual and crowd behaviours).
- A hazard model (i.e. smoke, heat and toxic product representation on population).
- CFD fire data from a CFD fire simulation tool (i.e. realistic fire hazard in emergency scenario).
- A virtual reality environment (i.e. immersing the user and to provide interactivity capabilities with the pedestrian simulation).

What types of interaction behaviours are necessary to facilitate crowd interaction?

As was discovered in Chapter 2 humans interact with each other face-to-face predominantly using voice and gestures. However, the gestures and voice can be difficult to understand, especially as the communication distance grows, until the gestures and the voice can no longer be comprehended. This is a key modelling aspect when designing the hand gestures and the voice commands in this research. All members of a crowd have specific roles, where some are more influential than others. Normally a role with a high influence value such as an official can act as an authority over the crowd.

As was discussed in Chapter 2 in emergency scenarios an authority figure can play a significant role in managing crowds. When the authority issues commands and gives instructions to a crowd in an emergency scenario, the crowd tends to have full eventual compliance to the authority. However, not all members of the crowd comply immediately as the compliance of an individual is highly dependent on the compliance of fellow individual crowd members. The compliance of individuals increases as the other members of the crowd comply to the hand gesture and voice commands.

How can the immersion and the realism of the real-time virtual reality in emergency scenarios be improved?

There are several factors that affect the immersion of a virtual reality environment. One commonly understood factor is the visual quality and the graphical fidelity of the represented virtual world. However, recent research as outlined in Chapter 2 has indicated that the realism and presenting naturally available information to the user plays a key factor in immersing the user and compelling the user to behave naturally (i.e. to mimic real-life behaviour) in a virtual environment (see Chapter 2). The research also showed that providing artificial information for the user encourages unavoidable artificial behaviour that does not exist in a real-world scenario where such artificial information is not present. The artificial information may include a presence of a heads-up display (HUD) detailing the user-controlled agent's attributes or rendering artificial artefacts in the virtual world such as a trajectory of an object.

To achieve realistic human behaviour in a simulated emergency scenario in a virtual world, the IRIS system uses a validated evacuation model together with fire data from a validated CFD Fire simulation. The use of validated behaviour models allows the user to be confident that the behaviour within the virtual world is as representative as possible of the behaviour in real-world scenarios. Therefore, the testing of procedures and training of first responders will be based on the best possible science and responsive behaviours in the most plausible way during emergency events. However, it is important to note that while the behaviour of the pedestrian model is validated the interactivity introduced in this thesis does affect the behaviour of the simulated agents. The interaction with the agents and their resulting behaviour was considered realistic by the participants of the case studies in Chapter

6. The impact on the validation is further discussed in Section 6.8.

What physical characteristics of a fire hazard can practically be included in virtual reality when applying emergency procedures?

A fire produces the following products that are of interest to a human hazard model: smoke, heat and toxic substances (as was discussed in Chapter 2). Heat and toxic substances can be invisible to a human eye; however, the effects of these products such as irritation can be very noticeable. Virtual reality cannot subject the user to realistic heat or toxic substances but can subject the user's agent to these products. The user can also see the effect of these products on surrounding agents in the virtual environment (i.e. crawling or incapacitated) as he would see in a real-life scenario. Smoke can be visualised in a virtual reality environment and can be used to obscure the user's vision as the user looks through the eyes of the user-controlled agent. The published literature as was discussed in Chapter 2 does not propose an accurate way of representing smoke visibility in real-time in a virtual environment and therefore the research in this thesis had to find a new solution to this problem. The linked IRIS system is discussed in Chapter 3. The case to validate the smoke visibility with external users is found in Chapter 4. The smoke visibility is further discussed and demonstrated in Chapter 6.

7.2 Research related future work

This section will be examining potential areas of further work related to the research conducted in this thesis. Implementation related further work that aims to improve the capabilities of the IRIS system and its use cases are covered in Section 7.3.

7.2.1 Calibration of hand gestures and voice commands

This thesis proposed 8 new hand gestures and voice commands, which are listed in Table 5.1, Chapter 5. As was discussed in Section 5.1 in Chapter 5 the gestures and commands were identified as part of the AUGGMED project (see Section 1.6, Chapter1). The gestures and

commands were described and verified in Chapter 5 and then further demonstrated in case studies in Chapter 6. While the appearance of the behaviour introduced to the computer-controlled agent through the use of the gestures and commands were considered realistic by external participants of the case studies, they have not been calibrated with real-world data.

The real-world data to calibrate the gestures and commands would likely come from a research study or from a real-world case. A real-world case could be an incident where detailed data can be extracted from evidence such as voice and video recordings to create a calibration data set. This data set would then be used to fill in the parameters for the hand gestures and voice commands. There will need to be further verification tests to ensure that the results from the IRIS system match the calibration data set that is used to calibrate the IRIS system. The calibrated behaviour should be validated with a different data set to bring confidence to the calibration.

7.2.2 Validation of hand gestures and voice commands

Once the calibration of the hand gestures and voice commands as detailed in Section 7.2.1 has been accomplished, the gestures and commands can then be validated. The calibration stage would ensure that the behaviour represented in the IRIS system would accurately mimic the behaviour in the calibration data set. The validation stage would enhance the calibration stage by providing further confidence in the expected behaviour and, therefore allow the IRIS system to be used as a more accurate interaction tool with evacuation models. The validation would involve comparing the virtual agent behaviour from the interaction with gestures to real-world behaviour obtained through an actual real-world case or research study.

The validation would help to identify whether the parameters suggested in Table 5.2 in Chapter 5 are sufficient to define all the important aspects of a gesture or command. However, before validation can be done to the gestures and commands, the parameter values used for the gestures and commands (see Table 5.3, Chapter 5) need to be replaced with real-world data. The effect of gestures and commands on the pedestrian model validation is discussed in Section 6.8 in Chapter 6.

7.2.3 More comprehensive case studies with more participants

The IRIS system has been tested with multiple users in this thesis and used successfully in the AUGGMED EU-Project (see Section 1.6, Chapter1). However, the system needs further testing in a larger variety of settings. The testing should include more procedures and training to find the extent of the usability of the IRIS system.

The efficient and accurate smoke rendering which is detailed and validated in Chapter 4 has also been tested on multiple users but testing of a larger variety of objects on multiple people would allow better mapping of the light extinction coefficient to real-world objects. Such comparison could include testing of illuminated and non-illuminated exit signs and different coloured smoke.

Tests to compare real-world smoke in a variety of colours, smoke age, and lighting conditions should be tested and compared more exhaustively to virtual smoke to establish more accurate smoke representation in VR. The same people who participate in the real-world smoke visibility tests should also be included in the virtual-world visibility tests to make a better comparison between the real smoke and the replicated, virtual smoke.

7.2.4 Comparison of the IRIS system to real-world training

This research proposed the IRIS system as a suitable tool for training. While it is clear that training can be accomplished with the IRIS system, it is not clear how well it compares to real-world training. Therefore, it is important to identify and establish a set of metrics to compare a real-world training with the IRIS system. This would involve running a set of real-world training scenarios that can be accurately represented and duplicated in a VR environment, the IRIS system. The tests would likely include control groups where, between the groups, the participants would have similar characteristics and capabilities in relation to the tasks given.

Performing these real-world training tests would establish a set of training objectives and performance metrics that the participants aim to reach. The training performance results

could then be compared between the real-world and virtual-world exercises to identify the improvements and regressions for using IRIS based VTEs. Financial cost to benefit ratio could be established to compare the real-world training costs and benefits to IRIS based VTE costs and benefits. The suitability of the IRIS system for various training scenarios should also be established.

One possible technique would be to include a 3-step process where in the first step one group is trained using traditional real-world methods and in the second step (possibly a parallel step) separate group is trained using an IRIS based VTE. In the final step the training objectives could be measured through a questionnaire or another method to assess knowledge. Otherwise reinforcement learning could be used to test the students in a real-world exercise, where both groups would need to translate the skills learned into a new scenario, however, this may be difficult to assess. At this point in time it is assumed that the aim of training with the IRIS system is likely to develop new skills or improve existing skills for real-world rather than virtual scenarios.

7.2.5 Compare the stochastic model with a deterministic model for further case studies

Chapter 6 details the case studies performed as part of this research. The case studies and the IRIS system utilised a stochastic evacuation model, buildingEXODUS. The randomness of a stochastic model introduces variability to base case results as seen in Chapter 6. The variability can hide the user impact in its noise.

The IRIS system is designed to be modular where each component could be replaced with another suitable component. This was demonstrated in the AUGGMED EU-Project (see Section 1.6, Chapter1) where the VR component, vrEXODUS, was replaced with a Unity3D game-engine based VR environment. Therefore, the buildingEXODUS component, which is stochastic, could be replaced with a deterministic model.

Instead of replacing buildingEXODUS with a different model, the other solution would be to introduce significant changes to the buildingEXODUS software to reduce the variability

in the output of the model. Many aspects of the EXODUS model are already deterministic, but some key parts are determined randomly or by probabilities. These include the local behaviour such as conflict resolutions in people to people interactions as well as in the movement model where a route cannot be determined from a potential map. Additionally, the user interaction with the simulation should start from fixed initial conditions, which includes setting agent attributes to fixed values, including location, age, gender, mobility, drive, response time and itineraries. However, this reduces the training opportunities from repeated runs as the user will learn the locations and behaviours of agents at the fixed starting point of the simulation. Therefore, it is necessary to set multiple fixed starting points in a deterministic model to provide deterministic variability to the outcome over multiple runs. This would provide a fixed base case for each run where the user impact may be more clearly visible than what is presented in Chapter 6. It is important to note that the user interaction with hand gestures and voice commands presented in this thesis are based on probabilities and would require an alternative approach to make them deterministic.

Changing an existing model to reduce the variability in its output may invalidate any validation which the model may have gone through. The aim of this research was to keep any modifications to an existing model to a minimum. Even the minimal but necessary changes introduced to the model, such as the commands and gestures, are potentially invalidating the model (see Section 6.8 in Chapter 6). Therefore, it may be more viable to use an existing deterministic model.

While the outcome of the deterministic model may be fully determined for a base case, similarly to a fixed seed in a pseudo random number generator, the user interaction may still cause a butterfly effect in the outcome of the model. In this case the outcome of the initially deterministic model may no longer be a representative outcome of a real-world situation in similar conditions. Reducing the variability of the model may improve the assessment accuracy of the training or testing of procedures but may also reduce the training opportunities or the impact from edge cases in testing of procedures. The comparison of different models and especially between stochastic and deterministic models as a key part of the IRIS system should be researched further.

7.2.6 Sensitivity analysis of the gestures and commands

Chapter 5 described key attributes (Table 5.2) for the hand gestures and voice commands based on the literature review in Chapter 2. The gestures and commands were verified in Chapter 5 to function as intended. The user feedback from the trials in Chapter 6 identified the response of the computer-controlled agents to the hand gestures and voice commands appeared realistic (Table 6.16 and Table 6.25). However, most of the attribute values for the gestures and commands were constants (Table 5.3) with the exception of the radius attribute. The radius attribute is reduced to the maximum visibility distance in smoke if the visibility is less than the assigned value of the radius.

The case studies in Chapter 6 explored the impact of smoke on the hand gestures and voice commands. To determine the full extent of the impact of the attributes in Table 5.3, they should be explored further with parameter sweeps with additional test and verification cases as required. These additional test cases could be used to identify suitable values and ranges for each command and provide insight into how they can be varied to achieve a desired outcome.

7.3 Implementation related future work

Section 7.2 identified further work related to research. This section identifies potential future work for the IRIS system to improve its capabilities and use cases.

7.3.1 Graphical Fidelity

Graphics Library

The first version of OpenGL, version 1.0, was released in 1992. Technology has since advanced a great deal. The main problem with OpenGL is that it was designed for single core CPUs as a state machine. This has been its main limitation since multi-core consumer CPUs became popular. AMD originally developed Mantle graphics library to tackle these problems, which

was referred to as glNext, a replacement to OpenGL. AMD donated their work to Khronos Group, which renamed Mantle to Vulkan. Vulkan is the official next generation OpenGL, aimed at modern hardware, to supersede OpenGL in the future [207]. However, its main problem is its compatibility, as it requires modern hardware. Unless the performance [208] of OpenGL is a problem in the developed application or the application requires a modern Vulkan specific capability, there is little incentive to break the backwards compatibility with older hardware by not using OpenGL.

Vulkan was announced in 2015 and did not exist when this research started in 2011 [136]. This leaves Vulkan adaptation as a future work. It is important to note that the IRIS system requires high performance to stay in real-time when rendering complex scenarios and therefore the added performance from Vulkan would be beneficial.

3D models

The 3D models used in this thesis and by vrEXODUS were bought off the shelf, and then modified by FSEG, before providing them to be used in vrEXODUS software program. These off the shelf models do not meet the same quality standards as purpose-built models would. Also, the animations are off the shelf applied animations, which do not work really well within the 0.5 metres by 0.5 metres space requirements set by the finite movement network in the buildingEXODUS software program. Furthermore, there are only 2 different models, with 20 different diffuse textures for each model. The 2 models are a male and a female model. This leaves out elderly people and children. The two models also leave out different body types, and disability representations. Multiple ethnic groups have been covered by the variety of textures available for these models.

The used off the shelf agent models further lack many of the modern texture maps expected by modern graphics standards [209]. The only texture maps provided are diffuse maps. Normal maps and bump maps would provide additional detail to each rasterised pixel, replacing many unnecessary triangles from the mesh. As an end result it enhances the details, while increasing performance. Specular maps would provide highlights, separating shiny and matt areas, such as belt buckle from cloth and eyes from skin. There are also many other

maps that can be used to enhance the visual quality of the image such as opacity maps, reflection maps and glow maps. Opacity provide transparency to specific areas, reducing the necessity to use triangles for that purpose. Reflection maps would make the surface appear reflective, such as metal. Glow maps would let an area to ignore light, such as an illuminated exit sign.

Other missing models from EXODUS include interior and exterior objects, such as vegetation and foliage, variety of chairs, lamps, tables, desks etc. These are not modelled in EXODUS and are necessary to include in the imported model from a 3D modelling software, such as Blender. A detailed 3D model of the airport structure was artistically created as part of the AUGGMED project (see Section 1.6, Chapter1) and this model was utilised in this thesis.

Lighting

The lighting is overly simplified. EXODUS lacks any capability to handle lights, such as ceiling lights. These would play a huge part in illuminating the scene and smoke correctly. It would help the user to distinguish between different objects in the geometry.

Part of lighting is shadows. With the lack of lights and with global ambient illumination, there is no correct way to address shadows in vrEXODUS. Perhaps post-processing capability could be added to vrEXODUS to address these limitations in EXODUS. However, this would add false data to the final image as EXODUS does not deal with this level of visibility when processing the population in any given scene.

Improving visibility in smoke particle fidelity

The smoke rendering research detailed in this thesis uses the existing zone model format designed for 2D pedestrian simulation and as was discussed: some changes were necessary to improve the visual representation in a 3D space. However, this can be taken a step further by increasing the granularity of the zones in horizontal spaces, which enables a smoother transition as smoke travels across a geometry/structure. The number of timesteps may also

be increased, but the current linear interpolation eliminates popping movement behaviour.

With hardware acceleration of modern graphics hardware, a high-end graphics card is capable of rendering hundreds of thousands or even millions of quads in real-time. Utilising e.g. geometry shaders, the smoke visibility data can be streamed to the graphics card and the quads can be generated on the fly, enabling the billboard smoke particle rendering detailed in this thesis to be rendered more efficiently. This level of optimisation would enable more complex smoke zone layout rendering and potentially the rendering of individual subdomains of a CFD scenario. Depending on the complexity of the CFD scenario, the number of subdomains can easily reach hundreds of thousands or millions of individual subdomains. While rendering individual subdomains enables the greatest level of fidelity possible from a CFD engine, the amount of data that needs to be processed and stored may not fit into the computer's memory i.e. when the simulation is complex and long enough with small timesteps, the amount of data can reach tens or even hundreds of gigabytes or more. While the larger simulations may be out of reach from the possibility of real-time rendering, smaller geometries such as an aircraft, a train, a small building or a naval vessel may be viable. This will in no doubt increase the visual quality, realism and therefore also immersion of the smoke rendering.

The smoke rendering technique detailed in this thesis suits well to large smoke zones by generating evenly spaced smoke particle grids. However, the even grids may not be suitable when rendering subdomain data from CFD engines as the subdomains may not necessarily be in uniform size and shape. Therefore, the spacing of the smoke particles in varying patterns and varying sizes would require further validation and testing.

Interactive real-time CFD fire simulation

The CFD fire simulation software used in this research was not capable of running in real-time and produce real-time fire data for the IRIS system (i.e. 400 second simulation (see Chapter 4) took 3 days to run). Once the CFD fire simulations can be run in real-time or faster, the IRIS system should be enhanced to enable further interactivity with fire data, such as firefighting, by providing real-time input to the real-time fire simulation.

7.3.2 Hardware and equipment

The application and tests covered in this thesis were performed with desktop PCs, with monitors, mouse and keyboard. What has not been covered is virtual reality headsets such as HTC Vive or Oculus Rift. Other input devices such as XBOX Kinect and Leap Motion hand tracking, or game controllers are not covered in this thesis. Haptic feedback or temperature-controlled surfaces are also not covered, such as wearable clothing. Smell related feedback is not covered either. As part of researching the appropriate interface for the user to provide input into the virtual reality, some preliminary prototypes on voice recognition over a microphone were done (i.e. the user says something to the microphone, which is interpreted by the computer and then the user-controlled agent does something in response), but due to time constraints not enough testing was done to draw any definitive conclusions at this stage and had to be left as future work.

Hardware such as body suits, cameras, headsets and user input devices can be used to provide additional interfaces for the user to immerse into the virtual reality. A wearable headset that covers the user's vision would stop the user being distracted by what is happening outside of the simulation. A body suit or cameras would be able to track some of the user's interactions with the simulation, such as which way the user is facing at any given time. An appropriate user input device may be a controller, control panel or even a microphone for voice input. The use of a microphone as well as other additional hardware were left as future work.

7.3.3 Audio feedback

Audio is a well-known stimulant towards a better immersion. Watching a movie without audio is a prime example of its importance. However, audio is not necessary for immersion, hence reading a captivating novel may be more immersive than watching a movie. Audio was purposefully not implemented. To adequately demonstrate audio quality and its impact, would be a huge undertaking and was not possible within the time constraints of this research.

Further a badly implemented audio could distract the user from the desired immersion.

But a correct and relevant audio cues in the future work could highly improve the immersion. These could include footsteps, gun shots, a bomb, or someone saying something. Also, ambient sounds, such as multiple busy crowds chatting in the background in an airport, could be used to increase the believability of the scene and situation. Ambient sounds can also work as audio cues, such as when a calm or even rushed chatter turns into alarming shouts and screams. Or just the sound of an alarm or a fire in the background can easily enhance the understanding of the situation. This could be assisted with a pair of noise blocking headsets to help the user to concentrate on the simulation, rather than be distracted by outside sounds.

Procedure tests could include a radio communication between multiple users. In a terrorist environment this radio communication could also be attacked. However, this kind of testing environment would require multiple rooms, one for each client, effectively blocking the usual capability of just shouting to one another outside the simulated environment. This would require using third party software or equipment, or the development of additional network capabilities. In the case studies presented in this thesis the communication between users happened outside VR (i.e. in the same room). The users spoke or shouted to each other to simulate the use of radio communication devices.

7.3.4 Optimisation

C and C++ languages are generally regarded as being high performance computer languages. These languages were used in the vrEXODUS and buildingEXODUS software for this thesis. However, even with the highest performing languages and compilers, it is easy to write slow software, especially when rapidly prototyping new capabilities. While the software (vrEXODUS and buildingEXODUS) runs in real-time and is fast enough for the demonstration cases in this thesis, it should be noted that FSEG provided state of the art, top of the range PC hardware for the work in this thesis. Typical commercial home PCs would not be expected to reach the same level of performance with current optimisation levels.

The network capabilities were tested in LAN environment with 1000mbps full-duplex connections between the server and clients. This is way above the bandwidth of a normal

domestic Internet connection. To run these cases over Internet, it should be noted that some further optimisations would be necessary on the server to reduce the amount of data being sent.

7.3.5 Commercialisation

To make the capabilities defined in this thesis available in a commercial product, there are a number of necessary user improvements which would need to be done. Some of these are identified in the following list:

- Most of the scenario settings are hard-coded into the software. These should be ideally made into forms and windows with text input areas, drop down menus, tick boxes and buttons.
- Server-side network handling code should support clients that drop out and join back later. This requires the server to memorise what information has been sent to other clients previously so that the new client can be synchronised accordingly. Obviously not all information is relevant and necessary to store, but it is necessary to store the latest snapshot of the current state and the necessary delta updates.
- Training or detailed documentation on how to use the software may be necessary to help the user understand what the software is capable of and how to use it properly. While keeping this in mind, it would be necessary to program in safeguards on the off chance that the user deviates from the typical usage.

7.3.6 Additional hand gestures and voice commands

The set of commands proposed in this thesis is not proposed as a comprehensive list of commands that a user may require in all emergency situations when interacting in the virtual environment. However, the commands were chosen as an example set of commands that can be used to manage individuals and crowds of agents in the IRIS system. Additional hand gestures and voice commands should be developed in the future versions of the IRIS

system to enable the user to give a larger variety of instructions, such as those required in a gun fight or managing a terrorist situation. It would be necessary to develop additional behaviours for the computer-controlled agents to enable the reaction to the additional hand gestures and voice commands.

It is important to note that enabling a very large variety of different hand gestures and voice commands, especially when given independently, can become difficult if each command is given its own button or button combination. Therefore, additional hardware capabilities may be necessary to include in the IRIS system, such as using a voice recognition through a microphone or gesture recognition through motion tracking.

The influence ranges of hand gestures and voice commands are currently base on a fixed value in this research. As a future work, the fixed ranges should be based on actual real-world data. Additionally, the affected visibility from surrounding agents should also be taken into account when calculating the comprehension probability of issued commands. This should also be based on real-world data and the visibility should be simulated in the evacuation model. Due to time constraints it was not feasible to perform these studies as part of this research and was left as future work.

In the current implementation of the voice command behaviour from computer-controlled agents, the agents do not search for the source of the voice command if they could hear it but cannot see it. In addition, the agents who could see the surrounding agents being influenced by a command, but cannot see the one issuing the command, do not obey the command. Implementing these missing behaviours would require further study and research to accurately represent real world behaviour, and thus is left as future work.

7.3.7 Network connection security

The current implementation assumes that all clients can be trusted. This means that the implementation is not designed for competitive gaming or other similar situations where a client may want to cheat the system or intentionally behave in a way that is unrealistic, such as flying in the air or teleporting instantly across the geometry. The target audience for this research is professionals who would intent to use the IRIS system in a serious way

and therefore cheating prevention is not considered important for this research.

What the security of this implementation does not cover is the authentication of the communication messages as well as their encryption. This, however, becomes essential when the programs communicate in an untrusted network such as Internet [210]. For the purpose of this research the programs are used in a trusted environment and network (i.e. local area network (LAN)) with trusted clients and therefore authenticating and cryptographically enhancing the secrecy and authenticity of the messages is not necessary.

The implementation is based on streamed network communication sockets (TCP) and therefore adding an extra layer of security is simple. The security can be added by wrapping the streaming plain text connection inside a TLS (Transport Layer Security) using for example OpenSSL library. This does not necessarily need a 3rd party certificate authority, because clients are provided by the same entity as the host. When the host program is set up and the client programs are distributed: the client programs can contain a copy of the host program's public key and therefore they can verify the authenticity of the host and securely exchange the keys for the TLS layer [210]. However, if there are multiple hosts (which is not the case during this research), then a certificate authority will make obtaining keys and exchanging them significantly easier, while maintaining reasonable security. The certificate authority can be the author of the host program and each running instance of the host program can obtain a unique key from the certificate authority for the lifetime of the instance. This key exchange and certificate authority parts of the program are not written during this research and is left as future work.

REFERENCES

[1] F. Williams-Bell, B. Kapralos, A. Hogue, B. Murphy, and E. Weckman, "Using serious games and virtual simulation for training in the fire service: A review," *Fire Technology*, vol. 51, pp. 553–584, May 2015. doi:10.1007/s10694-014-0398-1.

[2] A. BinSubaih, S. Maddock, and D. Romano, "Serious games for the police: Opportunities and challenges," *Special Reports & Studies Series at the Research & Studies Center (Dubai Police Academy)*, 2009.

[3] R. Bukowski and C. Séquin, "Interactive simulation of fire in virtual building environments," in *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, (New York, NY, USA), pp. 35–44, ACM Press/Addison-Wesley Publishing Co., 1997. doi:10.1145/258734.258757.

[4] T. S. Julien and C. Shaw, "Firefighter command training virtual environment," in *Proceedings of the 2003 Conference on Diversity in Computing*, TAPIA '03, (New York, NY, USA), pp. 30–33, ACM, 2003. doi:10.1145/948542.948549.

[5] A. Ren, C. Chen, and Y. Luo, "Simulation of emergency evacuation in virtual reality," *Tsinghua Science and Technology*, vol. 13, pp. 674–680, October 2008. doi:10.1016/S1007-0214(08)70110-X.

[6] Z. Xu, X. Lu, H. Guan, C. Chen, and A. Ren, "A virtual reality based fire training simulator with smoke hazard assessment capacity," *Advances in Engineering Software*, vol. 68, pp. 1 – 8, 2014. doi:10.1016/j.advengsoft.2013.10.004.

[7] M. Cha, S. Han, J. Lee, and B. Choi, "A virtual reality based fire training simulator integrated with fire dynamics data," *Fire Safety Journal*, vol. 50, pp. 12 – 24, 2012. doi:10.1016/j.firesaf.2012.01.004.

[8] R. Querrec, C. Buche, E. Maffre, and P. Chevaillier, "Multiagents systems for virtual environment for training. application to fire-fighting," *International Journal of Computers and Applications (IJCA)*, vol. 1, no. 1, pp. 25–34, 2004. doi:10.2316/Journal.208.2004.1.202-1453.

[9] A. Rahouti, G. Salze, R. Lovreglio, and S. Datoussaïd, "An immersive serious game for firefighting and evacuation training in healthcare facilities," *Int. J. Comput. Electr. Autom. Control Inf. Eng*, vol. 11, pp. 1038–1044, 2017.

[10] S. Shen, J. Gong, J. Liang, W. Li, D. Zhang, L. Huang, and G. Zhang, "A heterogeneous distributed virtual geographic environment—potential application in spatiotemporal behavior experiments," *ISPRS International Journal of Geo-Information*, vol. 7(2), no. 54, 2018. doi:10.3390/ijgi7020054.

[11] M. Pan and G. Niemeyer, "Catching a real ball in virtual reality," in *IEEE Virtual Reality 2017*, (Los Angeles, Californiam USA), March 2017. doi:10.1109/VR.2017.7892280.

[12] J. Dugdale, B. Pavard, N. Pallamin, M. E. Jed, and L. Maugan, "Emergency fire incident training in a virtual world," *Proceedings ISCRAM*, vol. 167, 2004.

[13] D. Helbing and P. Mukerji, "Crowd disasters as systemic failures: analysis of the love parade disaster," *EPJ Data Science*, vol. 1, p. 7, Jun 2012. doi:10.1140/epjds7.

[14] G. Still, *Crowd Dynamics*. PhD thesis, University of Warwick, August 2000.

[15] F. Meng and W. Zhang, "Way-finding during a fire emergency: an experimental study in a virtual environment," *Ergonomics*, vol. 57, no. 6, pp. 816–827, 2014. doi:10.1080/00140139.2014.904006.

[16] Marine Accident Investigation Branch, "Report on the investigation of a fatal accident during a vertical chute evacuation drill from the uk registered ro-ro ferry p&osl aquitaine in dover harbour on 9 october 2002." `https://www.gov.uk/maib-reports/positional-asphyxia-during-a-vertical-chute-evacuation-drill-from-ro-ro-passenger-ferry-p-osl-aquitaine-alongside-at-the-port-of-dover-england-with-loss-of-1-life`, January 2015. Accessed: August 26, 2018.

[17] K. Liu, X. Shen, A. Saddik, A. Boukerche, and N. Georganas, "Simsite: The hla/rti based emergency preparedness and response training simulation," in *Distributed Simulation and Real-Time Applications, 2007. DS-RT 2007. 11th IEEE International Symposium*, (Chania), pp. 59–63, IEEE, October 2007. doi:10.1109/DS-RT.2007.40.

[18] T. Jin, "Visibility through fire smoke (i)," *Bulletin of Japan Association for Fire Science and Engineering*, vol. 19, no. 2, pp. 1–8, 1970. doi:10.11196/kasai.19.2.1".

[19] T. Jin, "Visibility through fire smoke (ii)," *Bulletin of Japan Association for Fire Science and Engineering*, vol. 21, no. 1, pp. 17–23, 1971. doi:10.11196/kasai.21.17.

[20] T. Jin, "Visibility and human behavior in fire smoke," *SFPE Handbook of Fire Protection Engineering*, vol. 4, pp. 2–54, 2008.

[21] T. Yamada and Y. Akizuki, "Visibility and human behavior in fire smoke," in *SFPE Handbook of Fire Protection Engineering* (M. Hurley, D. Gottuk, J. Hall, K. Harada, E. Kuligowski, M. Puchovsky, J. Torero, J. Watts, and C. Wieczorek, eds.), pp. 2181–2206, New York, NY: Springer New York, 2016. doi:10.1007/978-1-4939-2565-0_61.

[22] X. Hu, Z. Wang, F. Jia, E. Galea, and M. Patel, "Simulating smoke transport in large scale enclosure fires using a multi-particle-size model," *Fire Safety Science*, vol. 10, pp. 445–458, 2011. doi:10.3801/IAFSS.FSS.10-445.

[23] Z. Wang, F. Jia, E. Galea, and J.-H. Choi, "A forensic analysis of a fatal fire in an indoor shooting range using coupled fire and evacuation modelling tools," *Fire Safety Journal*, vol. 91, pp. 892 – 900, 2017. doi:10.1016/j.firesaf.2017.03.029.

[24] K. Kang, "A smoke model and its application for smoke management in an underground mass transit station," *Fire Safety Journal*, vol. 42, no. 3, pp. 218 – 231, 2007. doi:10.1016/j.firesaf.2006.10.003.

[25] K. Kang and H. MacDonald, "Modeling smoke visibility in cfd," *Fire Safety Science*, vol. 8, pp. 1265–1276, 2005. doi:10.3801/IAFSS.FSS.8-1265.

[26] E. Kuligowski, "Computer evacuation models for buildings," in *SFPE Handbook of Fire Protection Engineering* (M. Hurley, D. Gottuk, J. Hall, K. Harada, E. Kuligowski,

M. Puchovsky, J. Torero, J. Watts, and C. Wieczorek, eds.), pp. 2152–2180, New York, NY: Springer New York, 2016. doi:10.1007/978-1-4939-2565-0_60.

[27] J. Batchelor, "Games industry generated $ 108.4bn in revenues in 2017." `https://www.gamesindustry.biz/articles/2018-01-31-games-industry-generated-usd108-4bn-in-revenues-in-2017`, January 2018. Accessed: August 11, 2018.

[28] K. Stuart, "Photorealism - the future of video game visuals — the guardian." `https://www.theguardian.com/technology/2015/feb/12/future-of-video-gaming-visuals-nvidia-rendering`, February 2015. Accessed: August 28, 2018.

[29] E. Kuligowski and B. Hoskins, *Recommendations for Elevator Messaging Strategies*. NIST, January 2012.

[30] G. Proulx and R. Fahy, "Human behavior and evacuation movement in smoke.," *ASHRAE Transactions*, vol. 114, no. 2, 2008.

[31] W. Grosshandler, N. Bryner, D. Madrzykowski, and K. Kuntz, "Report of the technical investigation of the station nightclub fire," *NIST NCSTAR*, 2005.

[32] R. Fahy and G. Proulx, *Analysis of published accounts of the world trade center evacuation*. US Department of Commerce, Technology Administration, National Institute of Standards and Technology Washington, DC, 2005.

[33] M. Kinsey, *Vertical Transport Evacuation Modelling*. PhD thesis, University of Greenwich, September 2011.

[34] History.com Staff, "Hurricane katrina." `http://www.history.com/topics/hurricane-katrina`, 2009. Accessed: August 06, 2018.

[35] A. Kolbe, R. Hutson, H. Shannon, E. Trzcinski, B. Miles, N. Levitz, M. Puccio, L. James, J. Noel, and R. Muggah, "Mortality, crime and access to basic needs before and after the haiti earthquake: a random survey of port-au-prince households," *Medicine, Conflict and Survival*, vol. 26, no. 4, pp. 281–297, 2010. doi:10.1080/13623699.2010.535279.

[36] J. Telford and J. Cosgrave, "Joint evaluation of the international response to the indian ocean tsunami: Synthesis report." http://www.tsunami-evaluation.org/NR/rdonlyres/2E8A3262-0320-4656-BC81-EE0B46B54CAA/0/SynthRep.pdf, July 2006. Accessed: August 06, 2018.

[37] J. Kim and C. Yoo, "More than 300 people missing after south korea ferry sinks - coastguard." http://uk.reuters.com/article/2014/04/16/uk-korea-ship-idUKBREA3F02420140416, 2014. Accessed: 20 April 2014.

[38] Channel4, "Terror at sea: The sinking of the concordia." http://www.channel4.com/programmes/terror-at-sea-the-sinking-of-the-concordia, 2012. Accessed: 31 January 2012.

[39] J. Walker, "Ten years of cruise ship fires - has the cruise industry learned anything?." http://www.cruiselawnews.com/2010/03/articles/fires-1/ten-years-of-cruise-ship-fires-has-the-cruise-industry-learned-anything/, 2010. Accessed: 13 May 2015.

[40] W. Bennett, "Ferry that hit rock may have taken short cut." http://www.independent.co.uk/news/ferry-that-hit-rock-may-have-taken-short-cut-1616086.html, 1995. Accessed: 13 May 2015.

[41] The Join Accident Investigation Commision of Estonia, Finland and Sweden, "Final report on the capsizing on 28 september 1994 in the baltic sea of the ro-ro passenger vessel mv estonia." http://web.archive.org/web/20061129205021/http://www.onnettomuustutkinta.fi/estonia/, December 1997. Accessed: 16 May 2015.

[42] UK Department of Transportation and The Stationery Office, *MV Herald of Free Enterprise Report of Court NO. 8074, Formal Investigation.* Stationery Office Books, 1987.

[43] R. Ballard and R. Archbold, *The Discovery of the Titanics.* Madison publications, 1987.

[44] S. Deere, *Ship Design with the Human Factor: Evacuation and Normal Operations Modelling in the Ship Design Process.* PhD thesis, University of Greenwich, June 2012.

[45] International Maritime Organization, "Solas, marpol amendments entered into force 1 january 2014." http://www.imo.org/MediaCentre/PressBriefings/Pages/01-amendments.aspx, 2014. Accessed: 16 May 2015.

[46] International Maritime Organization, *SOLAS Amendments 2010 and 2011.* International Maritime Organization, January 2012.

[47] The Royal Institution of Naval Architects, *Ship Recoverability: How Important is the Human Factors Input?*, Pacific 2012, International Maritime Conference, January 2012.

[48] M. Togher, *The Evaluation Of The 90 Second Certification Test And The Analysis Of Human Behaviour Within Aircraft Evacuation Modelling.* PhD thesis, University of Greenwich, 2014.

[49] R. Luscombe, "Four dead and 63 injured in new york passenger train derailment." https://www.theguardian.com/world/2013/dec/01/metro-north-passenger-train-derailed-bronx, December 2013. Accessed: 17 August 2018.

[50] Rail Accident Investigation Branch, "Rail accident investigation branch report derailment at grayrigg on 23 february 2007." https://www.gov.uk/raib-reports/derailment-at-grayrigg, 2008. Accessed: 17 May 2015.

[51] The Rt Hon Lord Cullen PC, *The Ladbroke Grove Rail Inquiry.* Her Majesty's Stationery Office, 2001.

[52] NTSB, "Derailment of amtrack auto train p052-18 on csxt railroad near crescent city, florida, on april 18, 2002." http://www.ntsb.gov/news/events/Pages/Derailment_of_Amtrak_Auto_Train_P052-18_on_the_CSXT_Railroad_near_Crescent_City_Florida_on_April_18_2002.aspx, August 2003. Accessed: 17 May 2015.

[53] NTSB, "Collision of two washington metropolitan area transit authority metrorail trains near fort totten station." http://www.ntsb.gov/investigations/AccidentReports/Reports/RAR1002.pdf, July 2010. Accessed: 17 May 2015.

[54] NTSB, "Board meeting : Collision of burlington northern santa fe freight train with metrolink passenger train at placentia, california, april 23, 2002." http://www.ntsb.gov/news/events/Pages/Collision_of_Burlington_Northern_

Santa_Fe_Freight_Train_with_Metrolink_Passenger_Train_at_Placentia_California_April_23_20.aspx, October 2003. Accessed: 17 May 2015.

[55] NTSB, "Uncontrolled movement, collision, and passenger fatality on the angels flight railway in los angeles, california, february 1, 2001." http://www.ntsb.gov/news/events/Pages/Uncontrolled_Movement_Collision_and_Passenger_Fatality_on_the_Angels_Flight_Railway_in_Los_Angeles_California_on_February.aspx, August 2003. Accessed: 17 May 2015.

[56] NTSB, "Collision of national railroad passenger corporation (amtrak) train 59 with a loaded truck-semitrailer combination at a highway/rail grade crossing." http://www.ntsb.gov/investigations/AccidentReports/Pages/RAR0201.aspx, February 2002. Accessed: 17 May 2015.

[57] NTSB, "Railroad accident brief, accident no. dca-01-mr-003." http://www.ntsb.gov/investigations/AccidentReports/Reports/RAB0201.pdf, March 2002. Accessed: 17 May 2015.

[58] NTSB, "Derailment of amtrak train no. 58, city of new orleans." http://www.ntsb.gov/investigations/AccidentReports/Pages/RAR0502.aspx, July 2005. Accessed: 17 May 2015.

[59] EU Publications Office, "AUGGMED - Automated Serious Game Scenario Generator for Mixed Reality Training." https://cordis.europa.eu/project/rcn/194875_en.html. Accessed: 26 June 2018.

[60] BMT Group, "AUGGMED - Homepage." http://auggmed-project.eu/about.aspx. Accessed: 26 June 2018.

[61] D. Boswell and S. Gwynne, "Air, fire and ice: fire & security challenges unique to airports," *Fire and Security Today*, pp. 30–37, January 2007.

[62] S. Gwynne and E. Kuligowski, "Simulating a building as a people movement system," *Journal of Fire Sciences*, vol. 27, pp. 343–368, July 2009. doi:10.1177/0734904109102387.

[63] J. Fruin, "Pedestrian planning and design," tech. rep., Metropolitan Association of Urban Designers and Environmental Planners, 1971.

[64] D. Tong and D. Canter, "The decision to evacuate: a study of the motivations which contribute to evacuation in the event of fire," *Fire Safety Journal*, vol. 9, no. 3, pp. 257 – 265, 1985. doi:10.1016/0379-7112(85)90036-0.

[65] M. Chu, *A Computational Framework Incorporating Human and Social Behaviors for Occupant-centric Egress Simulation*. PhD thesis, Stanford University, January 2015.

[66] G. L. Bon, *The Crowd, A Study of the Popular Mind*. McMaster University Archive for the History of Economic Thought, 1896.

[67] C. McPhail, "Blumer's theory of collective behavior: the development of a non-symbolic interaction explanation," *The Sociological Quarterly*, vol. 30, no. 3, pp. 401–423, 1989. doi:10.1111/j.1533-8525.1989.tb01528.x.

[68] B. Aguirre, M. Torres, K. Gill, and H. Lawrence, "Normative collective behavior in the station building fire," *Social science quarterly*, vol. 92, no. 1, pp. 100–118, 2011. doi:10.1111/j.1540-6237.2011.00759.x.

[69] J. Sime, "Movement toward the familiar: Person and place affiliation in a fire entrapment setting," *Environment and Behavior*, vol. 17, no. 6, pp. 697–724, 1985. doi:10.1177/0013916585176003.

[70] H. Tafjel and J. Turner, "The social identity theory of intergroup behavior," *Psychology of intergroup relations*, pp. 7–24, 1986.

[71] E. Kuligowski, "Human behavior in fire," in *SFPE Handbook of Fire Protection Engineering* (M. Hurley, D. Gottuk, J. Hall, K. Harada, E. Kuligowski, M. Puchovsky, J. Torero, J. Watts, and C. Wieczorek, eds.), pp. 2070–2114, New York, NY: Springer New York, 2016. doi:10.1007/978-1-4939-2565-0_58.

[72] R. Turner and L. Killian, "Collective behavior (3e éd.)," *Englewood Cliffs, NJ*, 1987.

[73] W. Feinberg and N. Johnson, "Primary group size and fatality risk in a fire disaster," *Human behavior in fire: Understanding human behavior for better fire safety design*, pp. 11–22, 2001.

[74] T. Powell, "London Bridge incident: Terrifying footage shows armed police storm bar shouting 'get down!'." `https://www.standard.co.uk/news/london/london-bridge-incident-terrifying-footage-shows-armed-police-storm-bar-shouting-get-down-a3556351.html`, June 2017. Accessed: 15 April 2018.

[75] BBC News, "London attack: Seven killed in vehicle and stabbing incidents." `https://www.bbc.co.uk/news/uk-40146916`, June 2017. Accessed: 15 April 2018.

[76] M. Chu and K. Law, "Incorporating individual behavior, knowledge, and roles in simulating evacuation," *Fire Technology*, Jul 2018. doi:10.1007/s10694-018-0747-6.

[77] I. Donald and D. Canter, "Intentionality and fatality during the king's cross underground fire," *European Journal of Social Psychology*, vol. 22, pp. 203–218, May/June 1992. doi:10.1002/ejsp.2420220302.

[78] P. Kramer, *Listening to Prozac*. New York: Viking Press, 1993.

[79] S. Gwynne, E. Galea, P. Lawrence, and L. Filippidis, "Modelling occupant interaction with fire conditions using the buildingexodus evacuation model," *Fire Safety Journal*, vol. 36, no. 4, pp. 327 – 357, 2001. doi:10.1016/S0379-7112(00)00060-6.

[80] D. Purser and J. McAllister, "Assessment of hazards to occupants from smoke, toxic gases, and heat," in *SFPE Handbook of Fire Protection Engineering* (M. Hurley, D. Gottuk, J. Hall, K. Harada, E. Kuligowski, M. Puchovsky, J. Torero, J. Watts, and C. Wieczorek, eds.), pp. 2308–2428, New York, NY: Springer New York, 2016. doi:10.1007/978-1-4939-2565-0_63.

[81] Z. Wang, F. Jia, and E. Galea, "Predicting toxic gas concentrations resulting from enclosure fires using local equivalence ratio concept linked to fire field models," *Fire and Materials*, vol. 31, pp. 27–51, May 2006. doi:10.1002/fam.924.

[82] E. Galea, Z. Wang, A. Veeraswamy, F. Jia, P. Lawrence, and J. Ewer, "Coupled fire/evacuation analysis of the station nightclub fire," in *Proceedings of 9th IAFSS Symposium Karlsruhe*, vol. 9, (Germany), pp. 465–476, 2008. doi:10.3801/IAFSS.FSS.9-465.

[83] K. McGrattan, S. Hostikka, R. McDermott, J. Floyd, and M. Vanella, "Fire dynamics simulator (version 4) user's guide." https://github.com/firemodels/fds/releases/download/FDS6.7.0/FDS_User_Guide.pdf, June 2018. Accessed: August 16, 2018. doi:10.6028/NIST.SP.1019.

[84] T. Korhonen and S. Hostikka, "FDS+Evac – Technical Reference + User's Guide." http://virtual.vtt.fi/virtual/proj6/fdsevac/documents/fds5evac110_usermanual_6aug2007.pdf, August 2007. Accessed: August 16, 2018.

[85] J. Bryan, "A selected historical review of human behavior in fire," *Fire Protection Engineering*, vol. 16, pp. 4–10, October 2002.

[86] H. Horton, H. Dixon, P. Sawer, H. Yorke, H. Bodkin, and E. Steafel, "Grenfell tower fire: Full details of all the missing people." https://www.telegraph.co.uk/news/0/grenfell-tower-fire-fatalities-missing-people/, June 2017. Accessed: August 16, 2018.

[87] T. Jin and T. Yamada, "Irritating effects of fire smoke on visibility," *Fire Science and Technology*, vol. 5, no. 1, pp. 79–90, 1985. doi:10.3210/fst.5.79.

[88] M. Conway, "Experimenting with smoke visibility and fire modelling: Part 1 - human viewing and light extinction," *International Fire Professional*, vol. 24, pp. 32–37, May 2018.

[89] G. Vigne and W. Węgrzyński, "Experimental and numerical evaluation of the influence of the soot yield on the visibility in smoke in cfd analysis," *Fire Safety Journal*, vol. 91, pp. 389 – 398, July 2017. doi:10.1016/j.firesaf.2017.03.053.

[90] P. Rubini and Q. Zhang, "Simulation of visibility in smoke laden environments," in *5th International Seminar on Fire and Explosion Hazards*, pp. 23–27, April 2007.

[91] P. Rubini, Q. Zhang, and J. Moss, "Simulation of visibility in smoke laden environments," in *5th International Seminar on Fire and Explosion Hazards*, pp. 23–27, 2007.

[92] E. Weingartner, H. Saathoff, M. Schnaiter, N. Streit, B. Bitnar, and U. Baltensperger, "Absorption of light by soot particles: determination of the absorption coefficient by

means of aethalometers," *Journal of Aerosol Science*, vol. 34, no. 10, pp. 1445 – 1463, 2003. doi:10.1016/S0021-8502(03)00359-8.

[93] J. Kozak, P. Hancock, E. Arthur, and S. Chrysler, "Transfer of training from virtual reality," *Ergonomics*, vol. 36, no. 7, pp. 777–784, 1993. doi:10.1080/00140139308967941.

[94] Finnish Ground Forces, "The mechanised exercise of the finnish army (arrow 17)." `https://maavoimat.fi/en/the-mechanised-exercise-of-the-finnish-army-arrow-17-`, May 2017. Accessed: 1 Aug 2018.

[95] The Finnish Defence Forces, "Arrow17 – iskukyky vahvistuu — enhancing effective engagement capability." `https://www.youtube.com/watch?v=EXUSs1Qz4zs`, May 2017. Accessed: 1 Aug 2018.

[96] The Finnish Defence Forces, "Arrow17 — monipuolinen ja onnistunut harjoitus — a diverse and successful exercise." `https://www.youtube.com/watch?v=yQKe82Fi0d4`, May 2017. Accessed: 1 Aug 2018.

[97] K. Meesters, "Towards using serious games for realistic evaluation of disaster management it tools.," in *AIM SG*, pp. 38–48, 2014.

[98] R. Lovreglio, V. Gonzalez, R. Amor, M. Spearpoint, J. Thomas, M. Trotter, and R. Sacks, "The need for enhancing earthquake evacuee safety by using virtual reality serious games," in *Lean and Computing in Construction Congress (LC3)*, vol. 1, pp. 381–389, July 2017. doi:10.24928/JC3-2017/0058.

[99] S. Smith and E. Ericson, "Using immersive game-based virtual reality to teach fire-safety skills to children," *Virtual Reality*, vol. 13, pp. 87–99, Jun 2009. doi:10.1007/s10055-009-0113-6.

[100] A. Sacfung, K. Sookhanaphibarn, and W. Choensawat, "Serious game for fire safety evacuation plan," in *Advanced Materials Research*, vol. 931, pp. 583–587, Trans Tech Publ, 2014. doi:10.4028/www.scientific.net/AMR.931-932.583.

[101] Z. Feng, V. A. González, R. Amor, R. Lovreglio, and G. Cabrera-Guerrero, "Immersive virtual reality serious games for evacuation training and research: A sys-

tematic literature review," *Computers & Education*, vol. 127, pp. 252 – 266, 2018. doi:10.1016/j.compedu.2018.09.002.

[102] I. Heldal, H. Wijkmark, and L. Pareto, "Simulation and serious games for firefighter training: Challenges for effective use," *NOKOBIT*, vol. 24, p. 12, 2016.

[103] J. Hormann, "Virtual reality," *FBI Law Enforcement Bulletin*, vol. 64, pp. 7–12, July 1995.

[104] T. Bos, "The impact of using virtual reality technology to train for law enforcement critical incidents." `http://lib.post.ca.gov/lib-documents/cc/36-Bos-j.pdf`, 2004. Accessed: August 07, 2018.

[105] D. Waller, E. Hunt, and D. Knapp, "The transfer of spatial knowledge in virtual environment training," *Presence: Teleoperators and Virtual Environments*, vol. 7, no. 2, pp. 129–143, 1998. doi:10.1162/105474698565631.

[106] R. Darken and W. Banker, "Navigating in natural environments: a virtual environment training transfer study," in *Proceedings. IEEE 1998 Virtual Reality Annual International Symposium (Cat. No.98CB36180)*, pp. 12–19, March 1998. doi:10.1109/VRAIS.1998.658417.

[107] S. Sharma, P. Devreaux, D. Scribner, J. Grynovicki, and P. Grazaitis, "Megacity: A collaborative virtual reality environment for emergency response, training, and decision making," *Electronic Imaging*, vol. 2017, no. 1, pp. 70–77, 2017. doi:10.2352/ISSN.2470-1173.2017.1.VDA-390.

[108] J. Keating, "The myth of panic," *Fire Journal*, vol. 76, no. 2, p. 57–61, 1982.

[109] S. Sharma, H. Singh, and A. Prakash, "Multi-agent modeling and simulation of human behavior in aircraft evacuations," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 44, pp. 1477–1488, Oct 2008. doi:10.1109/TAES.2008.4667723.

[110] L. Chittaro and R. Sioni, "Serious games for emergency preparedness: Evaluation of an interactive vs. a non-interactive simulation of a terror attack," *Computers in Human Behavior*, vol. 50, pp. 508 – 519, 2015. doi:10.1016/j.chb.2015.03.074.

[111] W. Viant, J. Purdy, and J. Wood, "Serious games for fire and rescue training," in *2016 8th Computer Science and Electronic Engineering (CEEC)*, pp. 136–139, Sep. 2016. doi:10.1109/CEEC.2016.7835902.

[112] G. Ha, H. Lee, S. Lee, J. Cha, and S. Kim, "A vr serious game for fire evacuation drill with synchronized tele-collaboration among users," in *Proceedings of the 22Nd ACM Conference on Virtual Reality Software and Technology*, VRST '16, (New York, NY, USA), pp. 301–302, ACM, 2016. doi:10.1145/2993369.2996306.

[113] P. Backlund, H. Engström, M. Gustavsson, M. Johannesson, M. Lebram, and E. Sjörs, "Sidh: A game-based architecture for a training simulator," *International Journal of Computer Games Technology*, vol. 2009, no. 472672, pp. 1–9, 2009. doi:10.1155/2009/472672.

[114] E. Ronchi, D. Nilsson, S. Kojić, J. Eriksson, R. Lovreglio, H. Modig, and A. Lindgren Walter, "A virtual reality experiment on flashing lights at emergency exit portals for road tunnel evacuation," *Fire Technology*, vol. 52, no. 3, pp. 623–647, 2016. doi:10.1007/s10694-015-0462-5.

[115] E. Ronchi and D. Nilsson, *A Virtual Reality experiment on the design of flashing lights at emergency exit portals for road tunnel evacuations*, vol. 3180. Department of Fire Safety Engineering and Systems Safety, Lund University, 2015.

[116] K. Andrée, D. Nilsson, and J. Eriksson, "Evacuation experiments in a virtual reality high-rise building: Exit choice and waiting time for evacuation elevators," *Fire and Materials*, vol. 40, pp. 554–567, 6 2016. doi:10.1002/fam.2310.

[117] M. Kinateder, E. Ronchi, D. Gromer, M. Müller, M. Jost, M. Nehfischer, A. Mühlberger, and P. Pauli, "Social influence on route choice in a virtual reality tunnel fire," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 26, pp. 116 – 125, 2014. doi:https://doi.org/10.1016/j.trf.2014.06.003.

[118] H. Gustavsson, H. Engström, and M. Gustavsson, "A multi-sampling approachfor smoke behaviour in real-time graphics," in *SIGRAD 2006. The Annual SIGRAD Conference; Special Theme: Computer Games*, no. 19, p. 17–21, Linköping University Electronic Press; Linköpings universitet, 2006.

[119] S. Arias, S. L. Mendola, J. Wahlqvist, O. Rios, D. Nilsson, and E. Ronchi, "Virtual reality evacuation experiments on way-finding systems for the future circular collider," *Fire Technology*, May 2019. doi:10.1007/s10694-019-00868-y.

[120] S. Arias, E. Ronchi, J. Wahlqvist, J. Eriksson, and D. Nilsson, *ForensicVR: Investigating human behaviour in fire with Virtual Reality*, vol. Report 3218 of *LUTVDG/TVBB*. Lund university, 11 2018.

[121] G. Cosma, E. Ronchi, and D. Nilsson, "Way-finding lighting systems for rail tunnel evacuation: A virtual reality experiment with oculus rift," *Journal of Transportation Safety and Security*, vol. 8, no. No. S1, pp. 101–117, 2016. doi:10.1080/19439962.2015.1046621.

[122] E. Ronchi, M. Kinateder, M. Müller, M. Jost, M. Nehfischer, P. Pauli, and A. Mühlberger, "Evacuation travel paths in virtual reality experiments for tunnel safety analysis," *Fire Safety Journal*, vol. 71, pp. 257–267, 2015. doi:10.1016/j.firesaf.2014.11.005.

[123] K. McGrattan, S. Hostikka, J. Floyd, H. Baum, and R. Rehm, "Fire dynamics simulator (version 5): Technical reference guide." `https://www.vtt.fi/inf/julkaisut/muut/2007/NIST_SP_1018_5.pdf`, October 2007. Accessed: July 31, 2019.

[124] K. McGrattan, B. Klein, S. Hostikka, and J. Floyd, "Fire dynamics simulator (version 5) user's guide." `https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication1019-5.pdf`, October 2007. Accessed: July 31, 2019. doi:10.6028/nist.sp.1019-5.

[125] D. McNeill, *Hand and mind: What gestures reveal about thought*. University of Chicago press, 1992.

[126] J. Cassell, D. McNeill, and K.-E. McCullough, "Speech-gesture mismatches: Evidence for one underlying representation of linguistic and nonlinguistic information," *Pragmatics & cognition*, vol. 7, no. 1, pp. 1–34, 1999. doi:10.1075/pc.7.1.03cas.

[127] K. Stuart, "Uk gamers: more women play games than men, report finds." `https://www.theguardian.com/technology/2014/sep/17/women-video-games-iab`, September 2014. Accessed: August 07, 2018.

[128] N. Sandstrom, J. Kaufman, and S. Huettel, "Males and females use different distal cues in a virtual environment navigation task1published on the world wide web on 27 january 1998.1," *Cognitive Brain Research*, vol. 6, no. 4, pp. 351 – 360, 1998. doi:10.1016/S0926-6410(98)00002-0.

[129] M. Pan and G. Niemeyer, "Catching a real ball in virtual reality." `https://www.youtube.com/watch?v=Qxu_y8ABajQ`, March 2017. Accessed: 16 August 2018.

[130] M. J. Seitz, *Simulating pedestrian dynamics: Towards natural locomotion and psychological decision making*. PhD thesis, Technische Universität München, Munich, Germany, 2016.

[131] A. K. Wagoum, M. Chraibi, J. Zhang, and G. Lämmel, "Jupedsim: an open framework for simulating and analyzing the dynamics of pedestrians," in *3rd Conference of Transportation Research Group of India*, 2015.

[132] S. Curtis, A. Best, and D. Manocha, "Menge: A modular framework for simulating crowd movement," *Collective Dynamics*, vol. 1, pp. 1–40, 2016. doi:10.17815/CD.2016.1.

[133] C. M. Pancake, "Multithreaded languages for scientific and technical computing," *Proceedings of the IEEE*, vol. 81, pp. 288–304, Feb 1993. doi:10.1109/5.214551.

[134] S. S. Vladimirovich, F. T. Anatolievna, and B. A. Valerievich, "Scheduling multithreaded processes by criterion of minimum of number data exchanges between processor cores," in *2018 7th Mediterranean Conference on Embedded Computing (MECO)*, pp. 1–4, June 2018. doi:10.1109/MECO.2018.8406009.

[135] M. Michael and M. Scott, "Simple, fast, and practical non-blocking and blocking concurrent queue algorithms," in *Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, pp. 267–275, 1996. doi:10.1145/248052.248106.

[136] Khronos Group, "Gdc 2015 - more on vulkan and spir-v: The future of high-performance graphics." `https://www.khronos.org/assets/uploads/developers/library/2015-gdc/Khronos-Vulkan-GDC_Mar15.pdf`, 2015. Accessed: 3 Jun 2018.

[137] M. Zook, "Old hierarchies or new networks of centrality?: The global geography of the internet content market," *American Behavioral Scientist*, vol. 44, no. 10, pp. 1679–1696, 2001. doi:10.1177/00027640121958113.

[138] R. Lee and F. Closs, "Method and means for demand accessing and broadcast transmission among ports in a distributed star network," August 1982. US Patent 4,347,498.

[139] F. Sanglard, "Quake 3 source code review: Network model." `http://fabiensanglard.net/quake3/network.php`, June 2012. Accessed: 26 June 2018.

[140] F. Sanglard, "Quake engine code review: Network." `http://fabiensanglard.net/quakeSource/quakeSourceNetWork.php`, March 2009. Accessed: 26 June 2018.

[141] F. Dunn, "Denial-of-service mitigation," in *Game Developers Conference 2018*, March 2018.

[142] Valve Software, "Gamenetworkingsockets - github." `https://github.com/ValveSoftware/GameNetworkingSockets`, June 2018. Accessed: 27 July 2018.

[143] D. Kritler, C. Brokke, and P. Tourreau, "Gigabit ethernet drives evolution of multimode fiber-optic specifications," *Lazer Focus World*, vol. 45(11), pp. 58–61, November 2009.

[144] D. Law, D. Dove, J. D'Ambrosia, M. Hajduczenia, M. Laubach, and S. Carlson, "Evolution of ethernet standards in the ieee 802.3 working group," *IEEE Communications Magazine*, vol. 51(8), pp. 88–96, August 2013. doi:10.1109/MCOM.2013.6576344.

[145] Network Working Group of the IETF, "Rfc 894: A standard for the transmission of ip datagrams over ethernet networks." `http://tools.ietf.org/html/rfc894`, April 1984. Accessed: 17 May 2015.

[146] H. Adiga, "Writing endian-independent code in c." `https://www.ibm.com/developerworks/aix/library/au-endianc/index.html`, April 2007. Accessed: 25 Jun 2018.

[147] id Software, "Quake/winquake/common.c - source code on github." `https://github.com/id-Software/Quake/blob/master/WinQuake/common.c#L1125`, December 1999. Accessed: 25 Jun 2018.

[148] A. Hunt, *Simulating Hospital Evacuation*. PhD thesis, University of Greenwich, August 2015.

[149] E. Galea and J. Perez Galparsoro, "A computer-based simulation model for the prediction of evacuation from mass-transport vehicles," *Fire Safety Journal*, vol. 22, no. 4, pp. 341 – 366, 1994. doi:10.1016/0379-7112(94)90040-X.

[150] Microsoft, "Direct3d 9 to direct3d 10 considerations (direct3d 10)." `https://docs.microsoft.com/en-us/windows/desktop/direct3d10/d3d10-graphics-programming-guide-d3d9-to-d3d10-considerations`, May 2018. Accessed: August 26, 2018.

[151] Khronos Wikipedia, "History of opengl." `https://www.khronos.org/opengl/wiki/History_of_OpenGL`. Accessed: August 26, 2018.

[152] Khronos Group, "Display lists and vertex arrays." `https://www.opengl.org/archives/resources/faq/technical/displaylist.htm`. Accessed: August 26, 2018.

[153] Valve Linux Team, "Faster zombies!." `http://blogs.valvesoftware.com/linux/faster-zombies/`, August 2012. Accessed: 29 June 2018.

[154] D. Cooney, "The application of virtual reality environments to evacuation modelling," Master's thesis, University of Greenwich, September 2000.

[155] P. Barta, B. Kovács, L. Szécsi, and L. Szirmay-kalos, "Order independent transparency with per-pixel linked lists," *Budapest University of Technology and Economics*, 2011.

[156] L. Bavoil and K. Myers, "Order independent transparency with dual depth peeling," *NVIDIA OpenGL SDK*, pp. 1–12, 2008.

[157] C. Everitt, "Interactive order-independent transparency," *White paper, nVIDIA*, vol. 2, no. 6, p. 7, 2001.

[158] G. Sellers, J. R. S. Wright, and N. Haemel, *OpenGL SuperBible: Comprehensive Tutorial and Reference*. Addison Wesley, seventh ed., July 2015.

[159] Khronos Group, "glblendequation - opengl 4 reference pages." `https://www.khronos.org/registry/OpenGL-Refpages/gl4/html/glBlendEquation.xhtml`. Accessed: 8 July 2018.

[160] T. Porter and T. Duff, "Compositing digital images," *SIGGRAPH Comput. Graph.*, vol. 18, pp. 253–259, Jan. 1984. doi:10.1145/964965.808606.

[161] Khronos Group, "glblendfunc - opengl 4 reference pages." `https://www.khronos.org/registry/OpenGL-Refpages/gl4/html/glBlendFunc.xhtml`. Accessed: 8 July 2018.

[162] T. Umenhoffer, L. Szirmay-Kalos, and G. Szijártó, "Spherical billboards and their application to rendering explosions," in *Proceedings of Graphics Interface 2006*, GI '06, (Toronto, Ont., Canada), pp. 57–63, Canadian Information Processing Society, 2006.

[163] M. Waschbüsch, S. Würmlin, and M. Gross, "3d video billboard clouds," *Computer Graphics Forum*, vol. 26, no. 3, pp. 561–569, 2007. doi:10.1111/j.1467-8659.2007.01079.x.

[164] M. J. Harris, *Real-Time Cloud Simulation and Rendering*. PhD thesis, University of North Carolina at Chapel Hill, 2003.

[165] N. Wang, "Realistic and fast cloud rendering," *Journal of Graphics Tools*, vol. 9, no. 3, pp. 21–40, 2004. doi:10.1080/10867651.2004.10504895.

[166] M. J. Harris and A. Lastra, "Real-time cloud rendering," *Computer Graphics Forum*, vol. 20, no. 3, pp. 76–85, 2001. doi:10.1111/1467-8659.00500.

[167] C. J. Bass and E. F. Anderson, "Smoke and shadows: Rendering and light interaction of smoke in real-time rendered virtual environments," in *2013 5th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*, pp. 1–4, Sep. 2013. doi:10.1109/VS-GAMES.2013.6624242.

[168] P. Decaudin and F. Neyret, "Volumetric billboards," *Computer Graphics Forum*, vol. 28, no. 8, pp. 2079–2089, 2009.

[169] R. Larsson, "Interactive real-time smoke rendering," Master's thesis, Chalmers University Of Technology, 2010.

[170] H.-S. Liao, T.-C. Ho, J.-H. Chuang, and C.-C. Lin, "Fast rendering of dynamic clouds," *Computers & Graphics*, vol. 29, no. 1, pp. 29 – 40, 2005. doi:10.1016/j.cag.2004.11.005.

[171] T. Kim, N. Thürey, D. James, and M. Gross, "Wavelet turbulence for fluid simulation," *ACM Trans. Graph.*, vol. 27, pp. 50:1–50:6, Aug. 2008. doi:10.1145/1360612.1360649.

[172] J. Love, "Technical artist bootcamp: The vfx of diablo," in *Game Developers Conference 2013*, March 2013.

[173] I. McEwan, "Array and textureless GLSL 2d/3d/4d simplex noise functions.." `https://github.com/ashima/webgl-noise/wiki`. Accessed: 28 May 2018.

[174] I. McEwan, D. Sheets, M. Richardson, and S. Gustavson, "Efficient computational noise in GLSL," *Journal of Graphics Tools*, vol. 16, no. 2, pp. 85–94, 2012. doi:10.1080/2151237X.2012.649621.

[175] M. McGuire and A. Fein, "Real-time rendering of cartoon smoke and clouds," in *Proceedings of the 4th International Symposium on Non-photorealistic Animation and Rendering*, NPAR '06, (New York, NY, USA), pp. 21–26, ACM, 2006. doi:10.1145/1124728.1124733.

[176] International Standards Organisation, *Fire safety engineering - Procedures and requirements for verification and validation of calculation methods - Part 1: General.* ISO 16730-1:2015.

[177] E. Ronchi, E. Kuligowski, P. Reneke, R. Peacock, and D. Nilsson, *The process of verification and validation of building fire evacuation models.* US Department of Commerce, National Institute of Standards and Technology, 11 2013.

[178] G. Mulholland, "Smoke production and properties," vol. 2, pp. 217–227, National Fire Protection Association, 1995.

[179] J. Klote and J. Milke, *Principles of smoke management.* American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) and Society of Fire Protection Engineers (SFPE), 2002.

[180] EU Publications Office, "Nacre - new aircraft concepts research." `https://cordis.europa.eu/project/rcn/75773_en.html`. Accessed: 19 July 2018.

[181] University of Greenwich, "Vela: Very effecient large aircraft." `http://fseg.gre.ac.uk/fire/VELA.html`. Accessed: 29 August 2018.

[182] E. Galea, L. Filippidis, Z. Wang, P. Lawrence, and J. Ewer, "Evacuation analysis of 1000+ seat blended wing body aircraft configurations: Computer simulations and full-scale evacuation experiment," in *Pedestrian and Evacuation Dynamics* (R. Peacock, E. Kuligowski, and J. Averill, eds.), (Boston, MA), pp. 151–161, Springer US, 2011. doi:10.1007/978-1-4419-9725-8_14.

[183] E. Galea, L. Filippidis, Z. Wang, and J. Ewer, "Fire and evacuation analysis in bwb aircraft configurations: computer simulations and large-scale evacuation experiment," *The Aeronautical Journal (1968)*, vol. 114, no. 1154, p. 271–277, 2010. doi:10.1017/S0001924000003717.

[184] Fire Safety Engineering Group, "Blended wing body aircraft fire and evacuation - youtube." `https://www.youtube.com/watch?v=aRc0Z32q8ms`. Accessed: 29 August 2018.

[185] A. Nabelek, J. Tampas, and S. Burchfield, "Comparison of speech perception in background noise with acceptance of background noise in aided and unaided conditions," *Journal of Speech, Language, and Hearing Research*, vol. 47, no. 5, pp. 1001–1011, 2004. doi:10.1044/1092-4388(2004/074).

[186] M. Deutsch and H. Gerard, "A study of normative and informational social influences upon individual judgment.," *The journal of abnormal and social psychology*, vol. 51, no. 3, pp. 629–636, 1955. doi:10.1037/h0046408.

[187] D. Nilsson and A. Johansson, "Social influence during the initial phase of a fire evacuation—analysis of evacuation experiments in a cinema theatre," *Fire Safety Journal*, vol. 44, no. 1, pp. 71 – 79, 2009. doi:10.1016/j.firesaf.2008.03.008.

[188] B. Latané and J. Darley, *The Unresponsive Bystander : Why Doesn't He Help?* New York, NY, USA: Appleton-Century-Crofts, 1970.

[189] E. Galea, S. Deere, C. Hopkin, and H. Xie, "Evacuation response behaviour of occupants in a large theatre during a live performance," *Fire and Materials*, vol. 41, no. 5, pp. 467–492, 2017. doi:10.1002/fam.2424.

[190] University of Minnesota, "Emergency guidebook for general aviation airports: A guidebook for municipal airport managers." `http://www.airtap.umn.edu/guide/operations/`. Accessed: 14 April 2018.

[191] E. Galea, P. Lawrence, S. Gwynne, L. Filippidis, D. Blackshields, and D. Cooney, "buildingEXODUS v6.3 Theory Manual," May 2017.

[192] K. Fridolf, E. Ronchi, D. Nilsson, and H. Frantzich, "The representation of evacuation movement in smoke-filled underground transportation systems," *Tunnelling and Underground Space Technology*, vol. 90, pp. 28–41, 2019. doi:10.1016/j.tust.2019.04.016.

[193] E. Ronchi, S. Gwynne, and D. A. Purser, "The impact of default settings on evacuation model results: a study of visibility conditions vs occupant walking speeds," in *Advanced research workshop evacuation and human behaviour in emergency situations EVAC11, Santander*, pp. 81–95, 2011.

[194] X. Chen, Z. Shao, and J. Yang, "Comparison between actual and simulated smoke for smoke detection certification in aircraft cargo compartments using the cfd method," *Fire Technology*, Jul 2019. doi:10.1007/s10694-019-00887-9.

[195] E. Galea, P. Lawrence, and L. Filippidis, "Extending the capabilities of the buildingexodus evacuation model to cater for hospital evacuations," in *Proceedings of the Third International Conference on Fire Research and Engineering*, (Chicago, USA), pp. 39–50, October 1999.

[196] S. Gwynne, E. Galea, M. Owen, P. Lawrence, and L. Filippidis, *Validation of the buildingEXODUS Evacuation Model*. CMS Press Paper 98/IM/29, 1998.

[197] S. Gwynne, E. Galea, P. Lawrence, M. Owen, and L. Filippidis, "A systematic comparison of model predictions produced by the buildingexodus evacuation model and the tsukuba pavilion evacuation data," *Journal of Applied Fire Science*, vol. 7, no. 3, pp. 235–266, 1998.

[198] P. L. E.R. Galea, M. Owen, "A demonstration application of the buildingexodus evacuation model for the evacuation of patients from hospitals," March 1997.

[199] H. Weckman, S. Lehtimäki, and S. Männikkö, "Evacuation of a theatre: Exercise vs calculations," *Fire and Materials*, vol. 23, no. 6, pp. 357–361, 1999. doi:10.1002/(SICI)1099-1018(199911/12)23:6<357::AID-FAM711>3.0.CO;2-5.

[200] H. Weckman, S. Lehtimäki, and S. Männikkö, "Evacuation of a theatre: Exercise vs calculations," in *The Proceedings of the First International Symposium on Human Behaviour in Fire* (J. Shields, ed.), pp. 479–488, 1998.

[201] E. Galea, S. Gwynne, M. Owen, P. Lawrence, and L. Filippidis, "A comparison of predictions from the buildingexodus evacuation model with experimental data," in *The Proceedings of the First International Symposium on Human Behaviour in Fire* (J. Shields, ed.), pp. 711–720, 1998.

[202] J. Parke, S. Gwynne, E. Galea, and P. Lawrence, "Validating the buildingexodus evacuation model using data from an unannounced trial evacuation," in *Proceedings of the 2nd International Pedestrian and Evacuation Dynamics Conference* (E. Galea, ed.), (Greenwich, UK), pp. 295–306, CMS Press, 2003.

[203] H. Jiang, S. Gwynne, E. Galea, P. Lawrence, F. Jia, and H. Ingason, "The use of evacuation simulation, fire simulation and experimental data in forensic fire analysis," in *Proceedings of the 2nd International Pedestrian and Evacuation Dynamics Conference* (E. Galea, ed.), (Greenwich, UK), pp. 341–354, CMS Press, 2003.

[204] E. Galea, "A general approach to validating evacuation models with an application to exodus," *Journal of Fire Sciences*, vol. 16, no. 6, pp. 414–436, 1998. doi:10.1177/073490419801600603.

[205] E. Galea, P. Lawrence, M. Owen, and L. Filippidis, "Adapting the buildingexodus evacuation model for hospital specific evacuation scenarios," in *Proceedings of the 8th International Fire Science and Engineering Conference: Interflam'99*, vol. 2, (London, UK), pp. 1247–1252, Interscience Communications Ltd., 1999.

[206] E. Galea, G. Sharp, P. Lawrence, and R. Holden, "Approximating the evacuation of the world trade center north tower using computer simulation," *Journal of Fire Protection Engineering*, vol. 18, no. 2, pp. 85–115, 2008. doi:10.1177/1042391507079343.

[207] D. Land, "GDC 2018 - Getting explicit: How Hard is Vulkan really?." https://www.youtube.com/watch?v=0R23npUCCnw, 2018. Accessed: 14 April 2018.

[208] A. Ladavac, "Advanced graphics techniques tutorial: The elusive frame timing: A case study for smoothness over speed," in *Game Developers Conference*, (Moscone Center, San Francisco, CA, USA), March 2018.

[209] Reallusion, "Types of maps." https://www.reallusion.com/iclone/help/3dxchange4/pro/Types_of_maps.htm. Accessed: August 28, 2018.

[210] N. Ferguson, B. Schneier, and T. Kohno, *Cryptography Engineering: Design Principles and Practical Applications*. Indianapolis, Indiana: Wiley Publishing, Inc., first ed., 2010.

[211] T. Kannan, "Virtual reality graphics for evacuation model (vrexodus)," Master's thesis, University of Greenwich, October 1995.

[212] T. Tan, "Virtual reality for aircraft evacuation model (exodus)," final year project bsc (hons) computing science (software engineering), University of Greenwich, May 1996.

[213] D. Brace, "Object vrexodus," final year project bsc (hons) computing science (software engineering), University of Greenwich, May 1996.

[214] J. Ewer, *An Investigation Into The Feasibility Problems And Benefits Of Re-Engineering A Legacy Procedural CFD Code Into An Event Driven, Object Oriented System That Allows Dynamic User Interaction*. PhD thesis, University of Greenwich, July 2000.

[215] A. Webber, "First person visualisation of smoke visibility from cfd data," final year project beng (hons) software engineering, University of Greenwich, March 2007.

APPENDICES

Appendix A

AIRPORT SMARTFIRE SIMULATIONS

As the IRIS system has the capability of importing a 3D model as the geometry in Wavefront OBJ file format, this section will show what a hand-built geometry would look like. The geometry was provided by the AUGGMED EU-Project (Project ID: 653590) in 2016 as a hand built version of the airport geometry demonstrated in this thesis. The CFD Fire simulation was initially ran for the airport geometry (as shown in this section) but was later abandoned in favour of the aircraft fire in this research as it did not provide as interesting navigational challenges as the aircraft geometry due to the very slow smoke spread in a large area. However, the visibility of the smoke is represented correctly in the screengrabs in this appendix. To demonstrate the effective visibility reduction in a smoke environment, the simulated computer-controlled agents have been made immune to the effects of the fire as well as to not to react or respond to it. As can be seen in the screengrabs below the visibility is severely reduced once the smoke has filled the large room in the airport.



Figure A.1: The airport geometry provided by AUGGMED project.

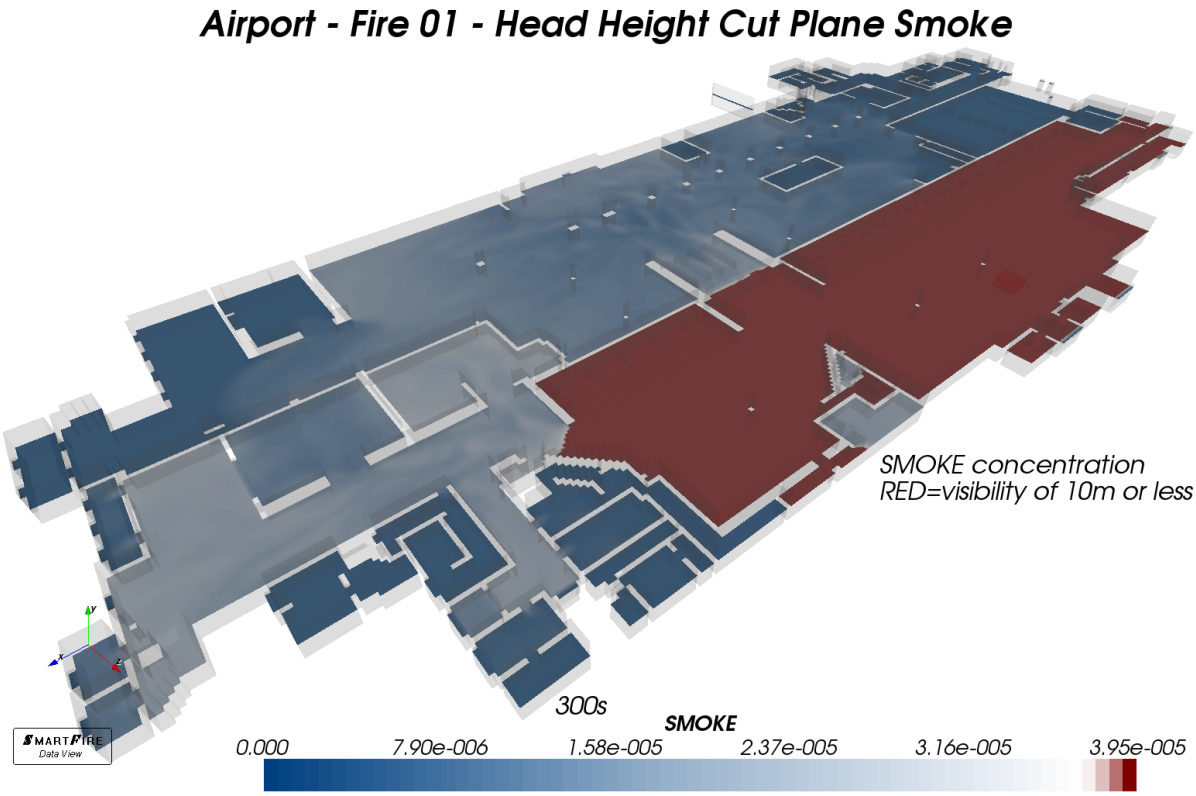


Figure A.2: Airport mass fraction of smoke graph in SMARTFIRE

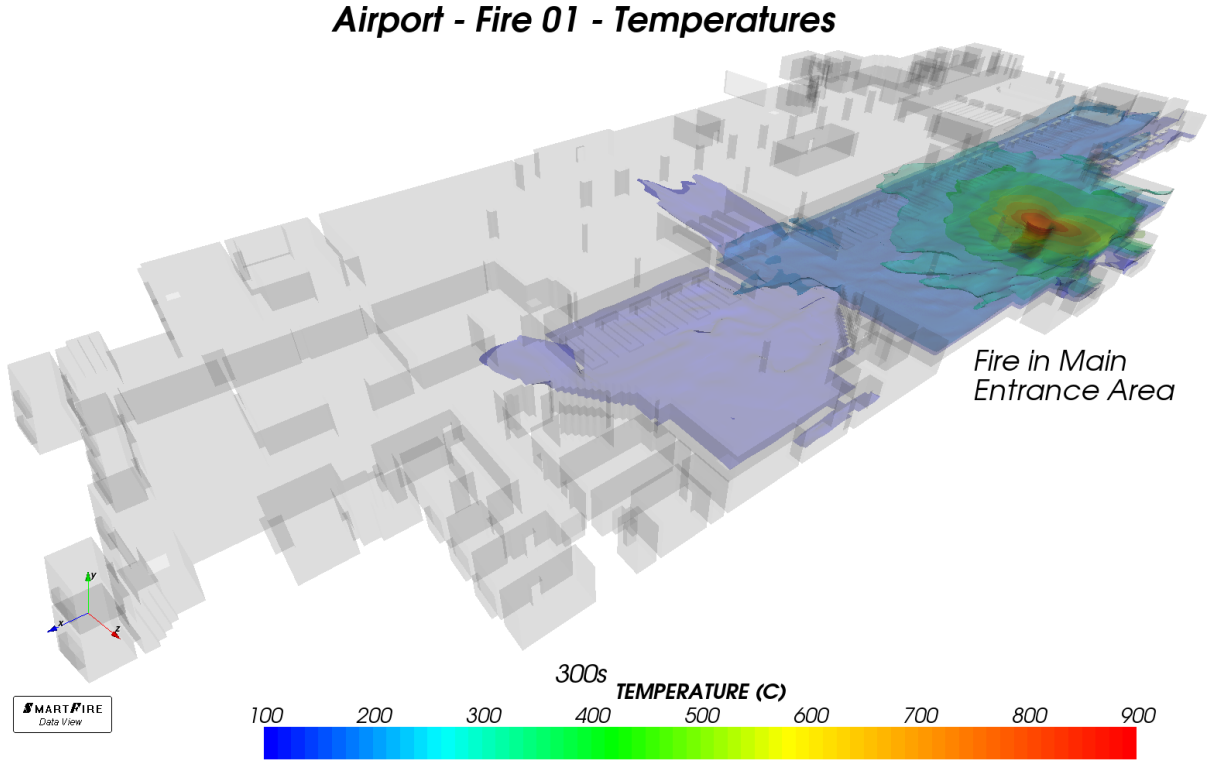
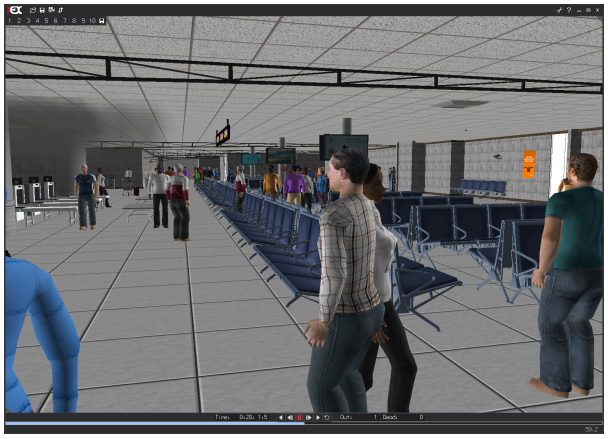
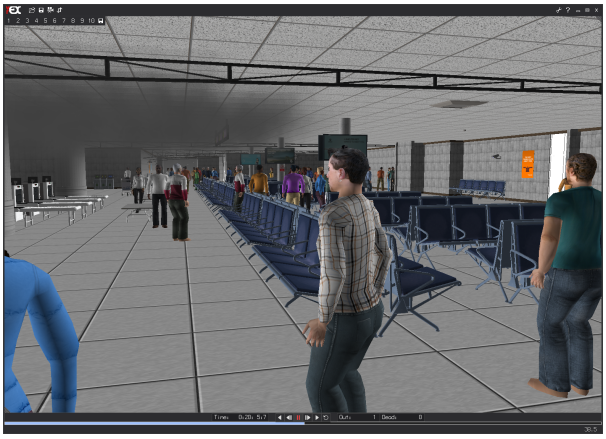


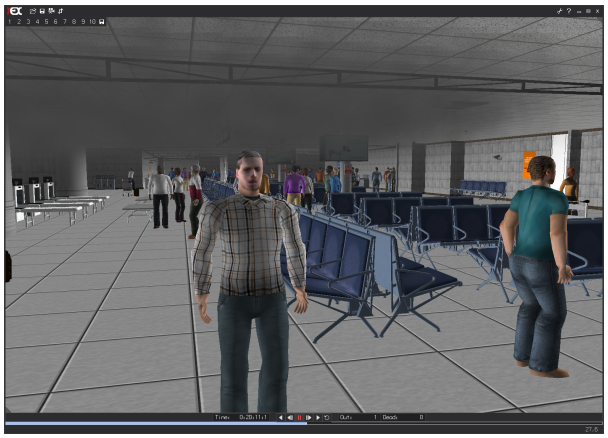
Figure A.3: Airport temperature graph in SMARTFIRE



(a) Time since start of fire: 1.5 seconds.



(b) Time since start of fire: 5.7 seconds.



(c) Time since start of fire: 11.1 seconds.



(d) Time since start of fire: 17.0 seconds.

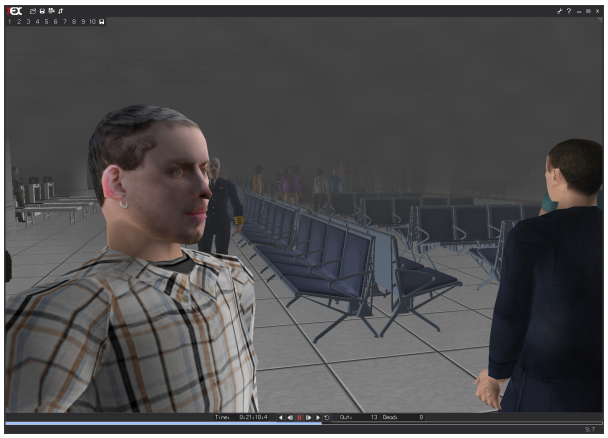
Figure A.4: vrEXODUS smoke visualisation progress of SMARTFIRE fire simulation in the airport geometry. Part 1.



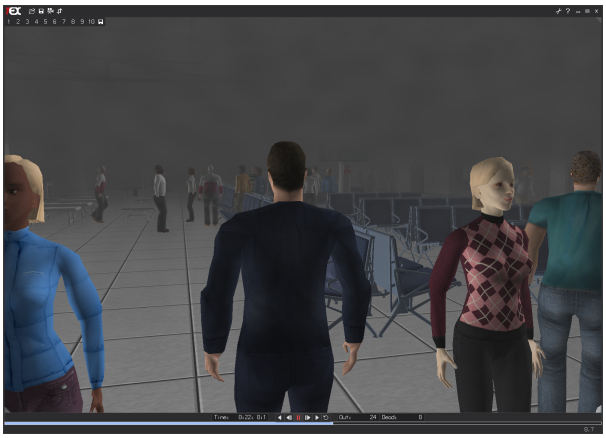
(a) Time since start of fire: 33.5 seconds.



(b) Time since start of fire: 46.0 seconds.



(c) Time since start of fire: 1 minute and 10.4 seconds.



(d) Time since start of fire: 2 minutes and 0.1 seconds.

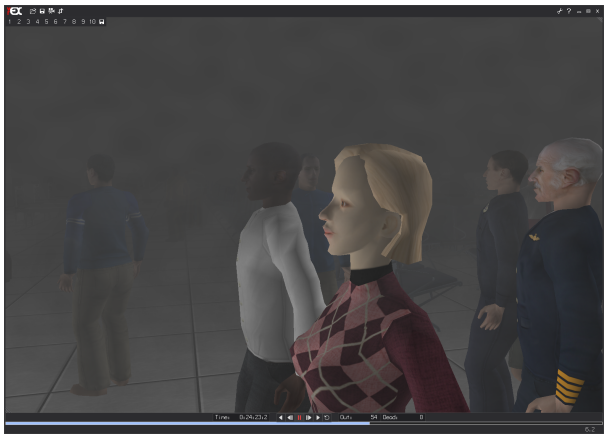
Figure A.5: vrEXODUS smoke visualisation progress of SMARTFIRE fire simulation in the airport geometry. Part 2.



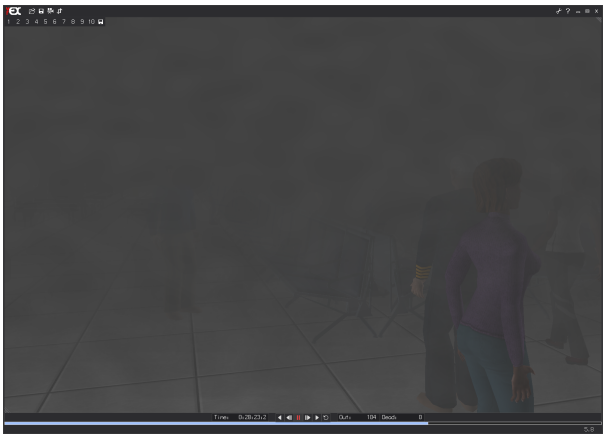
(a) Time since start of fire: 2 minutes and 52.5 seconds.



(b) Time since start of fire: 3 minute and 20.4 seconds.



(c) Time since start of fire: 4 minute and 23.2 seconds.

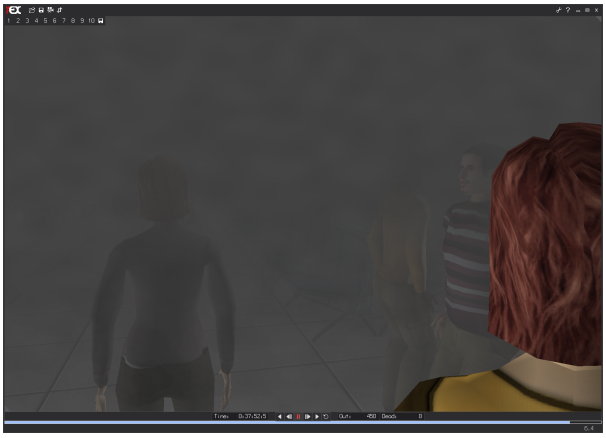


(d) Time since start of fire: 8 minute and 23.2 seconds.

Figure A.6: vrEXODUS smoke visualisation progress of SMARTFIRE fire simulation in the airport geometry. Part 3.



(a) Time since start of fire: 12 minute and 10.4 seconds.



(b) Time since start of fire: 37 minute and 52.5 seconds.

Figure A.7: vrEXODUS smoke visualisation progress of SMARTFIRE fire simulation in the airport geometry. Part 4.

Appendix B

UNIVERSITY OF GREENWICH RESEARCH ETHICS COMMITTEE ACCEPTANCE LETTER

An electronic copy of University of Greenwich Research Ethics Committee acceptance letter is attached to this appendix. The ethics approval is for the case studies detailed in Chapter 6 and the virtual smoke visibility validation tests described in Chapter 4. The data collection was accomplished through the approved questionnaire found in Appendix C in addition to the recorded simulation results and footfall contours presented in Appendix D.

Simo Haasanen
University of Greenwich
Faculty of Liberal Arts & Sciences
School of Computing & Mathematical Sciences
Greenwich Campus

Direct Line 020 8331 8842
Direct Fax 020 8331 8824
Email researchethics@gre.ac.uk
Our Ref UREC/18.4.6.i.i
Date: 12th March 2019

Dear Simo,

University Research Ethics Committee – Minute 18.4.6.i.i

TITLE OF RESEARCH: Immersive real-time multi-user interaction with computer simulated pedestrians during emergencies

I am writing to confirm that the above application has been **approved** by Chair's Action on behalf of the Committee and that you have permission to proceed.

I am advised by the Committee to remind you of the following points:

- You must notify the Committee immediately of any information received by you, or of which you become aware, which would cast doubt upon, or alter, any information contained in the original application, or a later amendment, submitted to the Committee and/or which would raise questions about the safety and/or continued conduct of the research;
- You must comply with Data Protection legislation;
- You must refer proposed amendments to the protocol to the Committee for further review and obtain the Committee's approval thereto prior to implementation (except only in cases of emergency when the welfare of the subject is paramount).
- You are authorised to present this University of Greenwich Research Ethics Committee letter of approval to outside bodies in support of any application for further research clearance.

On behalf of the Committee may I wish you success in your project.

Yours sincerely

[Signature redacted]

Peter Garrod
Secretary, University Research Ethics Committee

cc. Dr Peter Lawrence, FLAS
Professor Ed Galea, FLAS

University of Greenwich
Greenwich Campus
Old Royal Naval College
Park Row
London SE10 9LS
Telephone: +44 (0)20 8331 8000

University of Greenwich, a charity and company limited by guarantee, registered in England (reg. no. 986729). Registered Office: Old Royal Naval College, Park Row, London SE10 9LS

Appendix C

QUESTIONNAIRE FOR USE CASE STUDIES

A copy of the questionnaire used in the case studies and the validation case is presented in this appendix. The questionnaire is filled in two parts by the participant. The exercises were split into two separate sessions as the questionnaire contains a set of group exercises as well as a single user exercise.

A smoke visibility exercise is the first exercise the participants performed, and this was done as a single user exercise with no imposed time limit. This exercise aims to identify the point when an object becomes visible and distinguishable through virtual smoke. The exercise looks at both the light extinction coefficient and the number of particles per meter. The results of this exercise are recorded as part of the smoke validation case in Section 4.7 in Chapter 4. Examples of the visibility of the objects at key smoke obscuration levels are recorded in Appendix E.

The group exercises were comprised of three separate exercises. The first two exercises required a pair of users to evacuate an airport, first without smoke and then in smoke. The last group exercise required the pair of users to secure an area by directing computer-controlled agents away from a suspicious/reported package. Each exercise was performed three times to established if the participants could improve their results in the subsequent runs. Further details of the group exercises and the results are described in Chapter 6. Summary data of the group exercises are recorded in Appendix D.

The data in the questionnaires were collected anonymously by giving each participant a unique randomly generated four-digit participation number. Once the participant partnered up with another participant to perform the group exercise, the partner's participant number was recorded in the form to link the data together. The smoke visibility exercise took about one hour to complete by each participant. The group exercises took about two and half hours to complete by a single group of participants. The questionnaire is as follows:

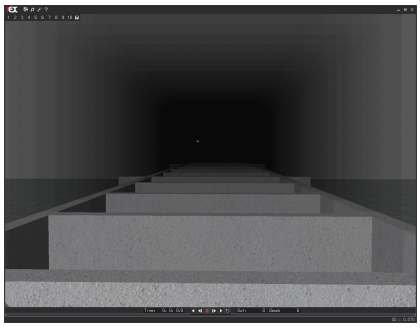
Use case for Immersive Real-time Interactive Simulation (IRIS) system	
Participation number:	**Date:**
Partner's number:	**Software version:**

Prior experience:

How frequently would you say you perform the following tasks?	Weekly / Monthly	Occasionally / Few times a year	Rarely	Years ago / Used to	Never
I play video games. (e.g. PC, mobile, console)					
I play first person video games.					

If relevant, how accurately would you say the following statements describe you or your prior experience?	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
I can easily navigate in a virtual 3D environment such as in a computer video game or virtual training tool.					
Most of the above experience are from using mouse and keyboard.					
I have been trained before using a virtual reality software / hardware.					

Smoke visibility exercises:





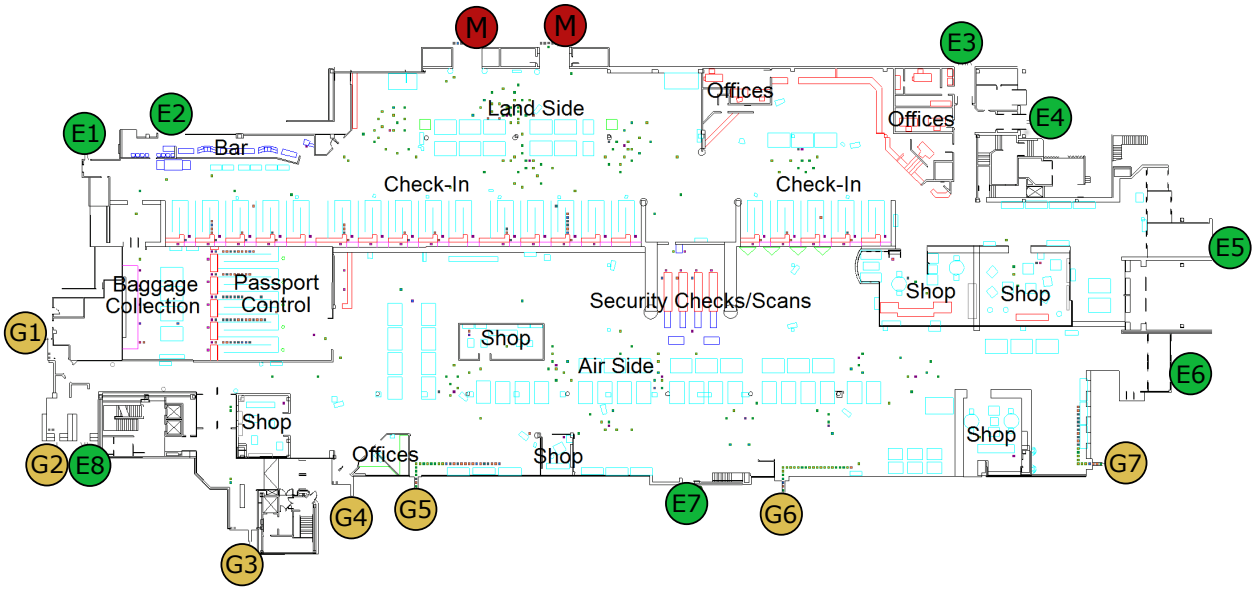
At what K value (at the bottom right of the screen) did the target just become visible / distinguishable? This exercise is looking for the threshold when the target becomes visible from being invisible. This exercise is also looking for variations between particle counts of smoke. Go through the number of particles per metre indicated with a P-value (P1 to P10, next to the K-value) and record the new K-value for each particle count if the smoke visibility changes.

	1	2	3	4	5	6	7	8	9
P1									
P2									
P3									
P4									
P5									
P6									
P7									
P8									
P9									
P10									

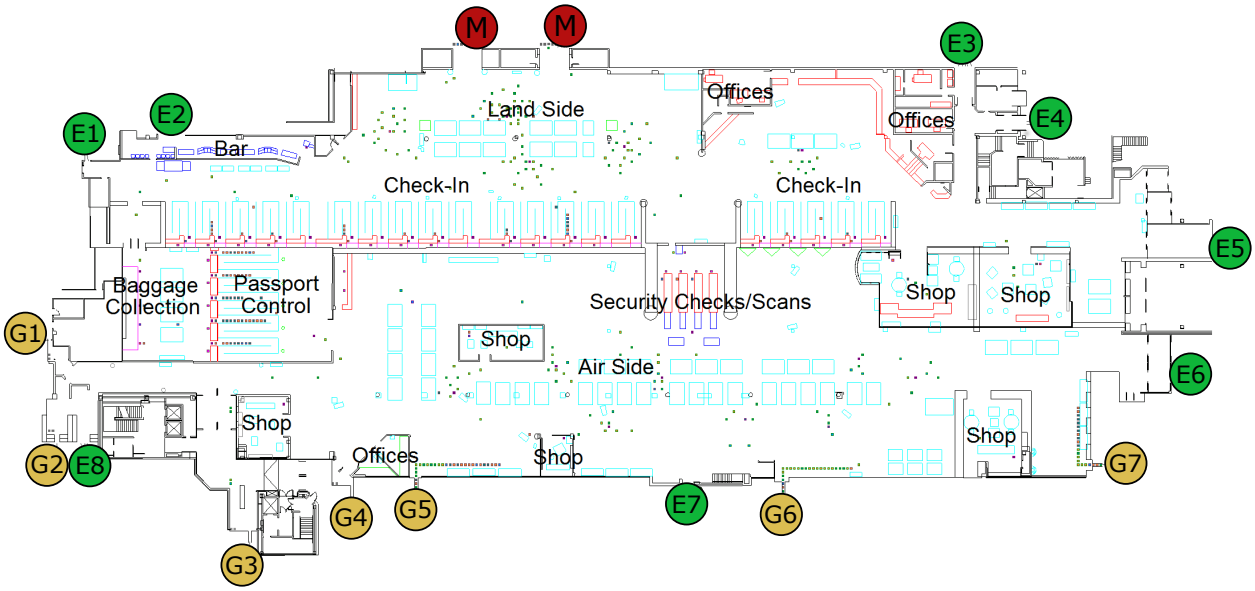
Assisted evacuation exercises:

No Smoke:

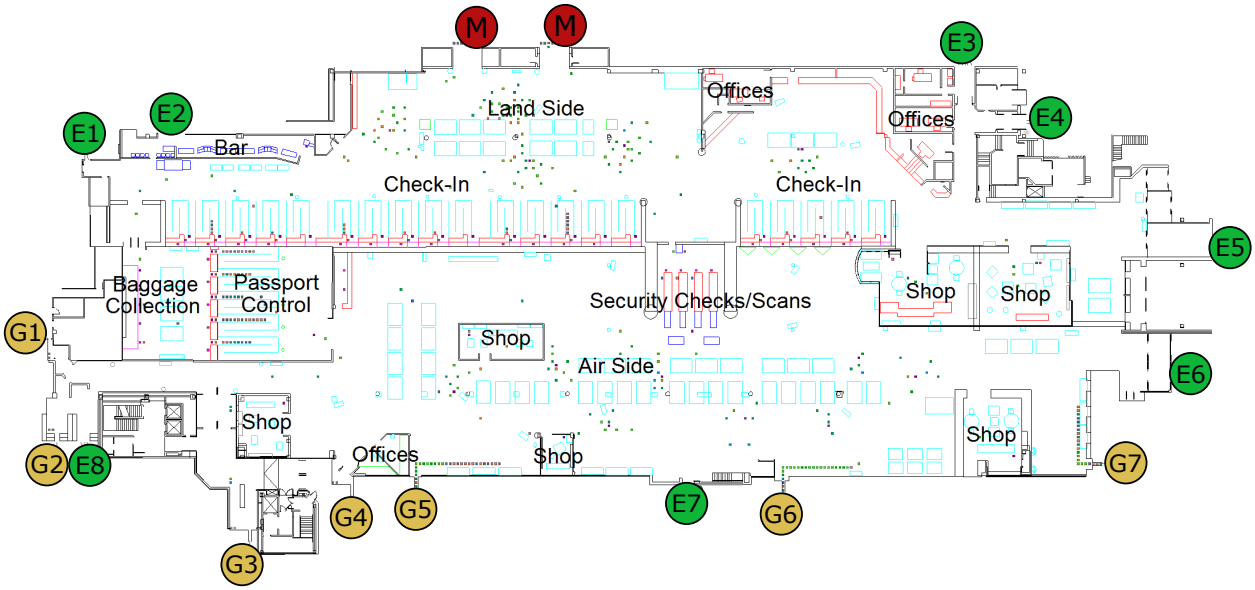
Please draw on the below map the route you took for exercise 1:



Please draw on the below map the route you took for exercise 2:

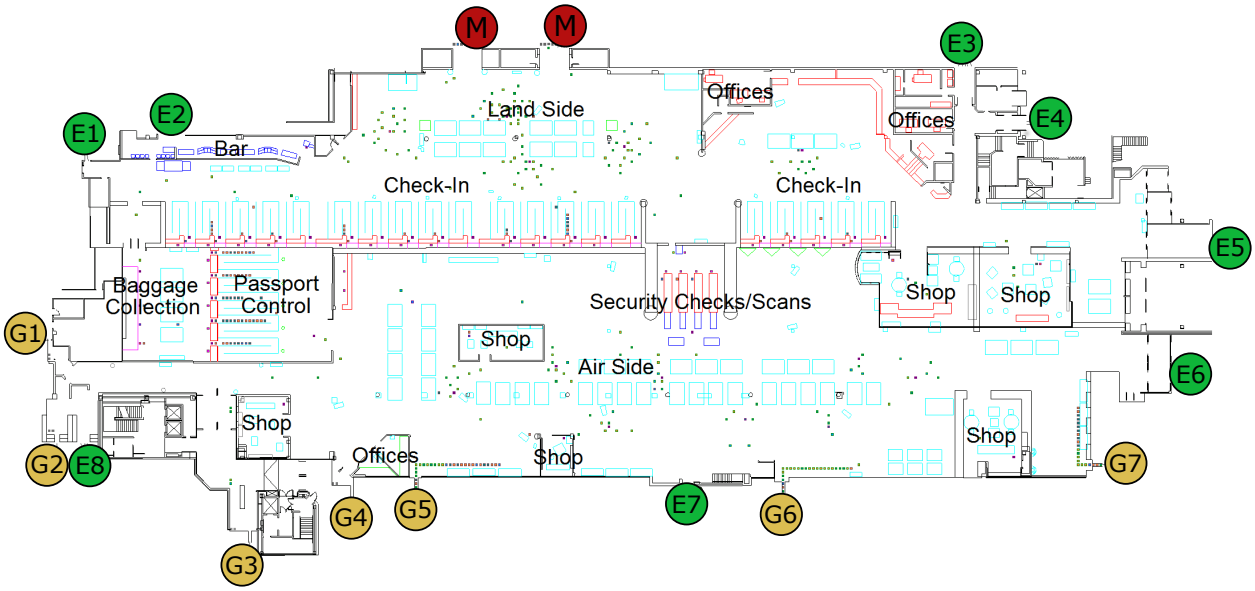


Please draw on the below map the route you took for exercise 3:

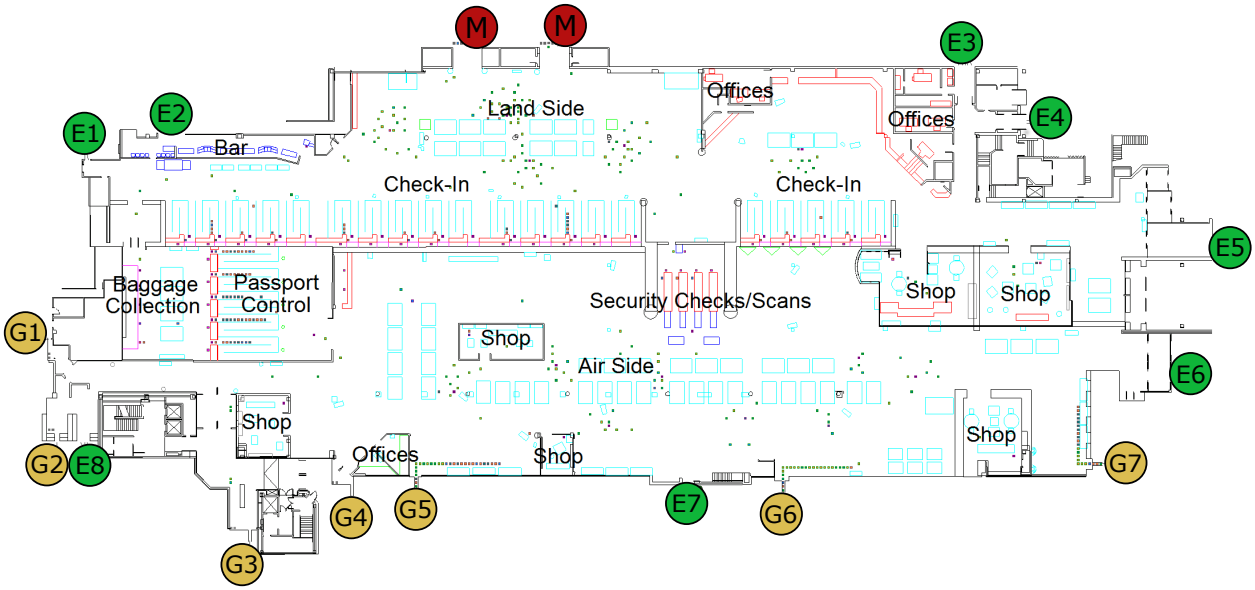


Smoke:

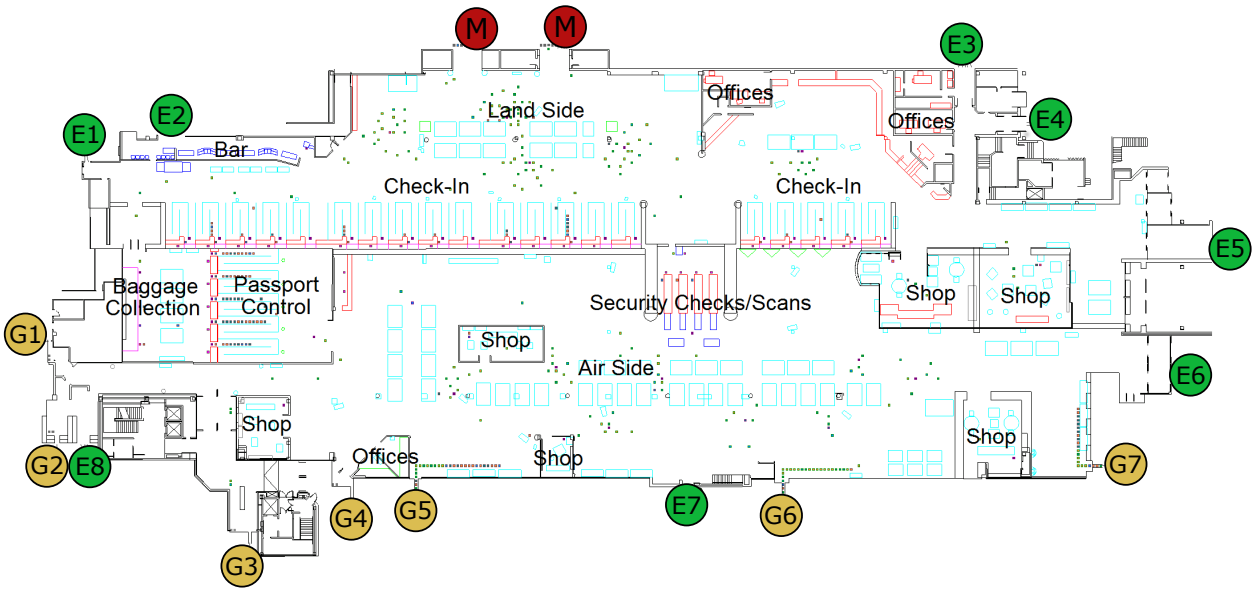
Please draw on the below map the route you took for exercise 4:



Please draw on the below map the route you took for exercise 5:



Please draw on the below map the route you took for exercise 6:

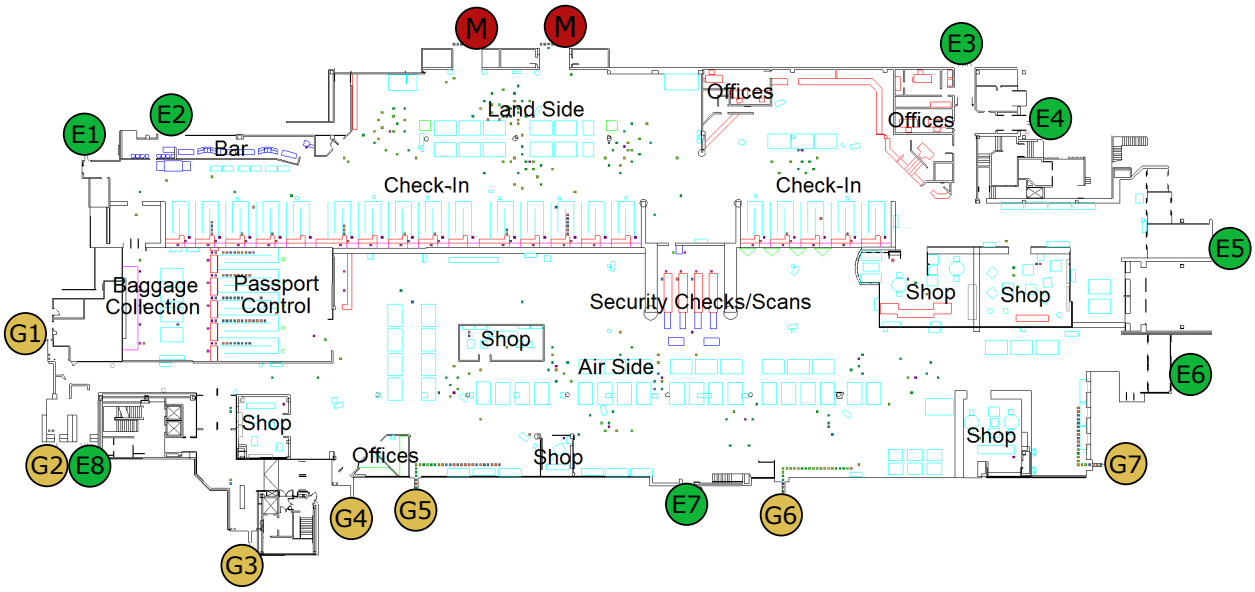


How did you perform during the exercise?	Exercise 1 (no smoke)	Exercise 2 (no smoke)	Exercise 3 (no smoke)	Exercise 4 (smoke)	Exercise 5 (smoke)	Exercise 6 (smoke)
Did you successfully evacuate the structure?						
Did you change your route from the previous run?						
Did you plan your route with your participation partners?						
Hand gesture and voice command impact:						
Did the agents respond to your commands?						
Did the agents respond to your commands how you expected?						
Was the behaviour of agents appear realistic?						
What commands did you use?						

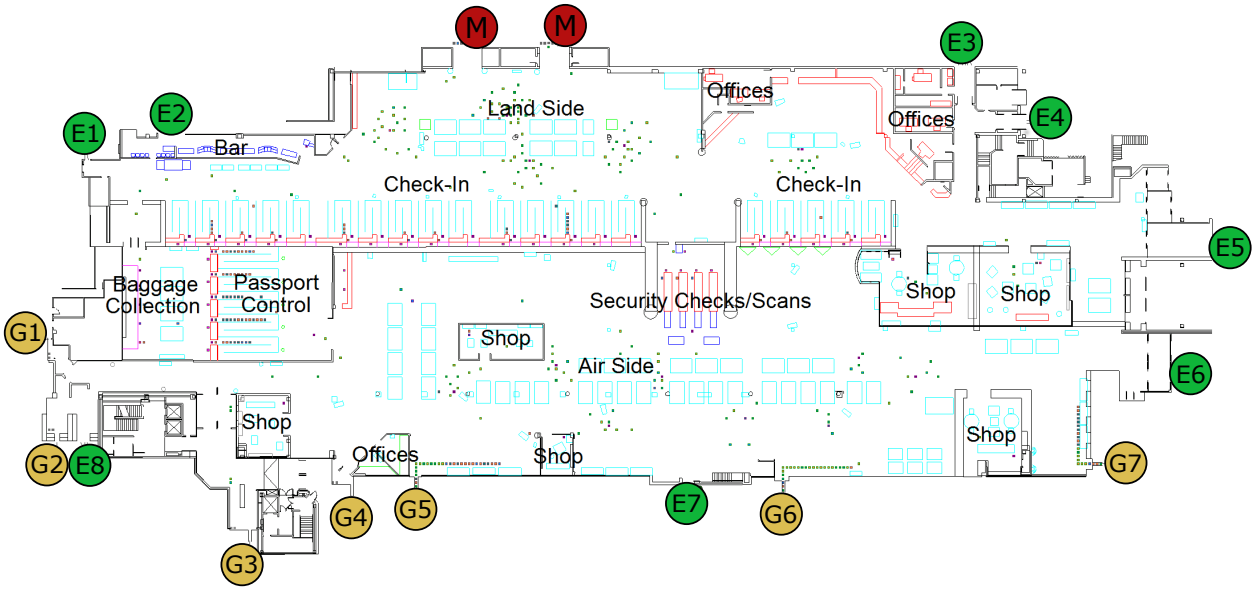
How realistic / realistically...	Very realistic	Realistic	Quite realistic	Not that realistic	Not realistic at all
... you thought the simulation was?					
... did the agents evacuate the building?					
... did the agents respond to your gestures?					
... did the agents move / navigate?					
How credible was the agent behaviour...					
... in response to your commands?					
... in the non-smoke environment?					
... in the smoke environment?					

Securing an area exercises:

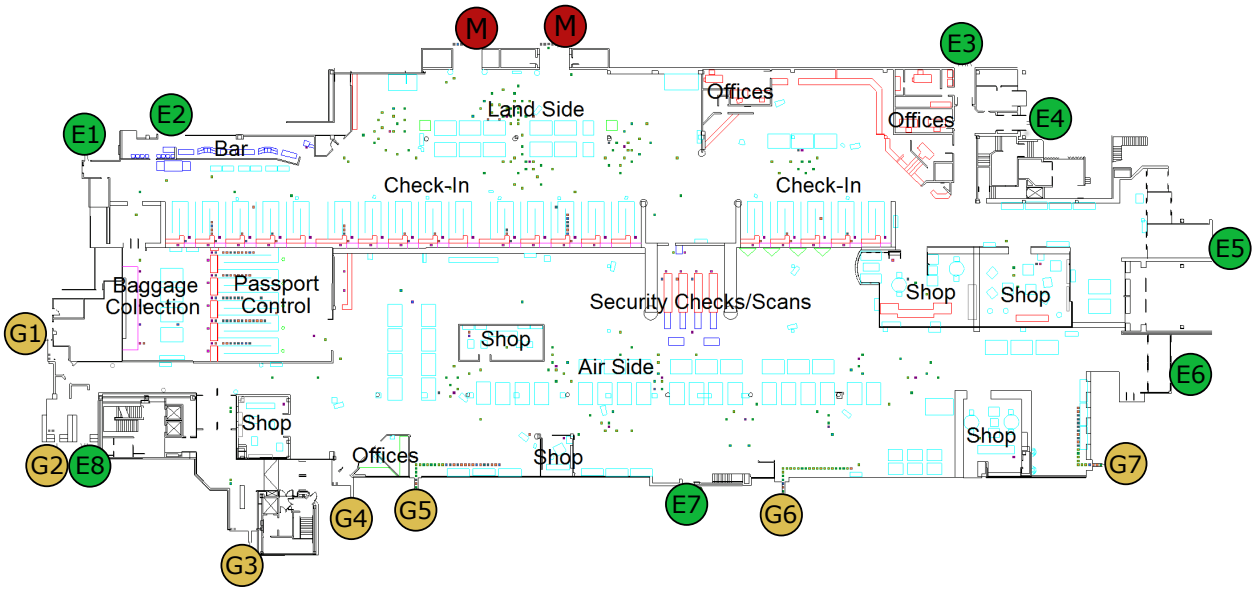
Please draw on the below map the location you occupied and the direction you directed the agents exercise 1:



Please draw on the below map the location you occupied and the direction you directed the agents exercise 2:



Please draw on the below map the location you occupied and the direction you directed the agents exercise 3:



How did you perform during the exercise?	Exercise 1	Exercise 2	Exercise 3
Did you successfully secure the area?			
How many agents got into the secured area?			
Did you change your position / direction from the previous run?			
Did you plan your position / direction with your participation partners?			

How realistic / realistically...	Very realistic	Realistic	Quite realistic	Not that realistic	Not realistic at all
... you thought the simulation was?					
... did the agents evacuate the building?					
... did the agents respond to your gestures?					
... did the agents move / navigate?					

General questions / feedback:

How accurately would you say the following statements describe you or your prior experience?	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
I feel that a system like this one, once it has been fully developed, could enhance current training methods. (In what way? Please also comment below)					
I feel that a system like this one, once it has been fully developed, would be an effective tool to identify, develop and explore new tactics when the threat or trends change.					
I feel that the statistics and assessment methods shown/described today will give sufficient performance feedback.					
I think other organisations would benefit from using this kind of system.					
I think a system like this one would enable you to exercise your training in decision making, analytical thinking and problem-solving skills.					
I found the movement controls (keyboard and mouse) intuitive to use.					
I found the gesture controls easy to use (keyboard).					
Comments:					

Can you foresee any immediate barriers in adopting a system such as this one into day-to-day training practice?

What did you particularly like or dislike about the IRIS system you tested today?

Did the Platform provide/Can you see the Platform providing a realistic training experience for you?

What would you like to see in the next version of the system? What future features (mentioned today) interested you?

Appendix D

USE CASE PERFORMANCE DATA FROM PARTICIPANTS

This appendix presents the evacuation performance graphs and footfall contours that were recorded as part of the group exercises in Chapter 6. Section D.1 shows the evacuation performance graphs that were taken from the point the group entered the simulation, which is exactly when 400 agents are in the simulation. Section D.2 shows the footfall contours for each group aiming to secure an area. Each pair of participants performed three runs for each exercise to establish if the participants could improve the results in the subsequent runs. The participant questionnaire associated with the runs can be seen in Appendix C. The results of the three runs from the assisted evacuation exercises are compared against a base case which is described in Section 6.3.2 in Chapter 6.

D.1 Assisted airport evacuation graphs

D.1.1 Assisted evacuation without smoke

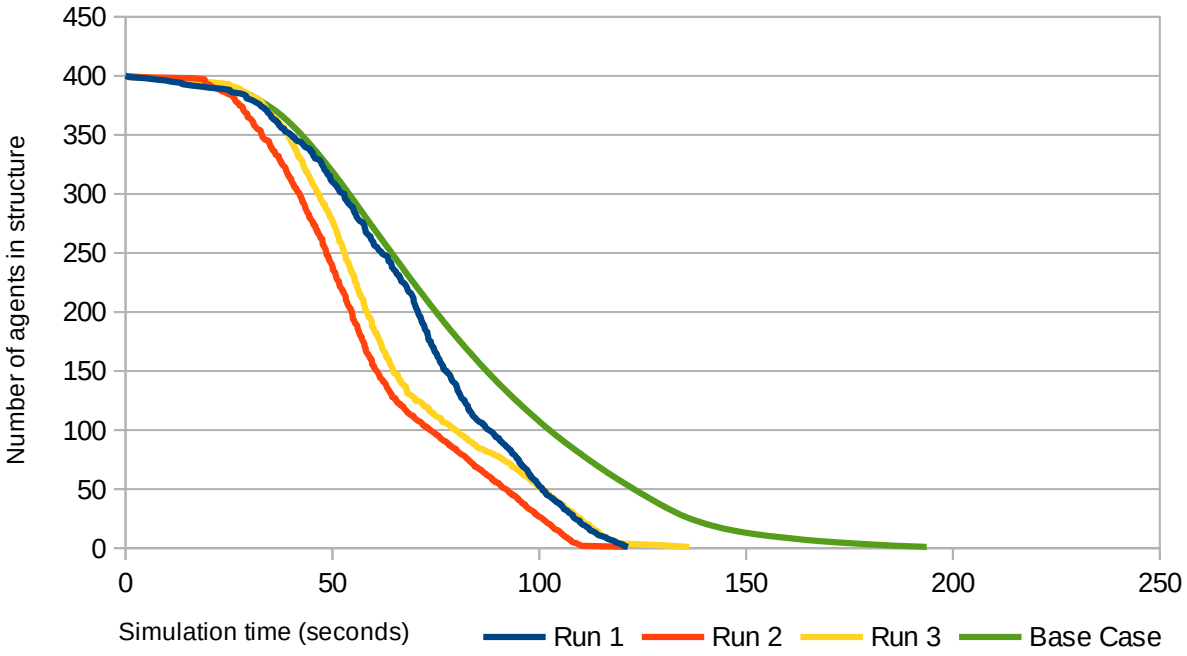


Figure D.1: Assisted evacuation without smoke: Group 1 performance

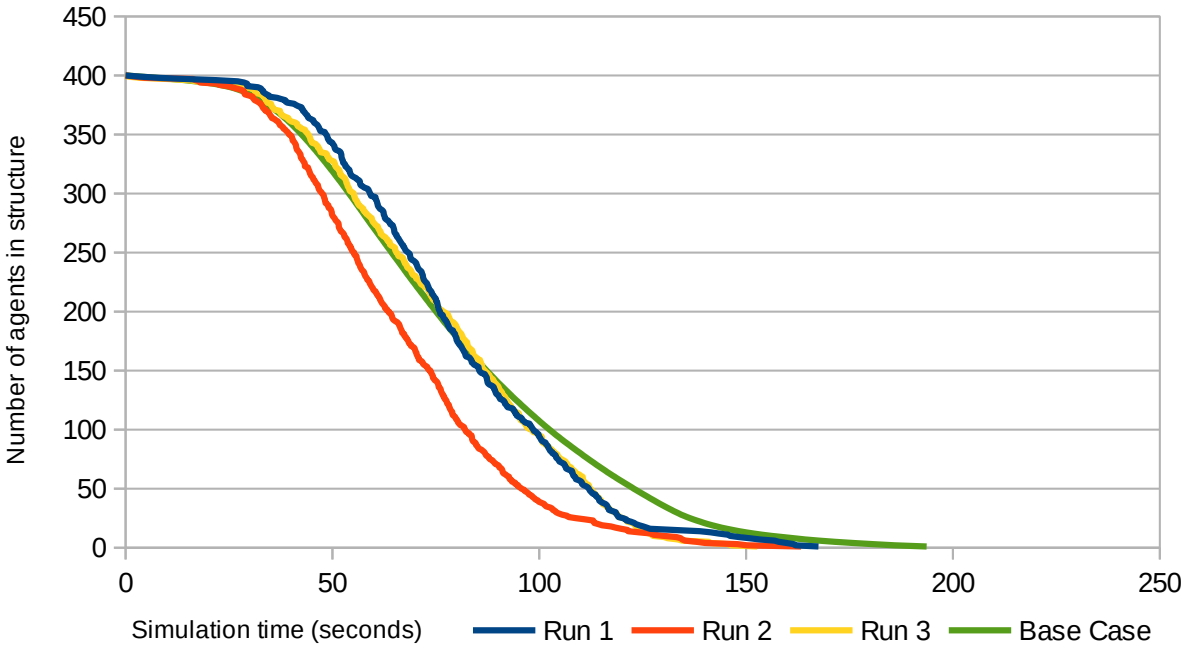


Figure D.2: Assisted evacuation without smoke: Group 2 performance

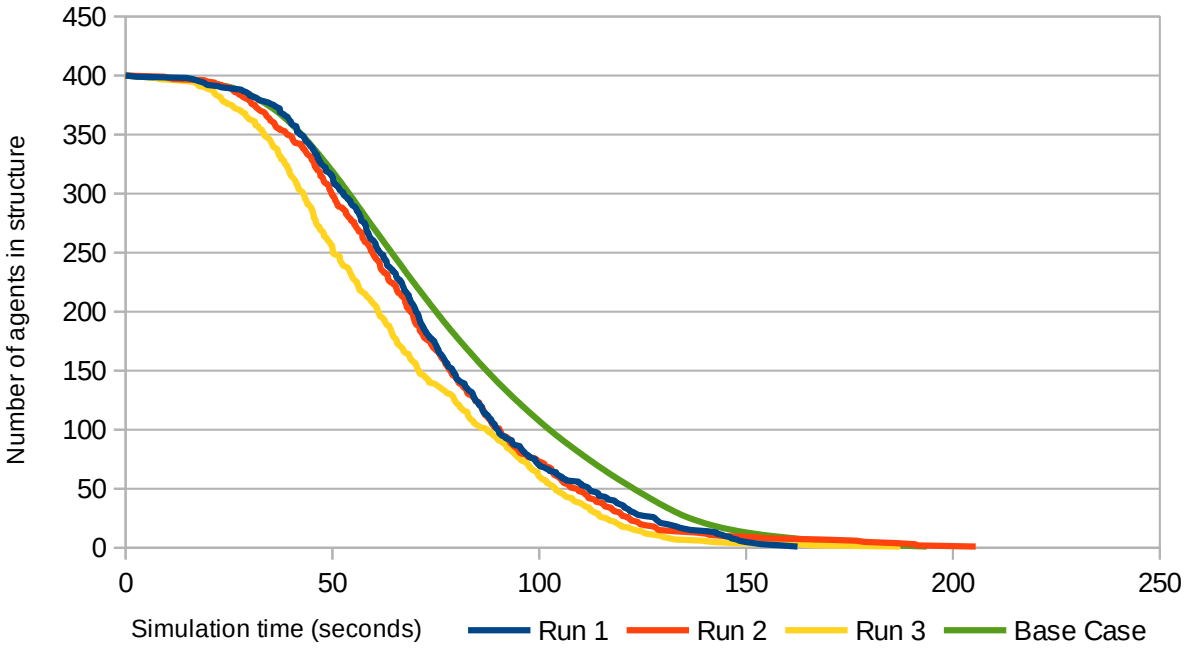


Figure D.3: Assisted evacuation without smoke: Group 3 performance

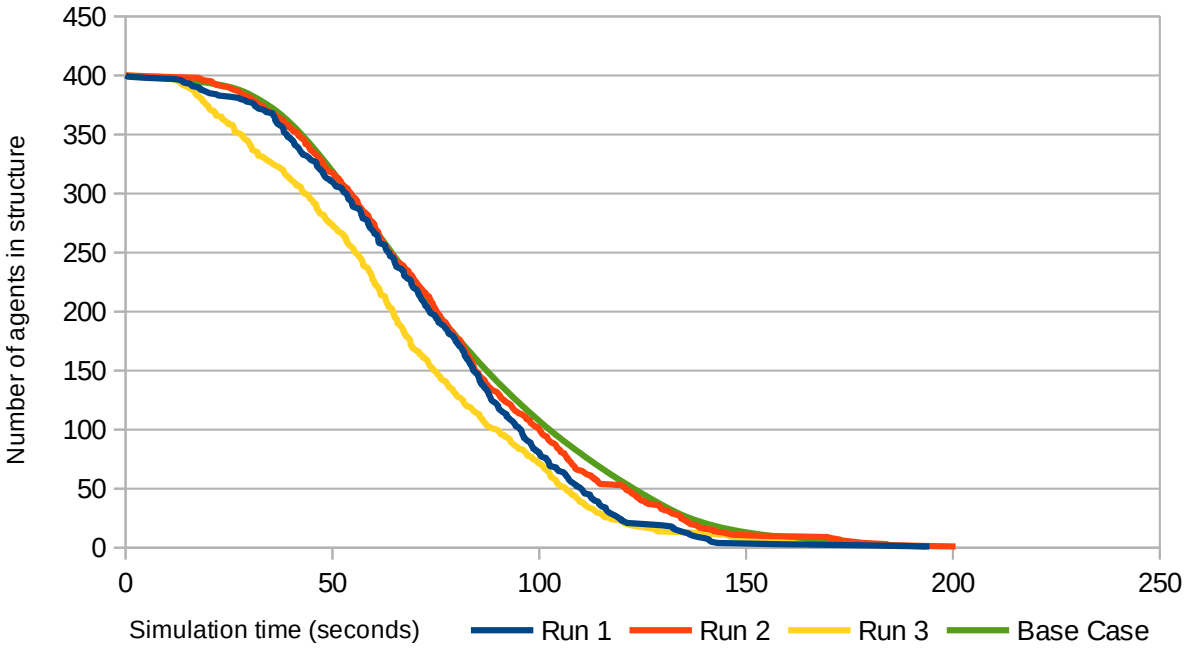


Figure D.4: Assisted evacuation without smoke: Group 4 performance

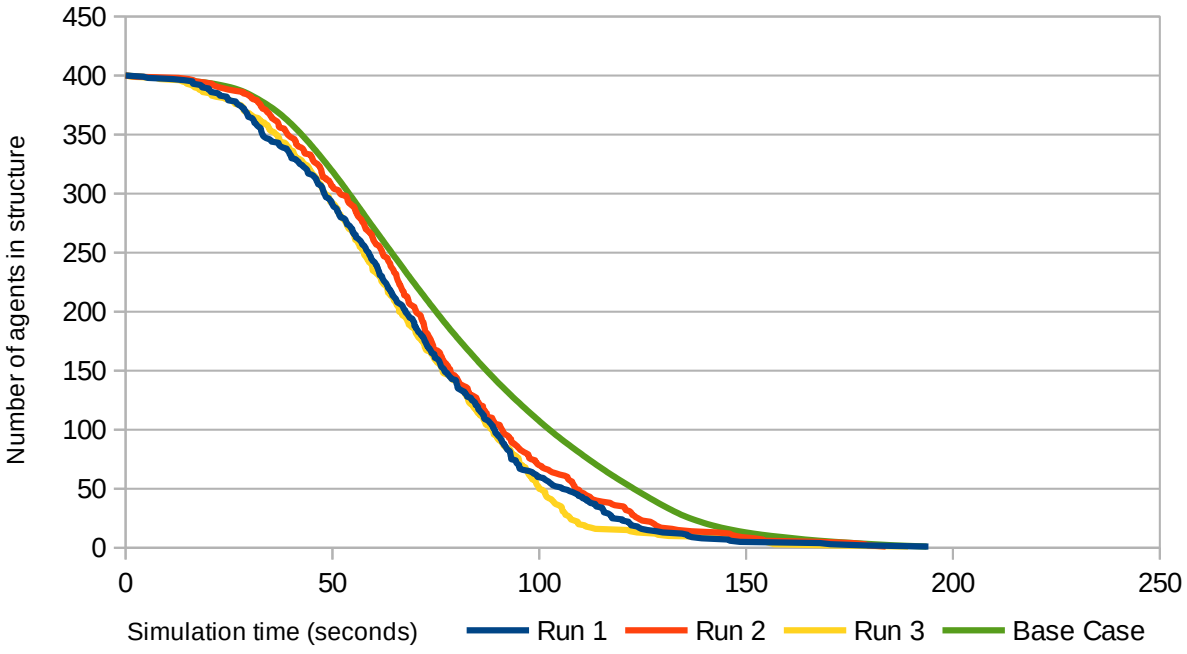


Figure D.5: Assisted evacuation without smoke: Group 5 performance

Total evacuation times (TET) for Figures D.1, D.2, D.3, D.4, and D.5 can be found in Table D.1.

Table D.1: Assisted evacuation without smoke: total evacuation time (TET) results in seconds

TET	Group 1	Group 2	Group 3	Group 4	Group 5
Run 1	121.45	167.49	162.42	194.30	194.07
Run 2	121.16	163.27	205.50	200.63	183.75
Run 3	136.25	152.71	187.31	165.38	189.18

D.1.2 Assisted evacuation in smoke

The distance of visibility in smoke was set to 8 metres.

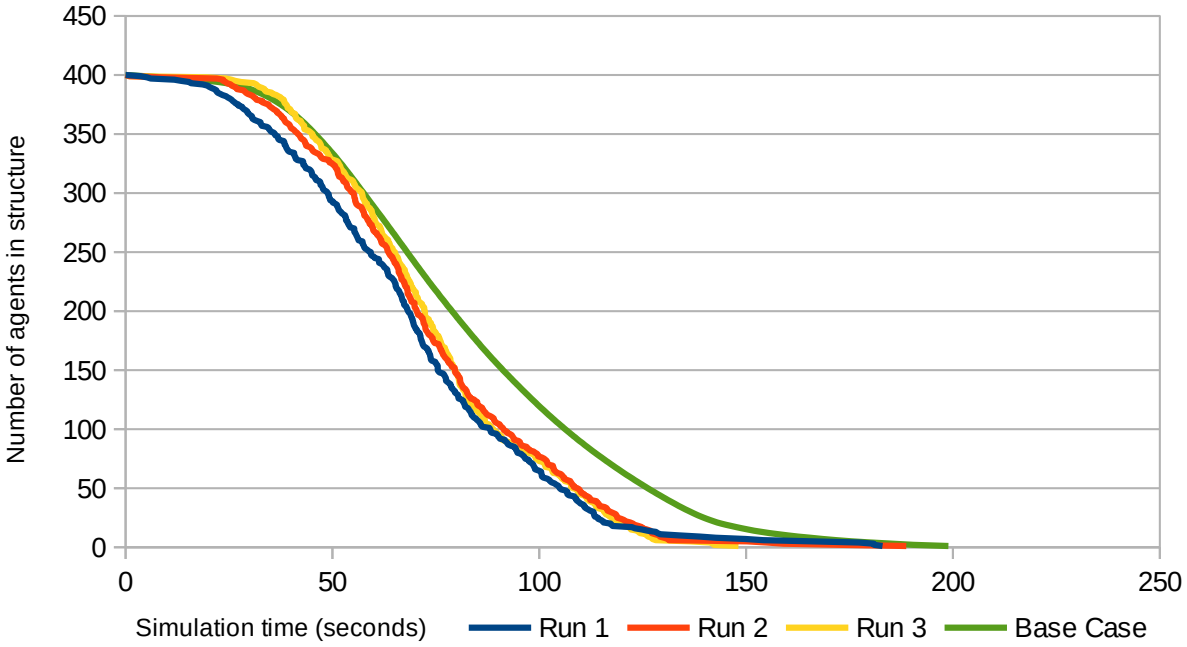


Figure D.6: Assisted evacuation in smoke: Group 1 performance

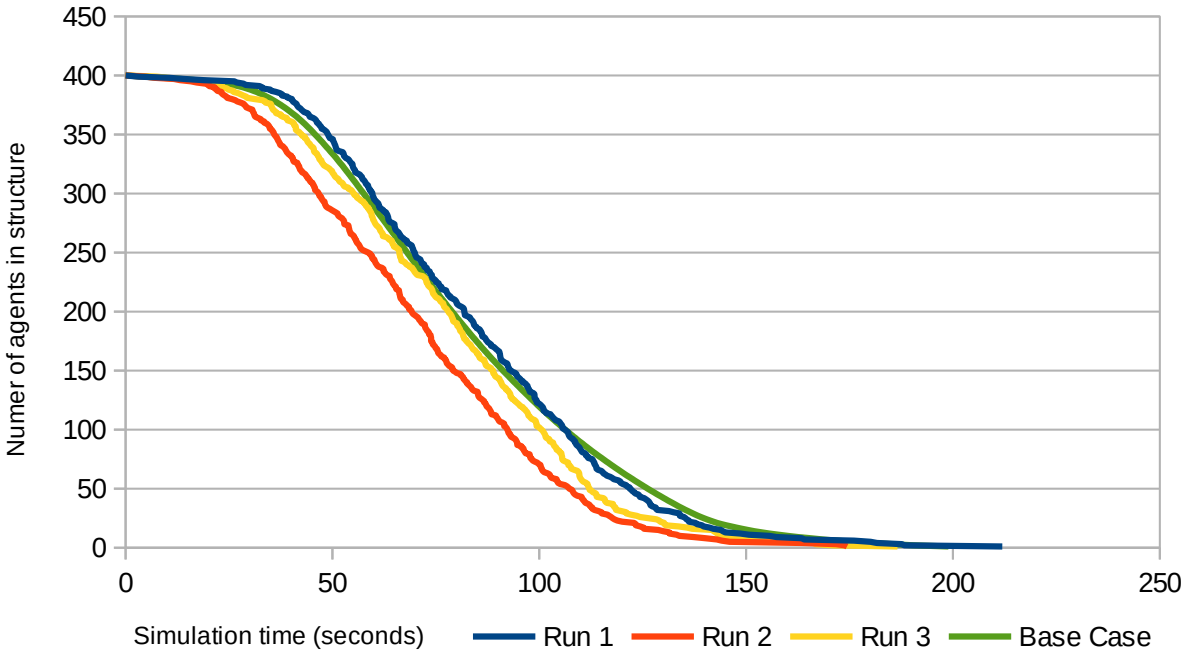


Figure D.7: Assisted evacuation in smoke: Group 2 performance

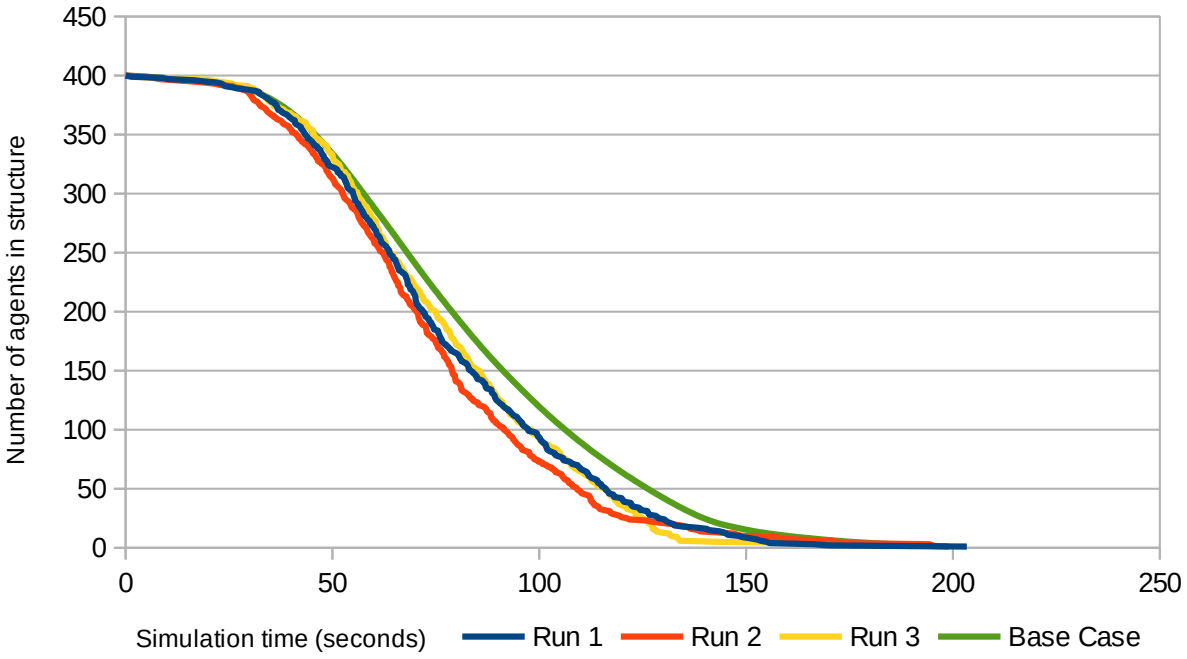


Figure D.8: Assisted evacuation in smoke: Group 3 performance

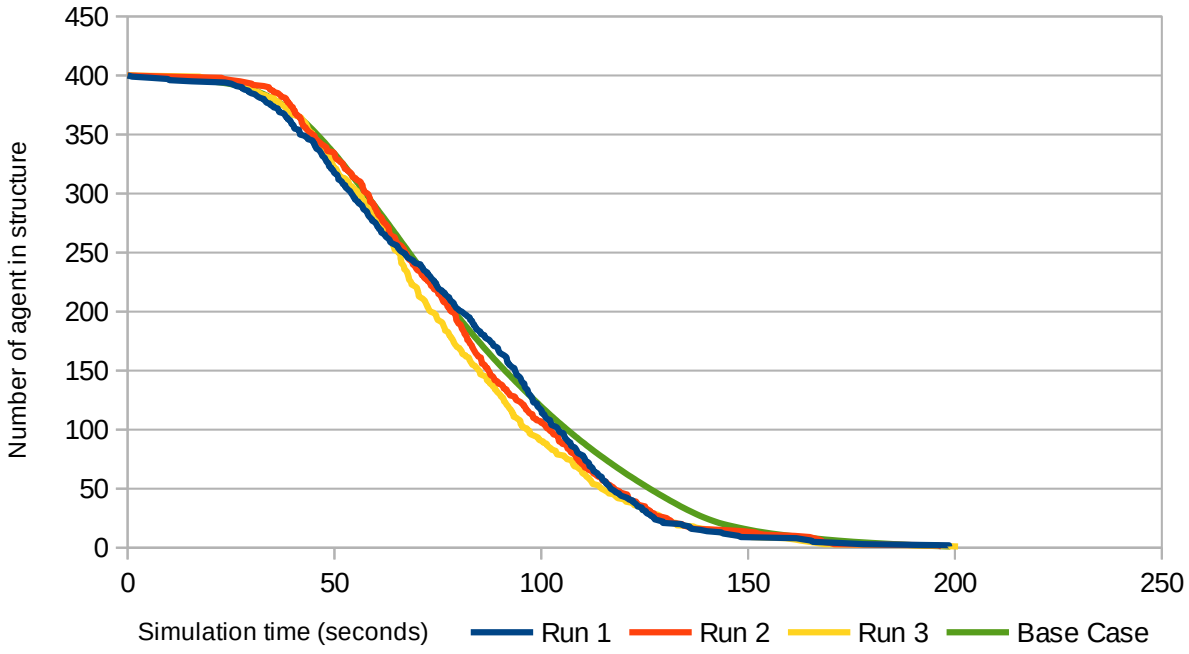


Figure D.9: Assisted evacuation in smoke: Group 4 performance

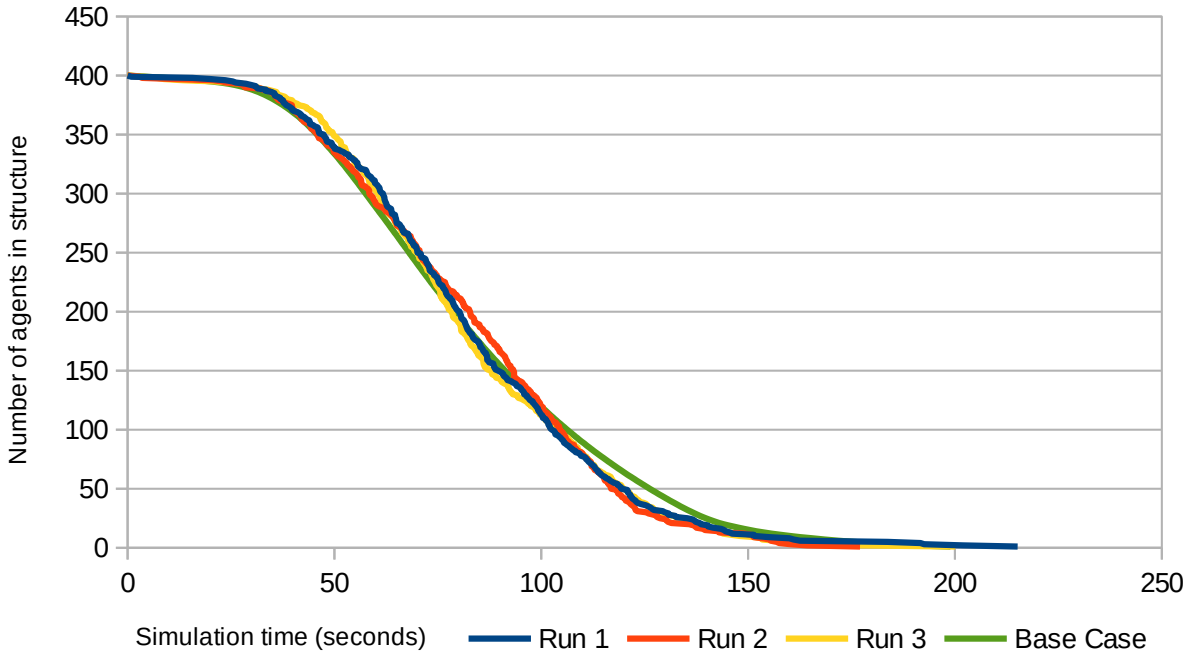


Figure D.10: Assisted evacuation in smoke: Group 5 performance

Total evacuation times (TET) for Figures D.6, D.7, D.8, D.9, and D.10 can be found in Table D.2.

Table D.2: Assisted evacuation in smoke: total evacuation time (TET) results in seconds

TET	Group 1	Group 2	Group 3	Group 4	Group 5
Run 1	182.97	211.94	203.35	198.61	215.75
Run 2	188.72	174.40	195.20	196.73	177.06
Run 3	148.16	186.59	190.06	200.75	200.08

D.2 Secure an area

D.2.1 Group 1 footfall

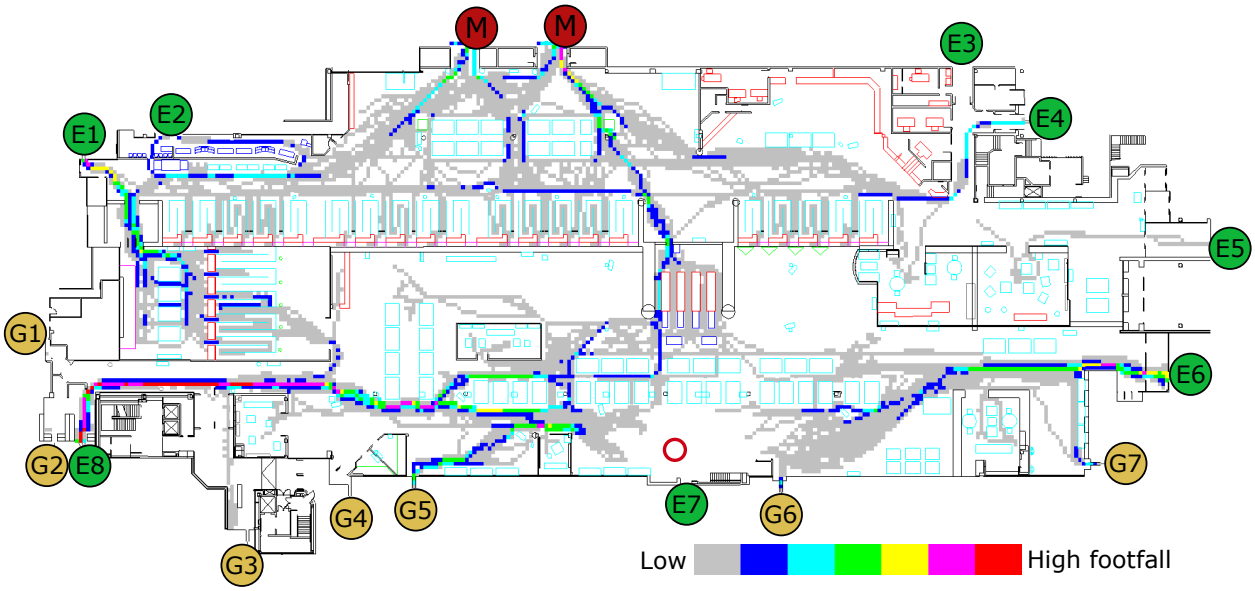


Figure D.11: Secure an area: Group 1, Run 1, footfall contour

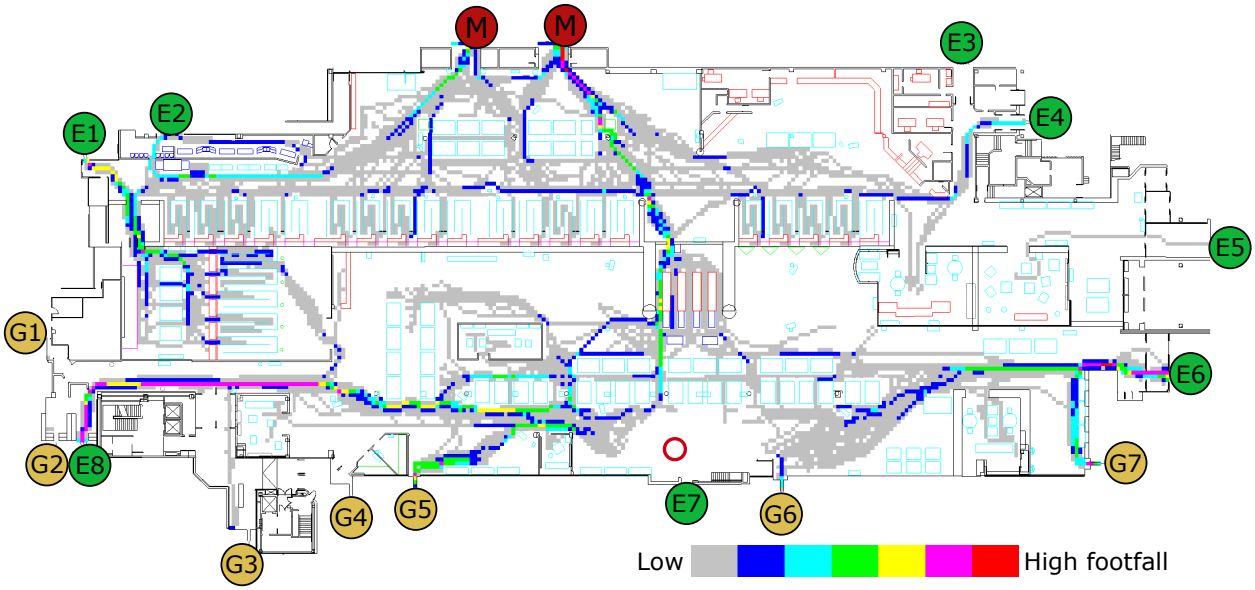


Figure D.12: Secure an area: Group 1, Run 2, footfall contour

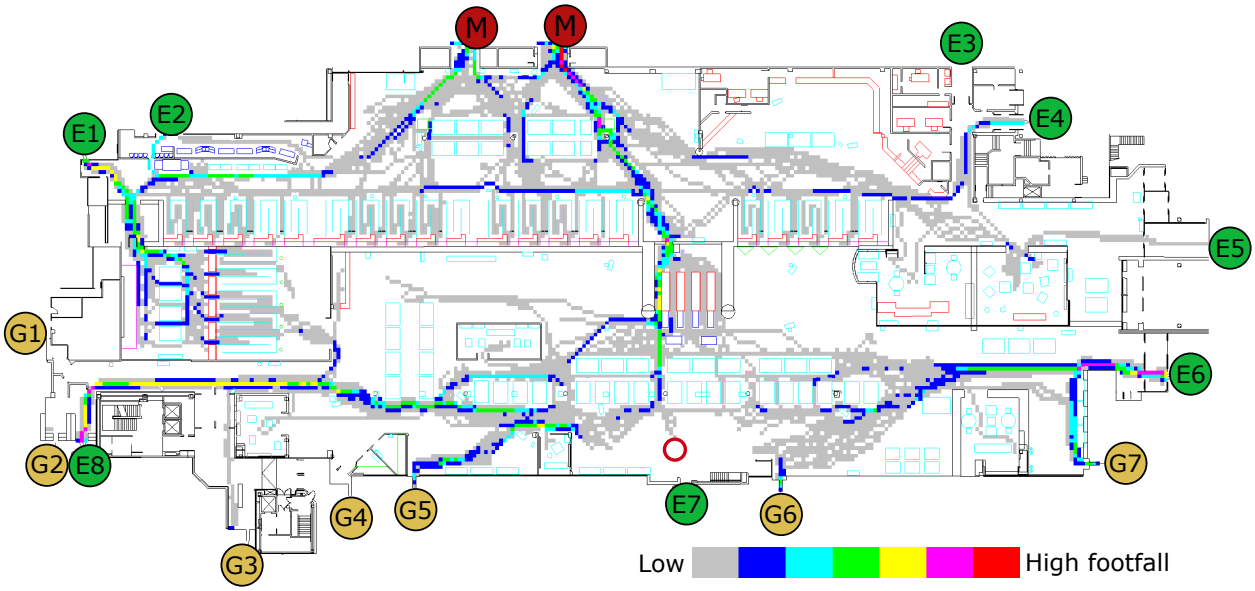


Figure D.13: Secure an area: Group 1, Run 3, footfall contour

D.2.2 Group 2 footfall

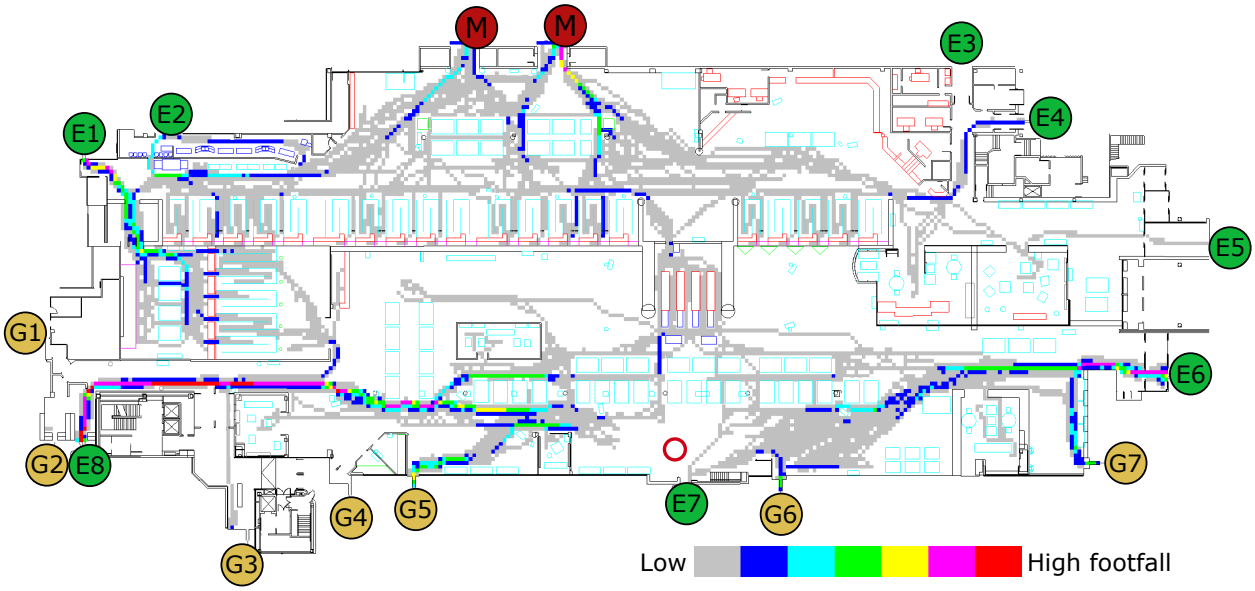


Figure D.14: Secure an area: Group 2, Run 1, footfall contour

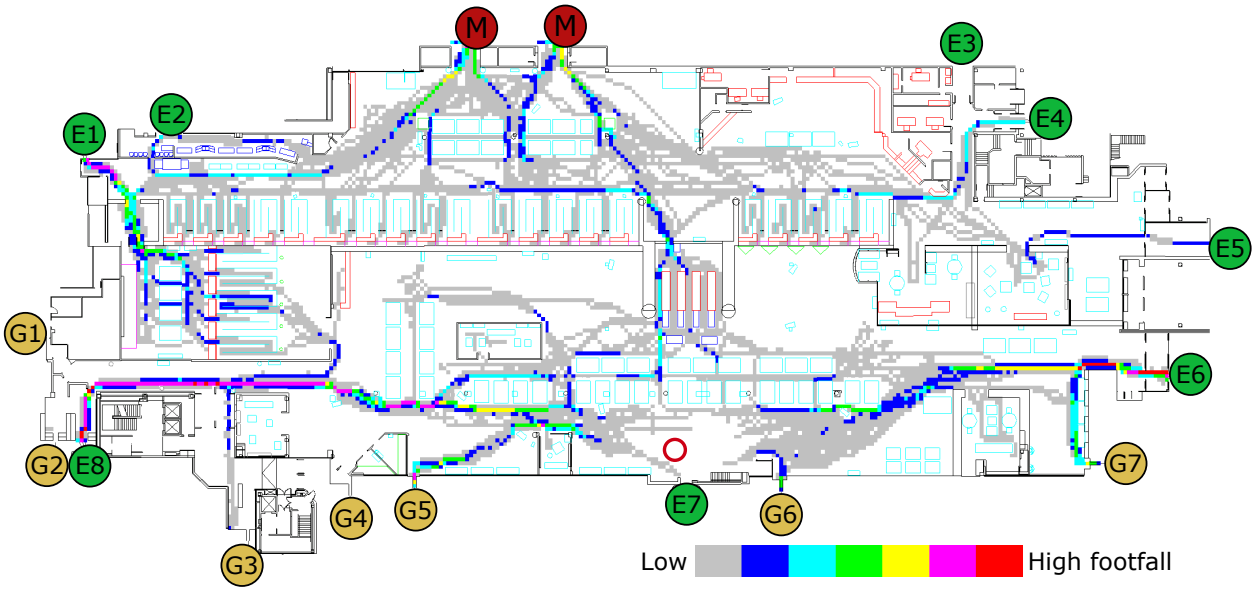


Figure D.15: Secure an area: Group 2, Run 2, footfall contour

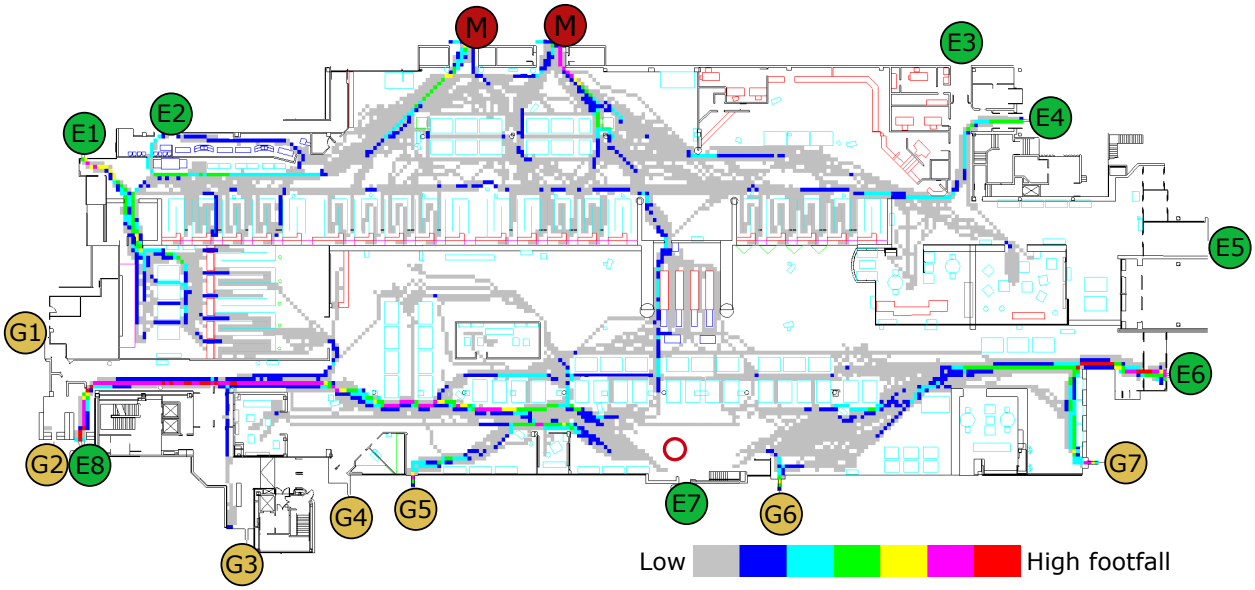


Figure D.16: Secure an area: Group 2, Run 3, footfall contour

D.2.3 Group 3 footfall

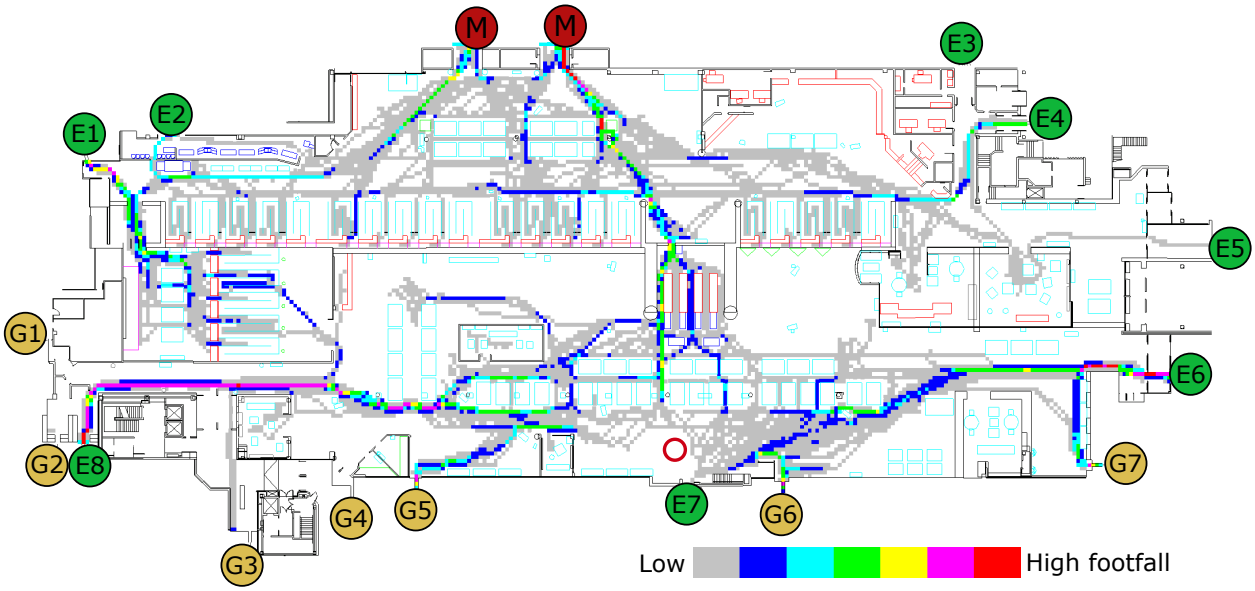


Figure D.17: Secure an area: Group 3, Run 1, footfall contour

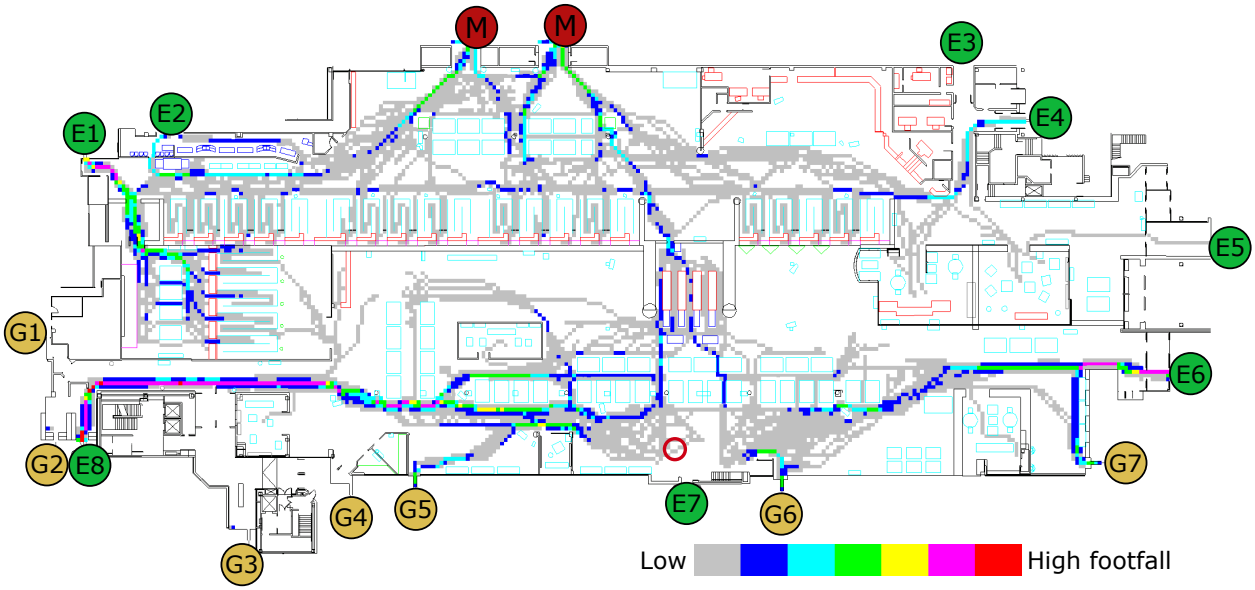


Figure D.18: Secure an area: Group 3, Run 2, footfall contour

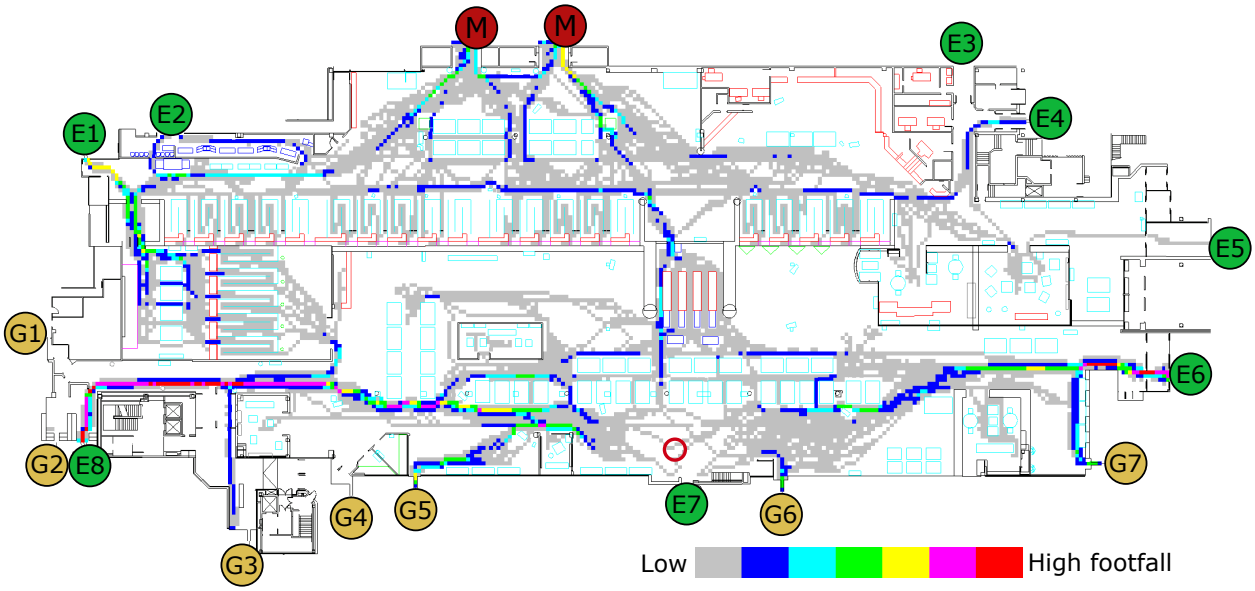


Figure D.19: Secure an area: Group 3, Run 3, footfall contour

D.2.4 Group 4 footfall

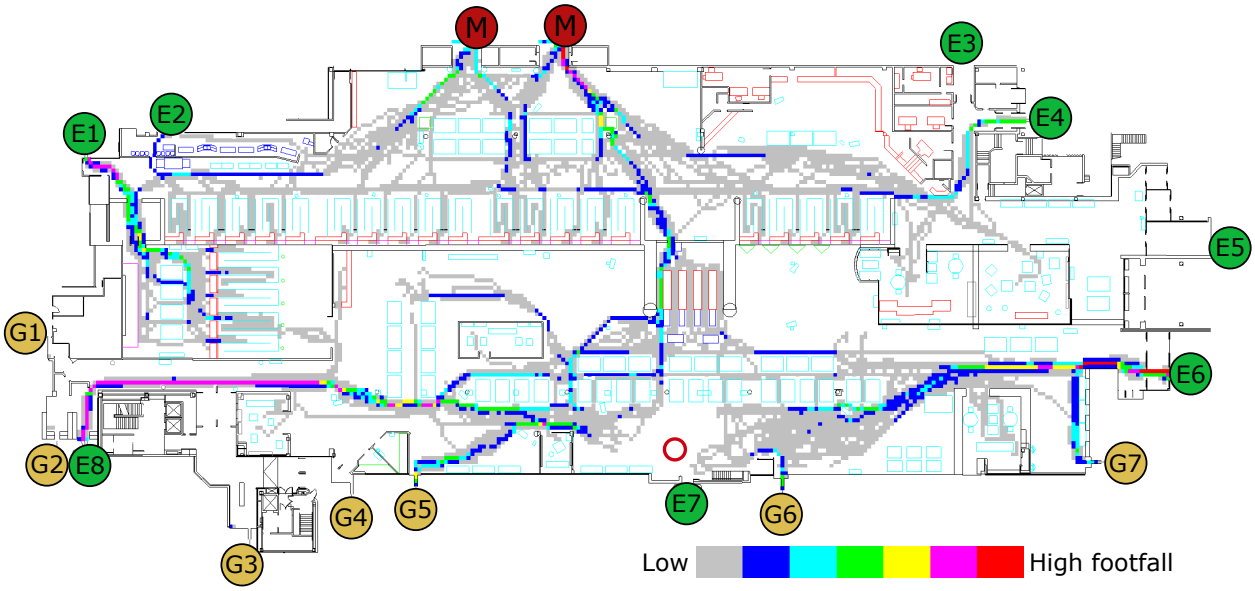


Figure D.20: Secure an area: Group 4, Run 1, footfall contour

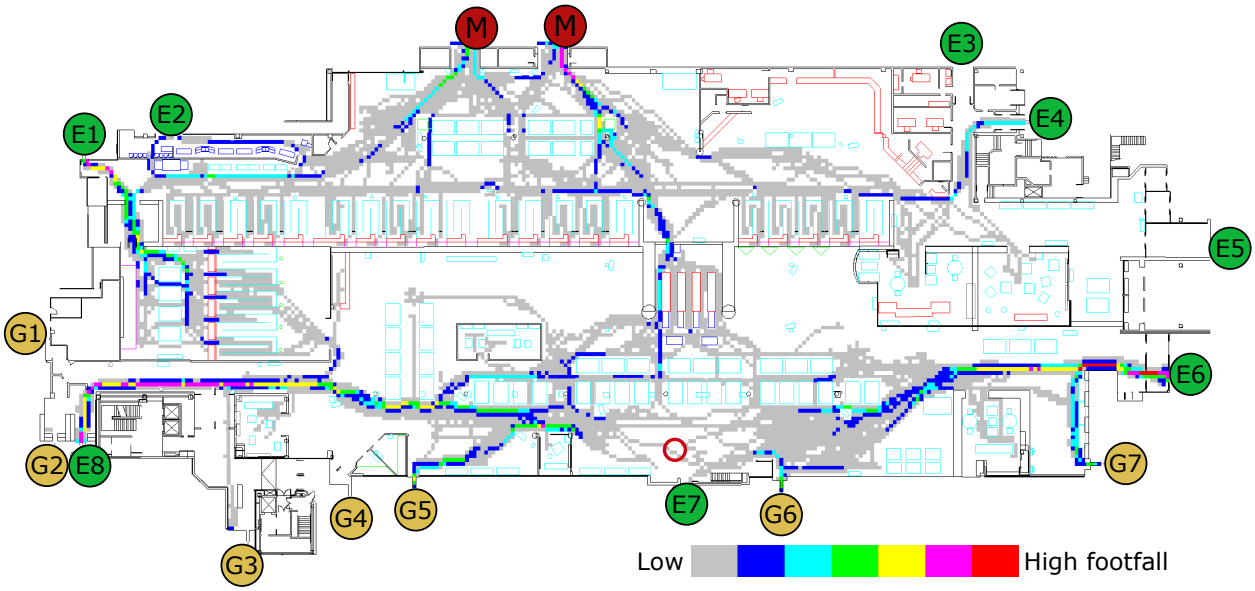


Figure D.21: Secure an area: Group 4, Run 2, footfall contour

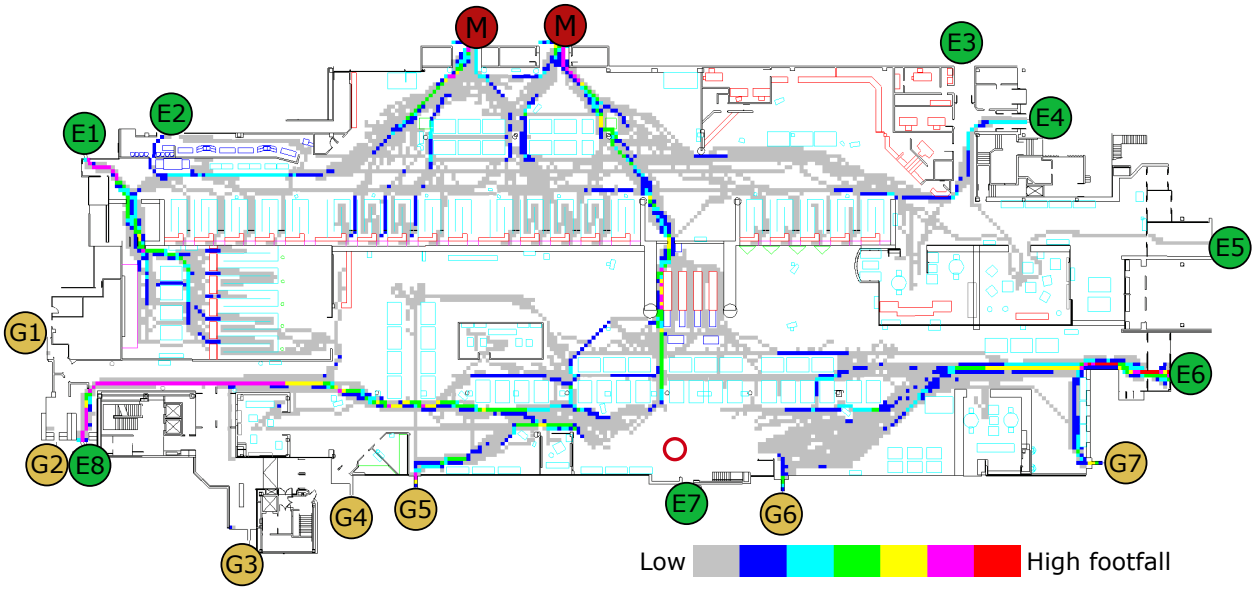


Figure D.22: Secure an area: Group 4, Run 3, footfall contour

D.2.5 Group 5 footfall

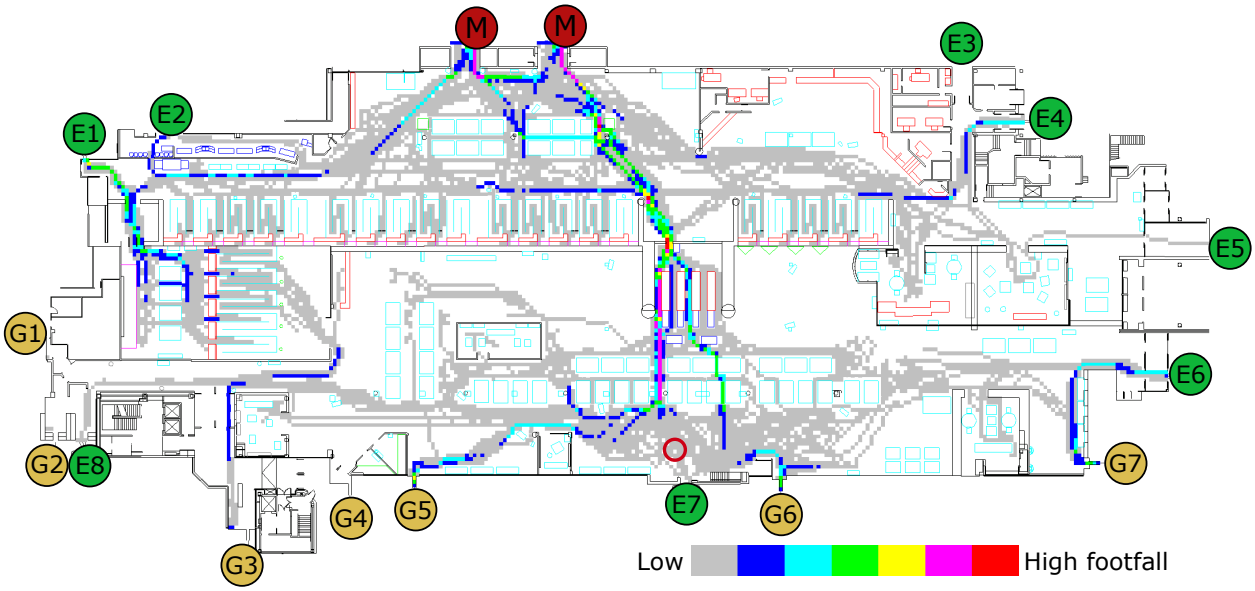


Figure D.23: Secure an area: Group 5, Run 1, footfall contour

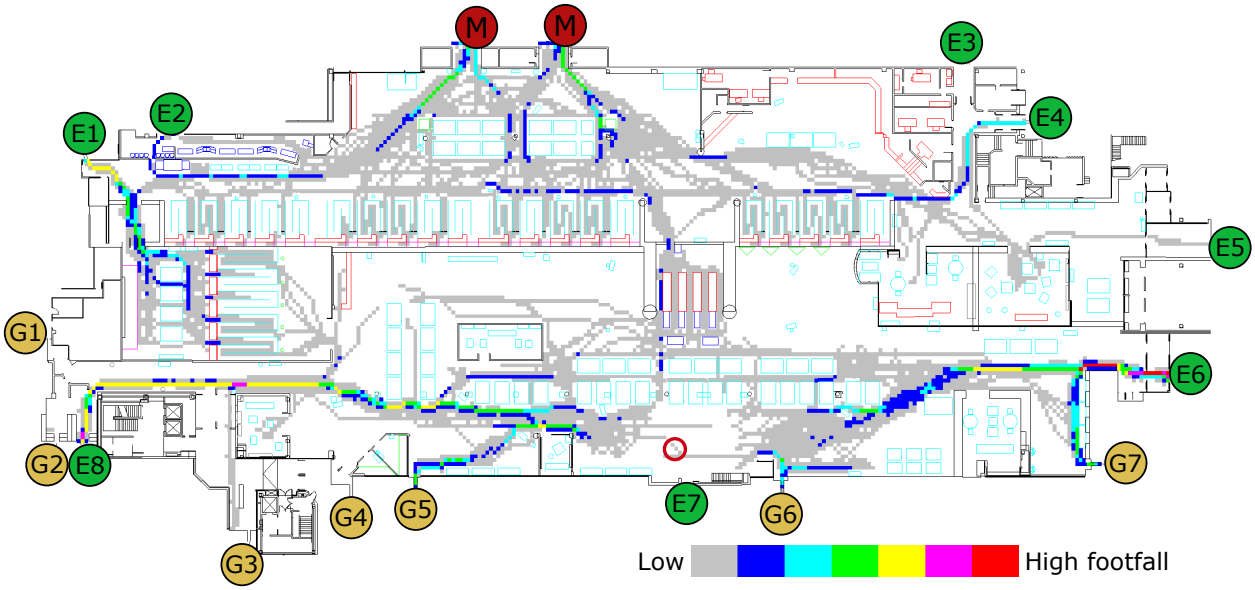


Figure D.24: Secure an area: Group 5, Run 2, footfall contour

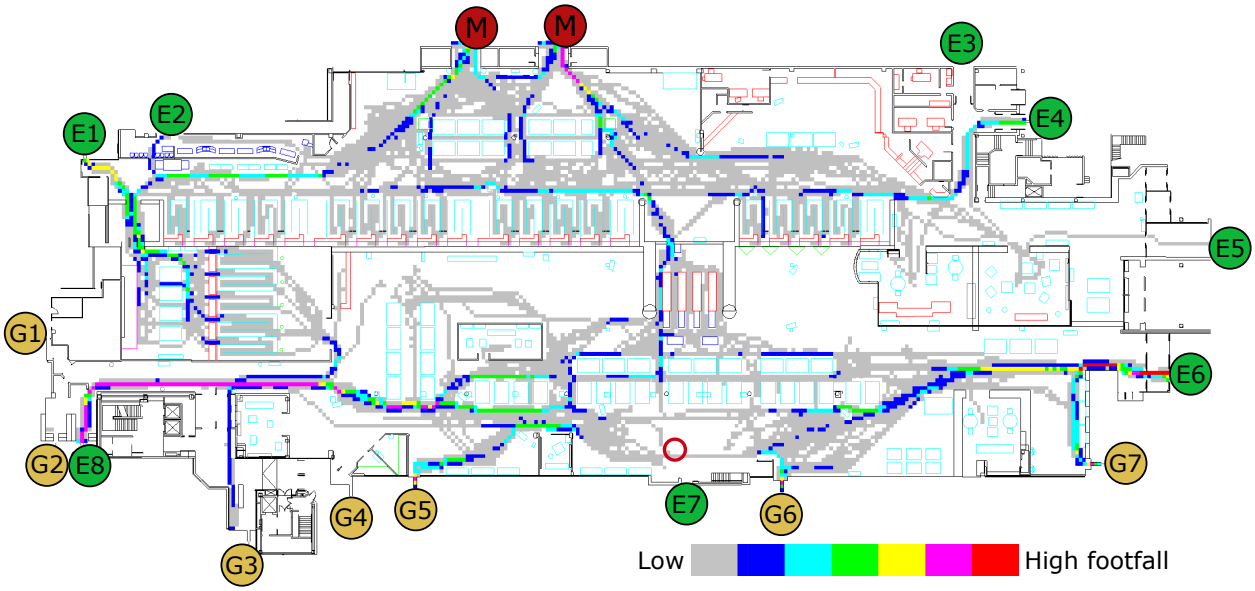


Figure D.25: Secure an area: Group 5, Run 3, footfall contour

Appendix E

SMOKE FIGURES USED IN THE QUESTIONNAIRE

All figures used in the smoke visibility tests are recorded here at various distances. The participants were shown these figures through smoke. The figures are arranged in the order they were shown to the participants. Each participant had full control over the smoke visibility. The program showed each participant a K value, which indicates the *light extinction coefficient*. The participants were to record the shown K value at the point when the object was barely visible. Smoke visibility was controlled with a mouse-wheel at increments of 0.01 with a minimum value of 0.00 and maximum value of 10.00 allowing for a total of 1000 levels of granularity. Visibility distance for the optical density level is recorded with a variable V, which is based on the research by Jin [18]. Jin provides the following equation in his paper [18]:

$$\sigma V = const \tag{E.1}$$

Where σ is the light extinction coefficient (i.e. the K value recorded by participants), V is the visibility distance in metres, and where $const = B$ and B indicates the level of visibility of the observed object. Non-illuminated signs have a visibility value, B, between 2 and 4. Therefore, for each smoke visibility test a screenshot has been recorded for the minimum visibility, B, value of 2 as well as the maximum value of 4 and the middle value of 3 (i.e. values 2, 3 and 4). Screenshots without smoke are also provided to show the

appearance of the object without smoke. To remain consistent to the rest of the thesis a term a is used for the remainder of this appendix to represent light extinction coefficient instead of K or σ. The relationship between light extinction coefficient a and optical density D is: $D = a/2.303$ [178, 179] (i.e. the relationship between natural logarithm and base 10 logarithm, $ln(10)$). The visibility distances for each B value in this appendix is calculated with the following Equation:

$$V = \frac{B}{a} \tag{E.2}$$

Where V is the visibility distance in metres, a is the light extinction coefficient and B is the visibility value of the object. All screenshots are taken at the first visibility level when the object should be visible with the accuracy of 0.01 step increments.

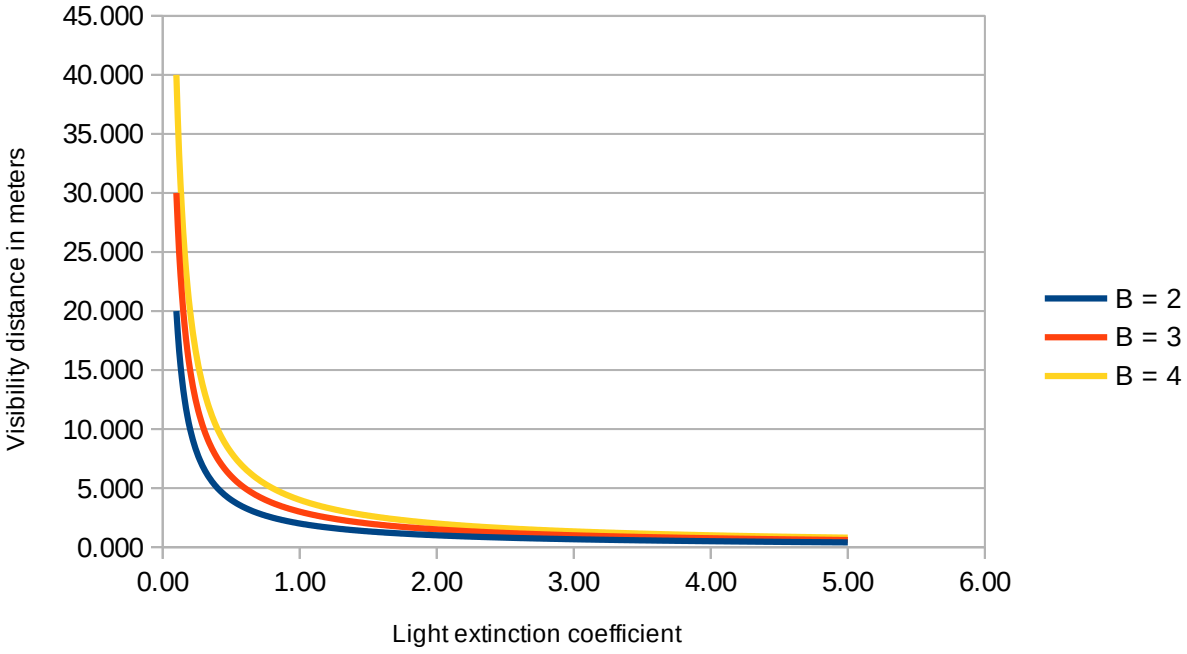
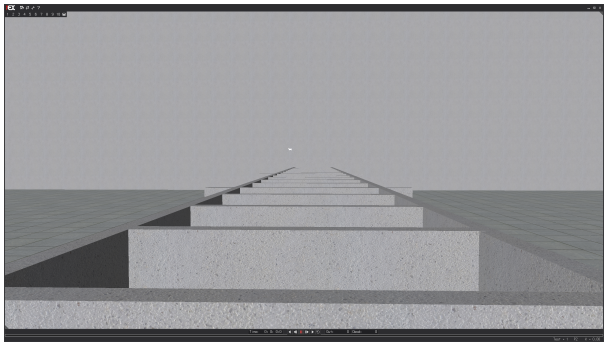
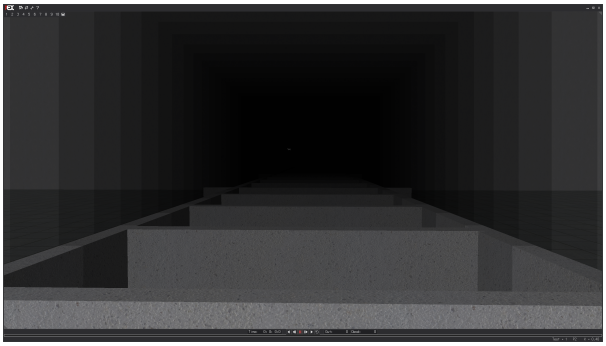


Figure E.1: Relationship between terms in Equation E.2

Figure E.1 shows the relationship between terms in Equation E.2. It shows the first half of the light extinction coefficient range available to the participants, except the first 10 values which range from 0.00 and 0.09 as they would negatively affect the scale of the graph.



(a) No smoke



(b) B = 2, a = 0.40, V = 5.000m



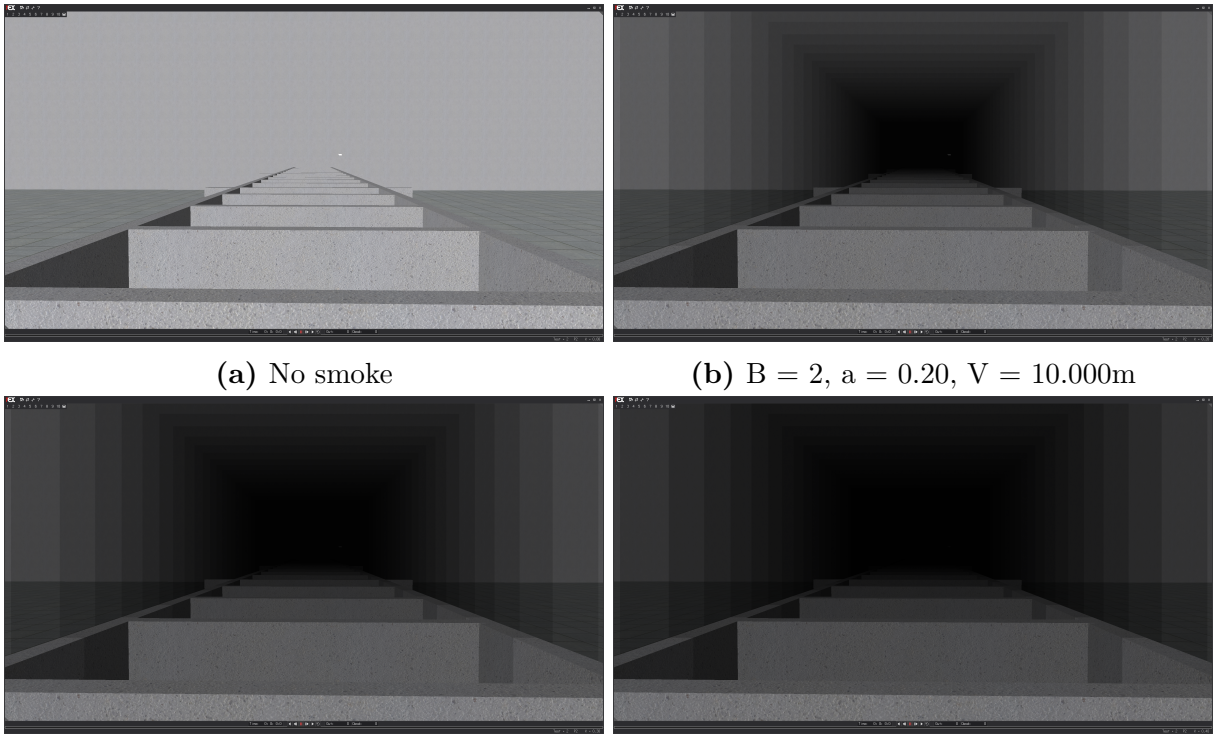
(c) B = 3, a = 0.60, V = 5.000m



(d) B = 4, a = 0.80, V = 5.000m

Figure E.2: 5cm cow figure, 5m distance, 2 particles per metre

Figure E.2 is the first figure shown to the participant in the smoke visibility exercise.



(a) No smoke

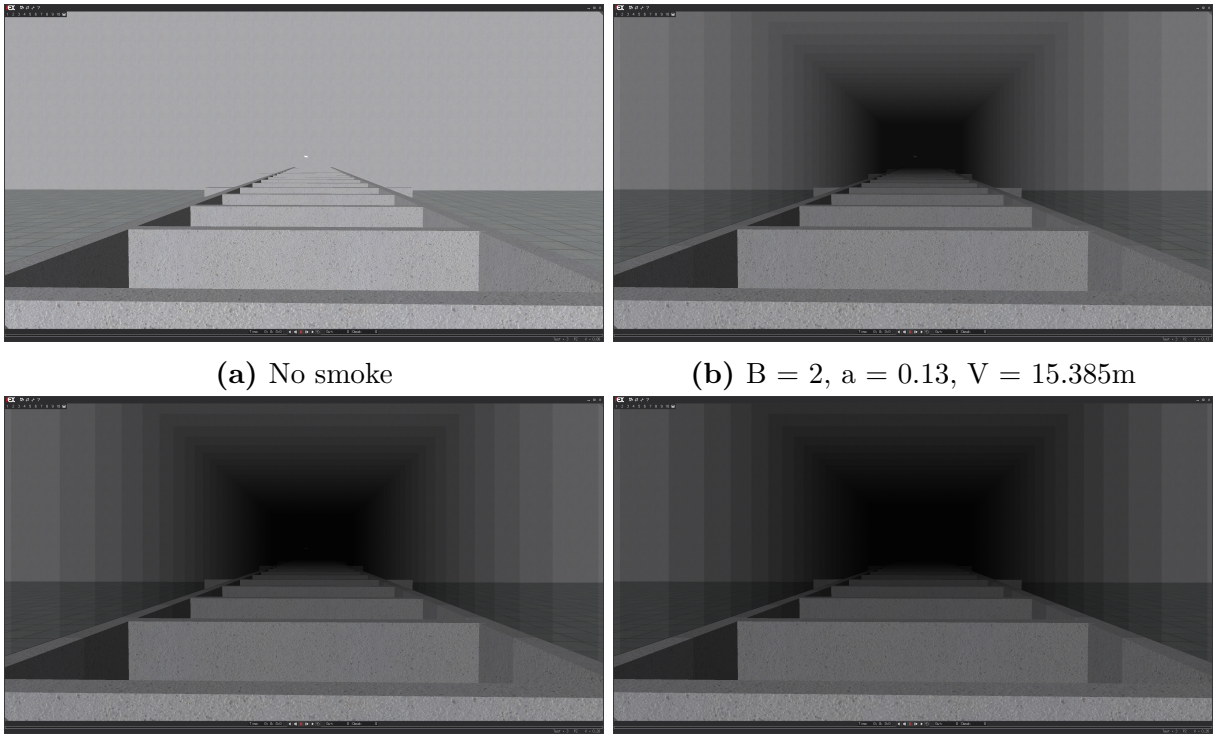
(b) B = 2, a = 0.20, V = 10.000m

(c) B = 3, a = 0.40, V = 10.000m

(d) B = 4, a = 0.40, V = 10.000m

Figure E.3: 10cm cow figure, 10m distance, 2 particles per metre

Figure E.3 is the second figure shown to the participant in the smoke visibility exercise.



(a) No smoke

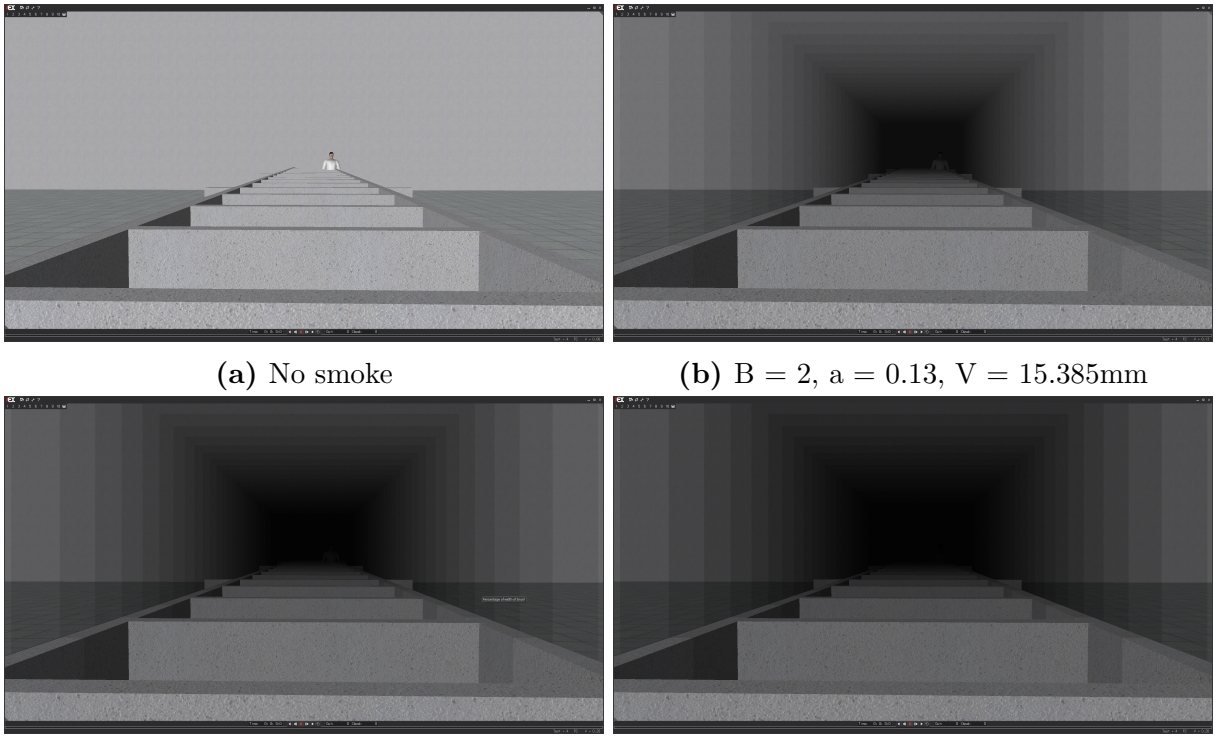
(b) B = 2, a = 0.13, V = 15.385m

(c) B = 3, a = 0.20, V = 15.000m

(d) B = 4, a = 0.26, V = 15.385m

Figure E.4: 15cm cow figure, 15m distance, 2 particles per metre

Figure E.4 is the third figure shown to the participant in the smoke visibility exercise.



(a) No smoke

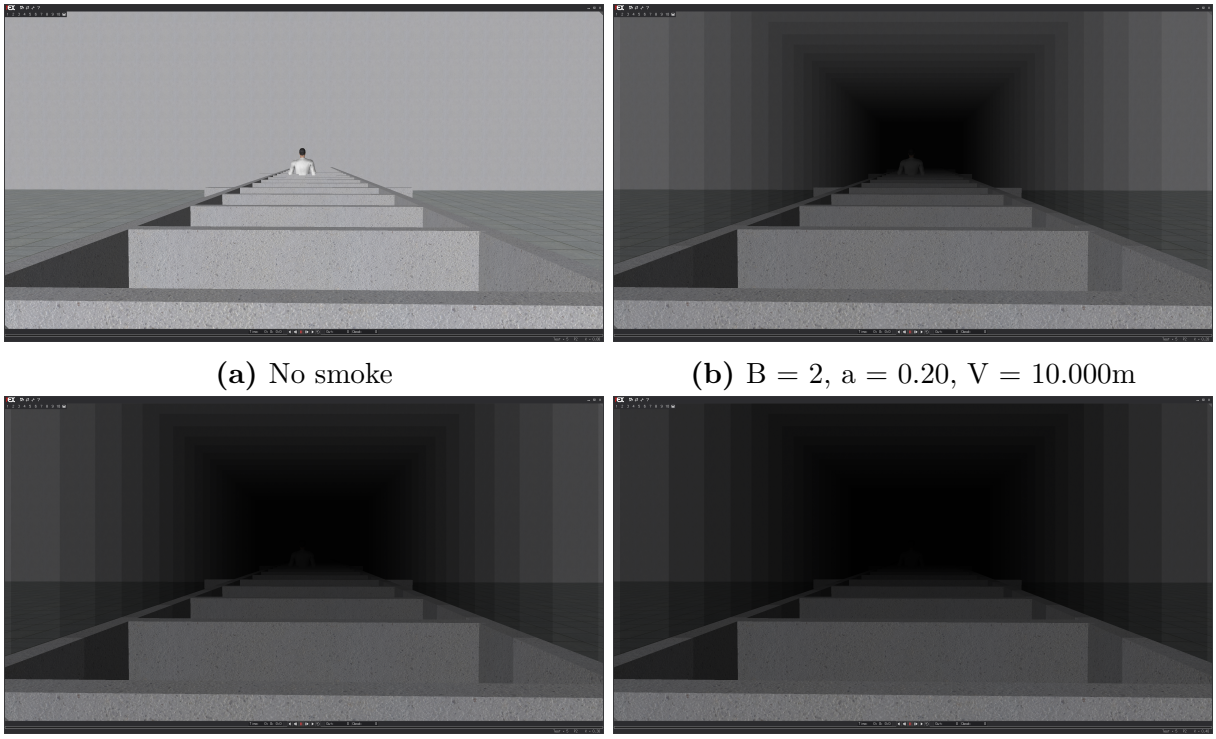
(b) B = 2, a = 0.13, V = 15.385mm

(c) B = 3, a = 0.20, V = 15.000m

(d) B = 4, a = 0.26, V = 15.385m

Figure E.5: Agent figure, 15m distance, 2 particles per metre

Figure E.5 is the fourth figure shown to the participant in the smoke visibility exercise.



(a) No smoke

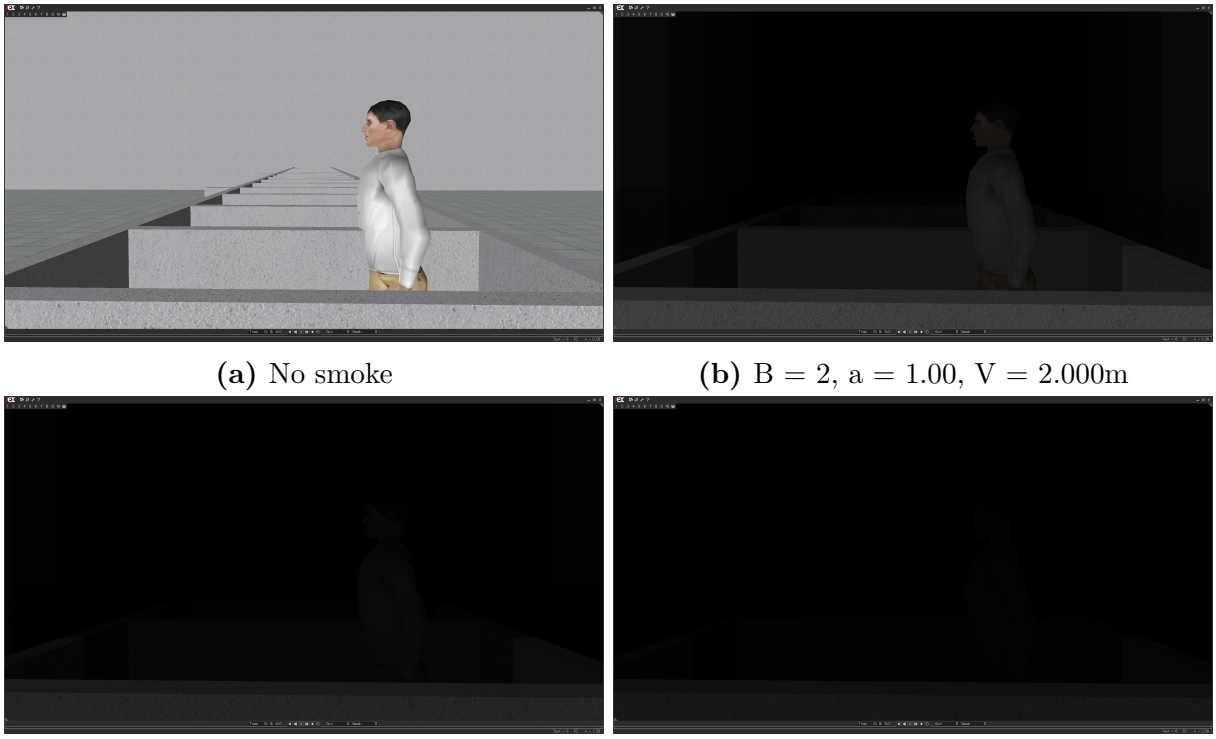
(b) B = 2, a = 0.20, V = 10.000m

(c) B = 3, a = 0.30, V = 10.000m

(d) B = 4, a = 0.40, V = 10.000m

Figure E.6: Agent figure, 10m distance, 2 particles per metre

Figure E.6 is the fifth figure shown to the participant in the smoke visibility exercise.



(a) No smoke

(b) B = 2, a = 1.00, V = 2.000m

(c) B = 3, a = 1.50, V = 2.000m

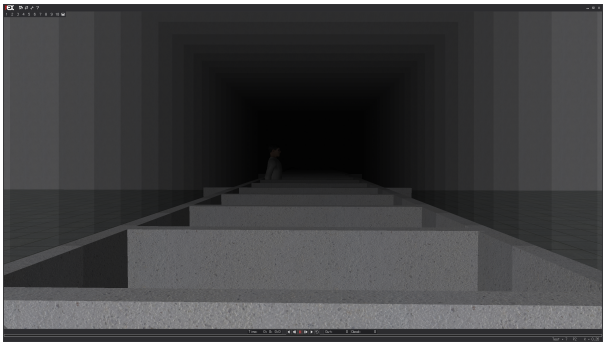
(d) B = 4, a = 2.00, V = 2.000m

Figure E.7: Agent figure, 2m distance, 2 particles per metre

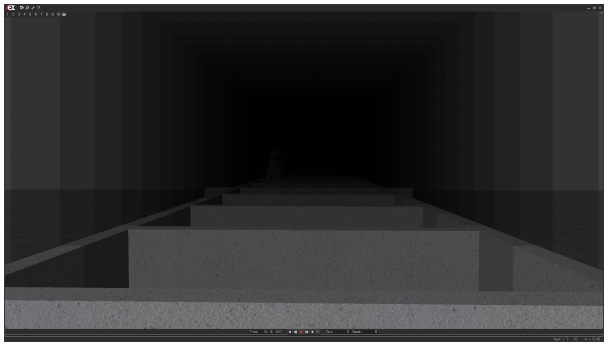
Figure E.7 is the sixth figure shown to the participant in the smoke visibility exercise.



(a) No smoke



(b) B = 2, a = 0.26, V = 7.692m



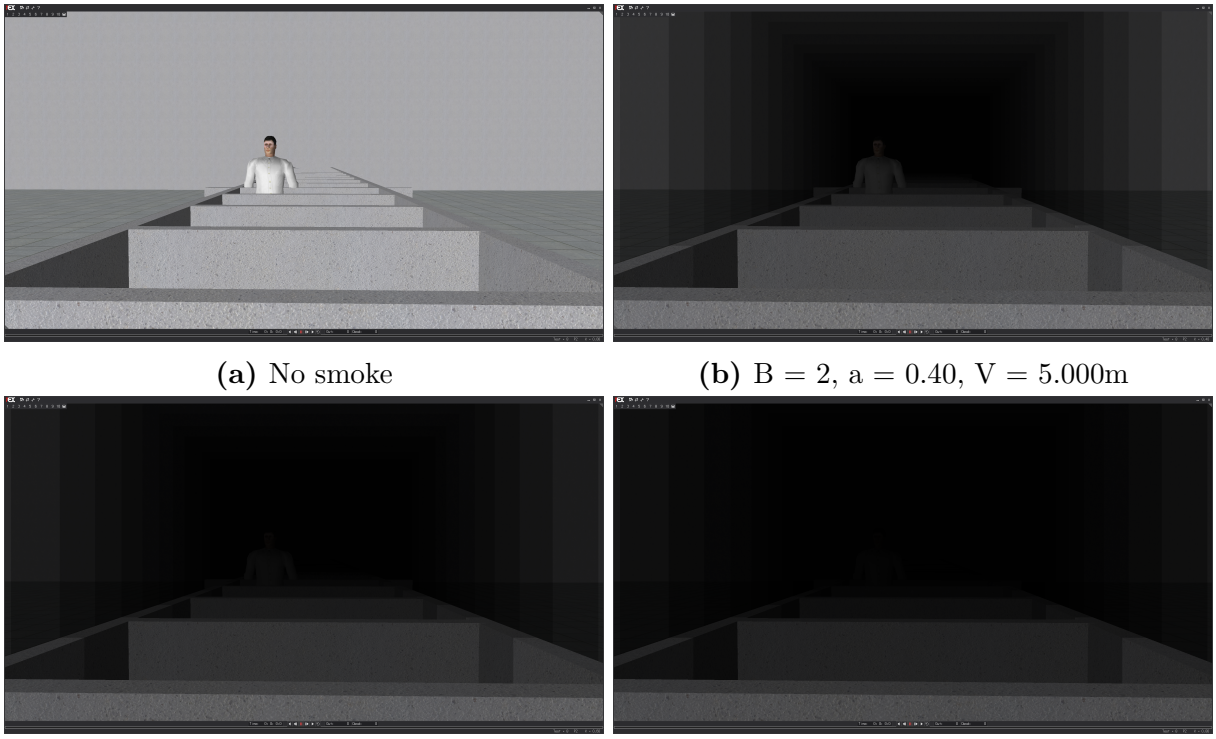
(c) B = 3, a = 0.40, V = 7.500m



(d) B = 4, a = 0.53, V = 7.547m

Figure E.8: Agent figure, 7.5m distance, 2 particles per metre

Figure E.8 is the seventh figure shown to the participant in the smoke visibility exercise.



(a) No smoke

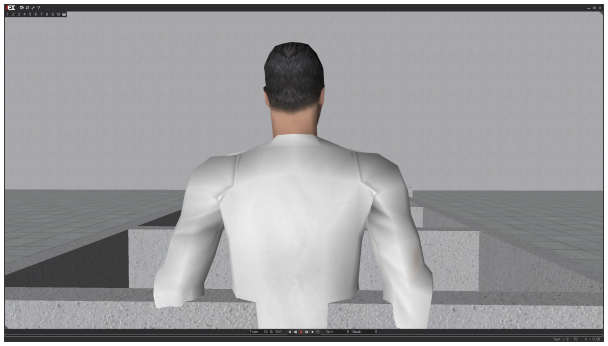
(b) B = 2, a = 0.40, V = 5.000m

(c) B = 3, a = 0.60, V = 5.000m

(d) B = 4, a = 0.80, V = 5.000m

Figure E.9: Agent figure, 5m distance, 2 particles per metre

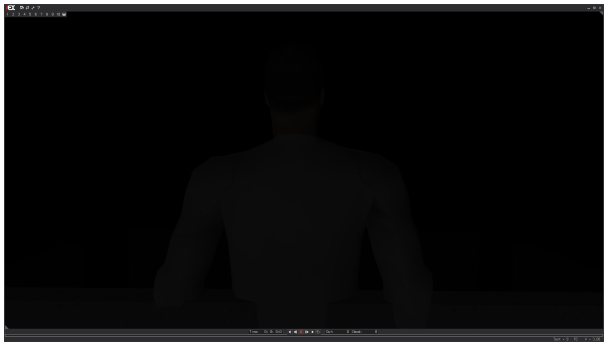
Figure E.9 is the eighth figure shown to the participant in the smoke visibility exercise.



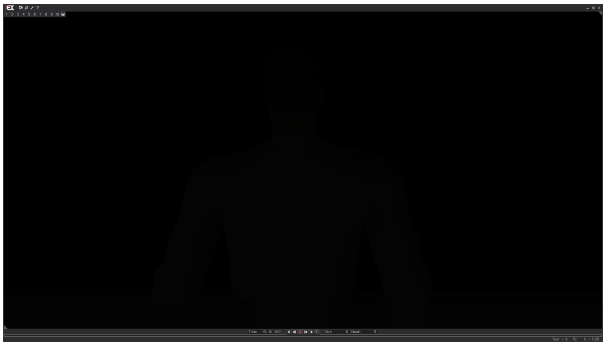
(a) No smoke



(b) B = 2, a = 2.00, V = 1.000m



(c) B = 3, a = 3.00, V = 1.000m



(d) B = 4, a = 4.00, V = 1.000m

Figure E.10: Agent figure, 1m distance, 2 particles per metre

Figure E.10 is the ninth and final figure shown to the participant in the smoke visibility exercise.

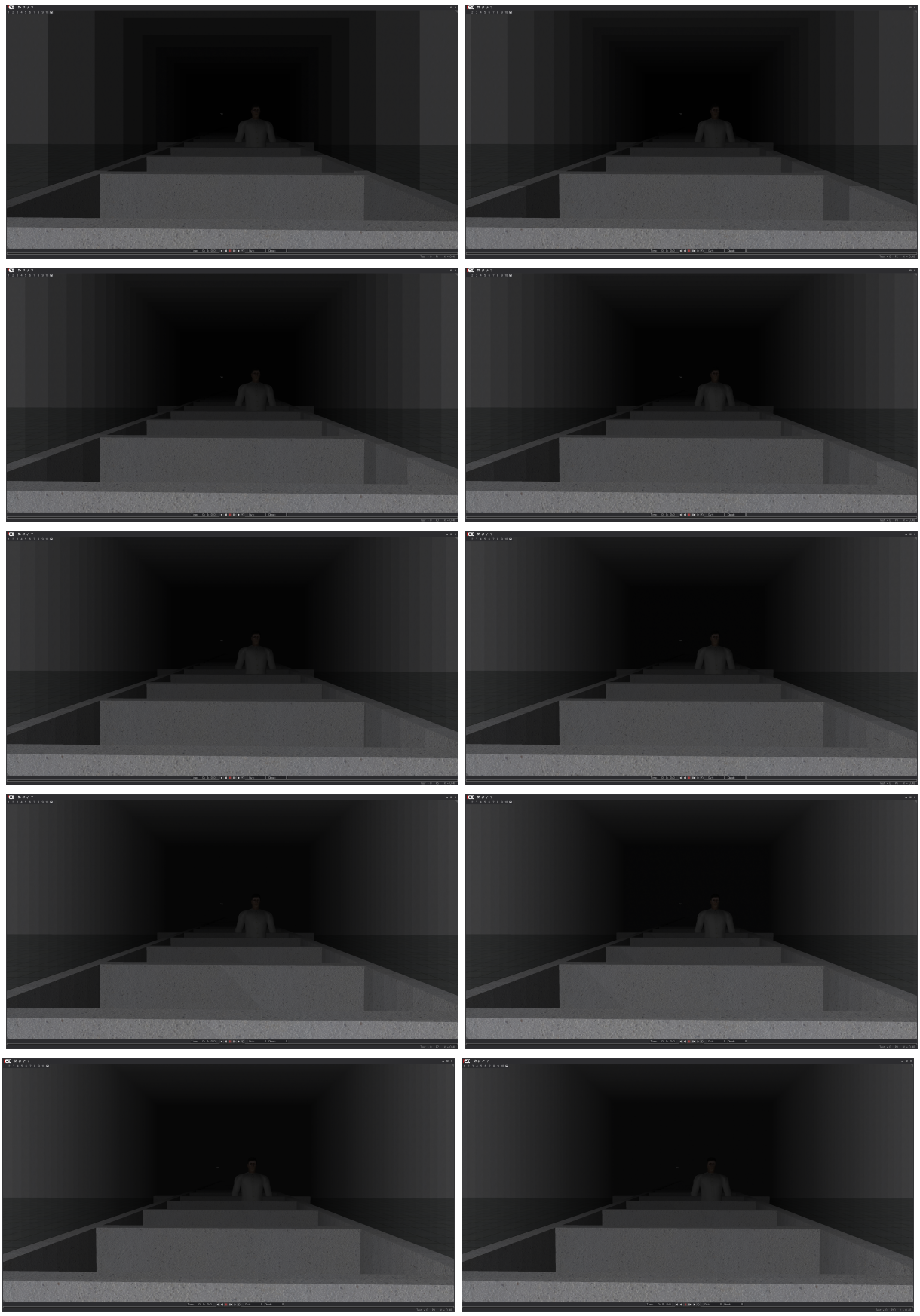


Figure E.11: 5cm cow figure and person shape agent figure at 5m distance, 0.40 light extinction coefficient, varying number of particles per metre. Left hand images have 1, 3, 5, 7 and 9 particles per metre respectively from top to bottom. Right hand images have 2, 4, 6, 8 and 10 particles per metre respectively from top to bottom.

Figure E.11 shows 5cm cow figure and a person shape (1.83cm tall) agent figure at 5m distance. The light extinction coefficient is 0.40, which is equivalent to 5m visibility for $B = 2$ value object, 7.5m visibility for $B = 3$ value object, 10m visibility for $B = 4$ value object. It is clear from the screenshots in Figure E.11 that the visibility stays constant even when the number of particles is varied per metre distance.

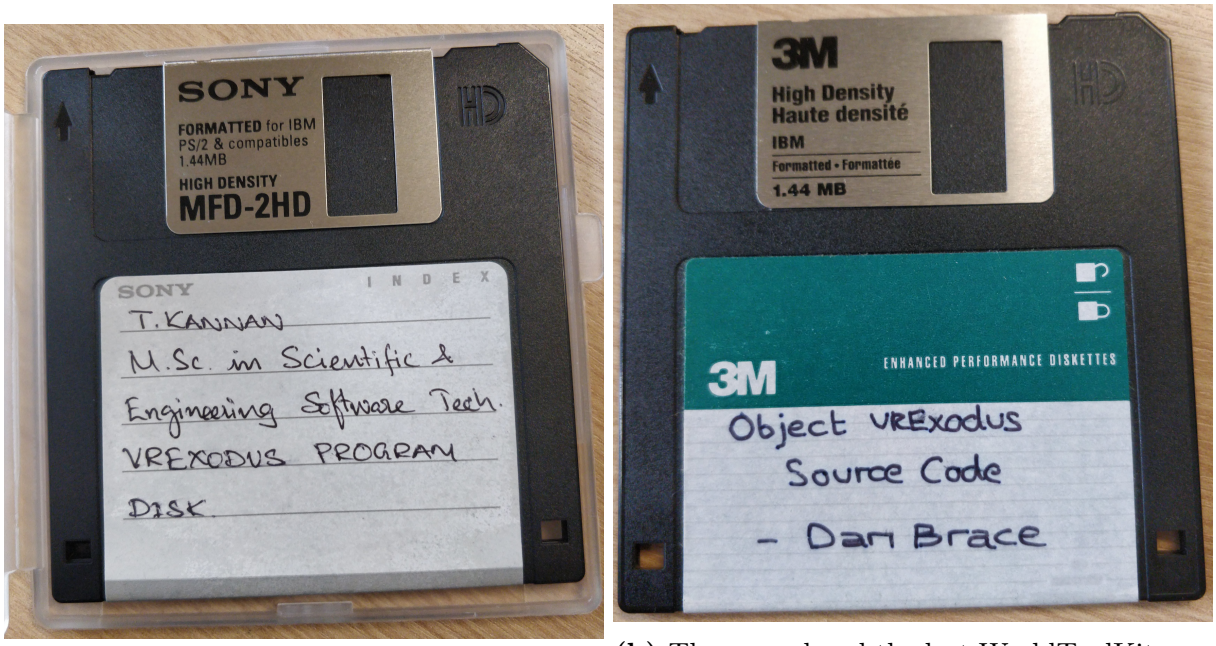
Appendix F

VREXODUS DESCRIPTION

F.1 History of vrEXODUS

The first version of vrEXODUS software was made by Kannan in October 1995 [211] (see Figure F.1a). This first version was completed as part of a MSc project, supervised by Lawrence and Galea. The first version of vrEXODUS was written in Borland C++ 4.0 and used a 3D engine called WorldToolKit 2.0. It was a 32-bit Windows application. The first version of vrEXODUS did not use hardware acceleration on 3D rendering and did not include any smoke representation. The 3D render was capable of showing agents, walls and chairs. Interestingly, Kannan lists in his future work section a suggestion for a first person camera. The first version of vrEXODUS focused on aircraft geometries, which are typically also smaller than buildings, cruise ships or trains. The minimum requirements were: MS-DOS version 5.0 and Microsoft Windows 3.1/NT, 16 MB of RAM, 486DX PC, 200 MB HDD and VGA or SVGA graphics that supports at least 16 colour graphics driver.

After Kannan's version there were two versions of vrEXODUS done in parallel in 1996 by Tan [212] and Brace [213] (see Figure F.1b). These were final year computer science projects in collaboration with FSEG.



(a) The first vrEXODUS ever written in 1995

(b) The second and the last WorldToolKit version vrEXODUS in 1996

Figure F.1: The last known copies of the first 2 vrEXODUS versions written in WorldToolKit for Windows version 2.0

Brace continued on Kannan's work, focusing on rendering aircraft fuselages with WorldToolKit for Windows. He achieved real-time performance by moving from 486-DX 50MHz ISA SVGA PC (at 2.5 frames/sec) to Pentium 66MHz PCI Local Bus PC (at 11 frames/sec). These were rendered on 640x480 resolution. Brace's work focused on moving Kannan's work into an Object Oriented code structure. This version of WorldToolKit was also a software renderer.

Tan worked in parallel to Brace, but built vrEXODUS using Superscape VRT 3D engine. Tan compared [212] WorldToolKit to Superscape VRT explaining Superscape VRT was able to leverage a hardware graphics card to improve the rendering performance, but did not require it. Other major difference to WorldToolKit was that Superscape VRT can be used without writing any programming code. However, Superscape VRT does provide an option to use Superscape Control Language (SCL) when needing to control all attributes of an object, which is based on C language. Superscape VRT used DOS environment rather than Windows. Tan's version of vrEXODUS was capable of rendering 270 agents [212], assuming no other objects were rendered. This was an inherent limitation of Superscape VRT software, only capable of rendering 32,676 objects, where the human model built by Tan used 122

objects from the total object budget. Tan proposed an LOD (Level Of Detail) optimisation of reducing the number of objects that make a human model when the model is further than a certain distance. It was not implemented, but his proposed low detail human model would consist of 58 objects, increasing the maximum number of drawn agents to 564. Tan's Superscape VRT version of vrEXODUS was also focused on aircraft fuselages. He developed 2 versions, a *run* version and a *walk* version, and depending on whether the agents needed to *walk* or *run*, then the appropriate version was used. Tan's version also consisted of only chairs, walls and agents and otherwise had very similar capabilities to Brace's version.

Brace's version was abandoned in favour of Tan's version. Tan's version was extended by Cooney in 2000 [154]. This work was done as part of a MSc project. However, by continuing to use the Superscape VRT software, the existing limitations of rendering objects continued to apply to Cooney's version of vrEXODUS. There were some discrepancies between EXODUS and vrEXODUS implementations. Space is represented in EXODUS as 1m = 30 units, where in Superscape VRT 1m = 10,000 units. EXODUS also moves the simulation forward at the rate of 12 time steps per second. Cooney's version of vrEXODUS continued to support chair, wall and agent models, but also introduced support for floors, exits and stairs (including landings). Each agent received an LOD version, which Tan had proposed as future work in his thesis. This increased the maximum number of agents to 5,000 with the lowest detailed model, assuming no other objects were displayed. Cooney also introduced new animations to extend Tan's *run* and *walk*: move up and down stairs, move onto and off stair case and death move. While vrEXODUS was not capable of rendering smoke, it was able to display a letter (H or T respectively) above an agent if it was subjected to serious heat (FIH) or toxicity (FIN) levels. Fatalities and incapacitated person would also leave a mark on the floor with the letters H and/or T. The significant difference in Cooney's version to previous versions of vrEXODUS is the capability to represent a building instead of an aircraft fuselage. Furthermore, Cooney included the capability to represent multi story buildings with stair cases.

During his PhD, Ewer [214] introduced SMARTFIRE CFD fire simulation software into the suite of software developed by FSEG. The SMARTFIRE software was a CFD simulation model. EXODUS received the capability to use the completed simulation files by SMARTFIRE. This also motivated the need for vrEXODUS to represent smoke from completed

EXODUS simulation files.

Cooney continued to develop vrEXODUS by adding a new capability to represent smoke as triangles drawn on the floor or as translucent 3D boxes filling up a smoke zone. Due to the hard object count limit of Superscape VRT, vrEXODUS was not able to represent large structures such as the World Trade Center incident in 2001 and was limited to only show a couple of floors of the building. vrEXODUS had to be redeveloped if the user wanted to view a different part of the building.

There was an inevitable need for a modernised version of vrEXODUS, which would use graphics cards for hardware acceleration in rendering and would lift the maximum object limit imposed by Superscape VRT. This new version was initially started by Webber in 2006 as an industrial placement student in FSEG. It was written in C++ and using Direct3D version 9 (the Windows XP version). However, it was never released and was later abandoned as an incomplete project. Webber proceeded to write a final year project after the industrial placement year on visualising smoke of a SMARTFIRE CFD model [215].

F.1.1 Current state and the limitations of vrEXODUS

The current released version of vrEXODUS completed by Haasanen in 2008 represented an entire new development based on the assumption that it was compatible with the previous version of vrEXODUS and had at least the same level of functionality with enhanced performance. The imposed limitations on the chosen graphics library and the programming language were set by FSEG team. The requirements were a DirectX 9 application (including Direct3D and DirectInput), written in C++ language for Microsoft Windows XP operating system. The initial version was made for buildingEXODUS, where the most pressing need was to be able to represent large structures with large populations. This new version of vrEXODUS eventually had all the previous capabilities, with none of the limitations (e.g. hard object count limit), with an added performance of hardware accelerated rendering and new features such as native video recording in WMV format. The new vrEXODUS also supports all the EXODUS model versions, agent actions and structural objects. As a performance test case a Trafalgar Square in London with over 120,000 animated agents renders over 24 frames

per second on a first generation Intel i7 960 with NVIDIA GTX 480 graphics card. There were a lot of performance improvements, however, the 3D models were unchanged from the MSc project by Cooney [154].

The current public release version of vrEXODUS is efficient but it is unsuitable for the IRIS system due to its cartoon style graphics (see the comparison between the old and new graphics in Chapter 4), the smoke visibility representation (i.e. colour gradient of the smoke is mapped to a visibility graph), and the absence of networking capabilities and multi-threading. Furthermore, it is based on an old version (version 9) of Direct3D graphics library, which was superseded by a new incompatible version in 2007 (see further discussion in Chapter 4). The change of graphics library in vrEXODUS is necessary and is further discussed in Chapter 4. As the release version of vrEXODUS software is built around DirectX library from Microsoft with high coupling (i.e. a single threaded application calling parts of the DirectX library in most subroutines), it is necessary to redevelop a new version from scratch to use a different library.