UNIVERSITY OF SHEFFIELD

DOCTORAL THESIS

# Constraint-Based Simulation of Virtual Crowds

*Author:*
John CHARLTON

*Supervisors:*
Prof. Paul RICHMOND
Dr. Steve MADDOCK

*A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy*

*in the*

Visual Computing Research Group
Department of Computer Science

14 September, 2023


The University Of Sheffield.

*"The intelligence of that creature known as a crowd is the square root of the number of people in it."*

Terry Pratchett, Jingo

UNIVERSITY OF SHEFFIELD

# *Abstract*

Faculty of Engineering
Department of Computer Science

Doctor of Philosophy

**Constraint-Based Simulation of Virtual Crowds**

by John CHARLTON

Central to simulating pedestrian crowds is their motion and behaviour. It is required to understand how pedestrians move to simulate and predict scenarios with crowds of people. Pedestrian behaviours enhance the range of motions people can demonstrate, resulting in greater variety, believability, and accuracy. Models with complex computations and motion have difficulty in being extended with additional behaviours. This is because the structure of these models are not designed in a way that is generally compatible with collision avoidance behaviours. To address this issue, this thesis will research a possible pedestrian model that can simulate collision response with a wide range of additional behaviours. The model will do so by using constraints, a limit on the velocity of a person's movement. The proposed model will use constraints as the core computation. By describing behaviours in terms of constraints, these behaviours can be combined with the proposed model.

Pedestrian simulations strike a balance between model complexity and runtime speed. Some models focus entirely on the complexity and accuracy of people, while other models focus on creating believable yet lightweight and performant simulations. Believable crowds look realistic to human observation, but do not match up to numerical analysis under scrutiny. The larger the population, and the more complex the motion of people, the slower the simulation will run. One route for improving performance of software is by using Graphical Processing Units (GPUs). GPUs are devices with theoretical performance that far outperforms equivalent multi-core CPUs. Research literature tends to focus on either the accuracy, or the performance optimisations of pedestrian crowd simulations. This suggests that there is opportunity to create more accurate models that run relatively quickly. Real time is a useful measure of model runtime. A simulation that runs in real time can be interactive and respond live to user input. By increasing the performance of the model, larger and more complex models can be simulated. This in turn increases the range of applications the model can represent. This thesis will develop a performant pedestrian simulation that runs in real time. It will explore how suitable the model is for GPU acceleration, and what performance gains can be obtained by implementing the model on the GPU.

# *Acknowledgements*

To my supervisors Prof. Paul Richmond and Dr. Steve Maddock.

Paul, for your kind words of encouragement and support throughout. For your endless enthusiasm, guidance, and patience.

Steve, for your insights, help in working through problems, and motivation to keep me excited.

Prof. David Fletcher, for making my time while working on RateSetter and the PTI more pleasant and enjoyable, and protecting me from the terrifying world of industry.

Mom, Dad, and my family, for your unconditional love and support.

And Jezabel, for always being there, and helping me through the toughest spots.

## Acknowledgements

This research was supported by the Transport Systems Catapult and the EPSRC grant “Accelerating Scientific Discovery with Accelerated Computing" (grant number EP/N018869/1)

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Background

Pedestrian simulations can be used to create virtual crowds of people moving around environments. They can range in scale from singular people to whole cities and can range in complexity from simple collision response to intelligent behaviour models. A pedestrian simulation is typically designed with a model in which each person has stated goals. Goals can be as simple as getting from point A to B, or more complex such as following a tour group or evacuating a collapsing building.

Simulations of pedestrians are used for a variety of purposes. They are used in architecture, visual graphics and gaming, and event planning. For example, in the field of architecture pedestrian simulations are used to predict how people will use a building before it is constructed. The use of simulations allows the discovery of any potential points of congestion aiding in the building design process [186]. Simulations of pedestrians are frequently used within media productions, both interactive (like video games) and non-interactive (like films). In these cases people are simulated to add a sense of realism to a virtual world. An example would be the Lord of the Rings films, where virtual people populate massive battles and cityscapes [197]. Within the context of event planning, large events have potentially many thousands of people in high density environments, such as the Hajj or large sports events like the Olympics [92]. Pedestrian simulation can be used to explore emergency response scenarios to provide understanding of how to limit dangers that may occur which helps protect people's safety.

Pedestrian simulations are important tools as it can often be impractical to study real pedestrian crowds. Experiments with hundreds of people are hard to control, are logistically difficult, and are prohibitively expensive. There are additional important safety and ethical concerns when attempting to study high-density crowds or evacuation measures. Simulated crowds can be used in lieu of real crowds to aid understanding. Pedestrian simulations can be used to discover preventative measures or policy impacting on safety, comfort and user experience prior to real crowds of people using infrastructure facilities. The study of virtual crowds and modelling of pedestrian behaviour can lead to a deeper understanding of real crowds, providing insight into the psychology [53], biomechanics [113, 126], and sociology of people [122].

A simulation is a computational experiment, and within pedestrian simulations there are a wide range of approaches to pedestrian modelling which vary in complexity and performance. They can range from fast large crowd approaches [221] to slow precise individual models [189]. Interactive virtual crowds, such as in video games, focus on modelling approaches which can be simulated efficiently to provide believable motion. Although computationally efficient, these crowd models can sacrifice accuracy, reducing their applicability to crowd studies. In contrast, studies of evacuation behaviour attempt to model collision response motion as accurately as possible. In addition, each person can display different characteristics from others which can alter the number of major injuries in emergency incidents, or their maximum walking speed [87, 82]. The accuracy and complexity of models can have a significant and detrimental impact on the model's performance in simulation.

Interactive crowd simulations need to run in real-time, as opposed to offline simulations which may take many hours to compute solutions. Real-time simulations are those that simulate the model as fast, or faster, than the time that advances within the model. Offline models are less concerned with the performance of the model, and more about the correctness of it. As the field of crowd dynamics has developed, the distinction between real-time and offline models has grown. Offline models develop further complexity, while real-time models perform faster, both due to hardware advances and due to performance optimisations within their software implementations. This has left an opportunity to develop models within the middle-ground: models which perform complex calculations and run quickly [59]. These types of models are useful in real-time predictors, such as warning if a crowd might become too dense for safety [174], or in first-person interactive media, where the whole crowd should look compelling and nearby people to the user are believable [208].

There are no behavioural models which are generally applicable to the full range of dynamics exhibited by real crowds. Behavioural models vary greatly depending on the desired application and area of study. Some models are designed to cover a wide range of expected emergent behaviours in particular circumstances, e.g. a low-density crowd simulation [88]. Other models are for modelling specifically observed behaviours [67, 111]. Crowd models are developed to address particular observational phenomena, and there is no model which stands out as being suitable for representing a broad variety of different crowd behaviour [59].

The crowd scenario being replicated and the purpose of the proposed simulation determines which models are suitable to use. Different models will be suited to different scenarios. A major challenge in model selection can be the difficulty in extending existing models with new specific behaviours. The addition of new modelling behaviours can greatly affect the existing model behaviour, which can affect the overall accuracy of the simulated results.. Duives et al. [59] compare how various model approaches handle different scenarios. They find that simulation models are highly dependent on the behaviour to be simulated. They also find that most types of models, excluding cellular automata, struggle to simulate new infrastructure. Cellular automata are shown to provide flexibility as they are rule-based. The addition of new behaviours is incorporated by describing them as rules, which has no impact on the original computation of cellular automata's behaviour. Many other types of models, including continuous-space models, which are important in high-density scenarios, do not have an easy way to incorporate additional behaviour. The approach of cellular automata, which uses a consistent method of describing rules, inspires a potential framework for non rule-based models. This approach should involve describing behaviours in an emergent way that can be incorporated into the

remainder of the model. The proposed approach, which will be detailed within the thesis, to incorporate additional behaviours is by describing behaviours as mathematical constraints.

Constraints are used within simulations to limit the motion of virtual people. An example is a constraint on maximum velocity that limits how fast a person can move. This constraint arises from a person's leg stride and fitness. A desperate person who wants to reach their goal as quickly as possible is constrained to travel below their maximum velocity. Another example is during evacuation, people want to leave the danger area, but are constrained by how fast they can move.

Real people in crowds are constrained: they are limited in the motions they can carry out. This can be in the form of hand-holding with another person, which limits the distance from each other. It can be in the form of biomechanics, as a person cannot immediately run in the opposite direction without slowing down. Constraints can also be less rigid. An example is walking with a tour group, constrained to walk close enough to hear the guide. Such a constraint is not an absolute requirement, but still impacts the person by constraining them from being too far away. Because such a wide variety of concepts can be described as constraints, it follows that a wide variety of behaviour can be described as constraints. This makes a constraint-based approach appealing for its potential capability to be easily extended with additional short-term behaviours.

Within pedestrian models the notion of constraints is usually abstracted within the calculations. This is done to simplify the model concepts as well as the calculation. For example, force-based models constrain the maximum amount of force and acceleration a person experiences, and this is accounted for within the force calculation. Models which intrinsically handle mathematical constraints are found within the velocity-based family of models. Velocity-based models consider geometric shapes and constraints as limitations on the available velocity.

## 1.2 Aims and Research Statement

The aim of this thesis is to investigate a constraint-based approach to pedestrian modelling. It will explore how a constraint-based approach can be easily extended with additional emergent behaviours. The model will target real time simulations, with the aim of simulating as many people in real time as possible with accurate motion. Existing highly accurate models have difficulty in being extended with general emergent behaviours, and need specific alterations for each combination of behaviour. This thesis will investigate how to extend the concept of a constraint to include more complex pedestrian motion in the form of additional emergent behaviours. Particular examples of trailing luggage, groups, interactions with autonomous vehicles, and platform-train interface will be used to explore how suitable such behaviours are to a constraint-based approach. Such behaviours should be able to be simulated while maintaining accurate pedestrian collision response.

Real time simulations are limited in how many people they can simulate, and how many complex computations they can perform. This limits the complexity and scope of simulations. If the simulation performance improves, then more people, and more complex computations, can be performed while still being in real time. There are many ways to improve the run time of a simulation. The largest theoretical performance improvement comes from utilising GPU hardware. These devices can perform many more computations than multi-core CPU implementations. Measured performance improvement when running a GPU implementation compared

to a CPU implementation is difficult to predict prior to experimentation, but pedestrian simulations have been shown to be suitable candidates for GPU acceleration [172]. It is important that a GPU implementation makes use of data parallelism to maximise the device utilisation.

This thesis proposes an extensible constraint-based modelling approach for crowd simulations and makes the following Thesis statement:

- **A real-time crowd model using a union of constraints allows the addition of behavioural complexity without sacrificing simulation accuracy.**

The thesis statement naturally leads to the following research questions (RQ) which are addressed within this Thesis work.

RQ 1 **What modelling approach is most suitable for steering and short-term emergent human behaviours in large dense crowds?**

RQ 2 **Can constraints be used to provide a flexible modelling approach suitable to capturing a wide range of crowd collision response behaviours?**

RQ 3 **Can accuracy of interactions between individuals be maintained when adding complex behaviours?**

RQ 4 **Is a constraint based modelling approach suitable for acceleration using data parallelism on GPUs?**

RQ 5 **Can a constraint based simulation of crowds be accelerated to meet the target of real-time performance?**

RQ 1: Between the many different pedestrian simulation models, this research question will explore what their advantages and disadvantages are in the context of large dense crowds. This research question will involve a comparison of modelling approaches, and consider which are appropriate for the domain of emergent behaviours within large dense crowds. This question can be answered by contrasting between different approaches to explore their strengths and weaknesses, and drawing conclusions as to which approaches are most suitable for large dense crowds.

RQ 2: A flexible modelling approach is one which can simulate collision response with different emergent behaviours. Such an approach can be extended with additional behaviour without impacting the motion of pedestrian steering. Flexibility can be assessed through the ease of extending the model behaviours. To constitute a wide range of behaviours, the approach should be demonstrated to handle at least three different behaviours specifically, with the capability for more. Collision response behaviours are those that affect the short- and near- term motion of a person.

RQ 3: Accuracy is a measure of how well a simulation can reproduce real, observable behaviours. A model can be accurate for certain scenarios, but not for others. To answer this research question the approach needs to demonstrate quantifiably similar behaviour to observations for chosen scenarios. The kinds of interactions between individuals are those that affect the walking speed and trajectory of people nearby. A simple example of such an interaction is avoiding someone who is crossing another's path, known as collision avoidance.

RQ 4: Accelerating performance is achieved by improving the speed of simulations. GPU hardware is an appealing candidate for improving performance due to the theoretically greater computation that can be achieved. Data parallelism is a key

concept to ensure performance gains using GPUs. Data parallelism involves distributing the data equally across the computational cores of the device. It requires a balance of work for each core. The suitability is appropriate if the constraint-based approach can be implemented in a way that uses data parallelism. This involves a balanced workload and use of device memory.

RQ 5: Real time performance is achieved when a simulation computes each simulation step faster than the amount of time that was simulated. Real time, in the context of rendering, is usually quantified as running at least 30 times a second. At this rate of computation the simulation can be displayed as a smooth video running at the same refresh rate [11]. Many simulations can have performance improved by running them on more powerful computer clusters, and using cutting edge devices. However, this research question focuses on modest consumer-grade hardware, and enhancing performance through code optimisations. Achieving high performance is important to handle scalability. Larger crowds with more complex behaviours take longer to compute, and are important when considering large venues like the London Olympic Stadium which has a capacity of 80,000 people (current capacity for concerts, as of 2023 [206]).

## 1.3 Contributions

This thesis makes the following novel contributions:

C 1 Description and performance evaluation of a constraint-based simulation model. The ability to control the importance of the constraints determines which behaviours should be prioritised, permitting model behaviours without impacting the rest of the functionality of the model. This contribution will help to answer RQ 2 and RQ 3.

C 2 Description and performance evaluation of a 2D linear problem solver. Linear problems are a key computation of constraint-based problems. The performance of this computation has a large impact on the simulation model that uses it. A novel GPU implementation is proposed to maximise performance gains by solving numerous (i.e. batch) linear problems simultaneously. The solver will be shown to outperform other solvers for sufficiently large numbers of problems by considering alternative state-of-the-art approaches. This will specifically address RQ 4.

C 3 A performance evaluation and demonstration of the linear problem solver applied to the constraint-based simulation model for crowds (contribution 1). The optimisations to the linear program solver will show real-time simulation of tens of thousands of people therefore answering RQ 5.

### 1.3.1 Publications

The work in this thesis has led to the following publications:

- Simulating crowds and autonomous vehicles - J Charlton, LRM Gonzalez, S Maddock, P Richmond [39]
- Fast simulation of crowd collision avoidance - J Charlton, LRM Gonzalez, S Maddock, P Richmond [38]
- Two-dimensional batch linear programming on the GPU - J Charlton, S Maddock, P Richmond [37]

## 1.4 Definition of Terms

- **Steering, collision avoidance**. Both refer to the process of avoiding collisions, with other people or obstacles
- **Person, pedestrian**. A real human.
- **Agent**. A virtual person, one that exists in code, but not physically.
- **Crowd**. A large group of people in the same physical space at the same time. The crowd can contain smaller social groups, such as families and groups of friends, as well as individuals with their own goals [2].
- **Model**. An abstract representation of a real process. A crowd model is a way of describing crowds using mathematics, computation, and rules.
- **Simulation**. An implementation of a model. A crowd simulation is able to take a description of people (such as starting positions) and compute where they will move over time.
- **Framework**. A piece of software which can be selectively altered through user code.
- **Luggage**. Refers specifically to trailing luggage, the sort held by handle and rolled along the floor behind the owner.

## 1.5 Thesis Outline

- Chapter 2 provides fundamental background information on crowd simulations. It discusses potential modelling techniques which are suitable for potentially answering the research question of this thesis. It examines in-depth the aspects of collision avoidance, groups, and luggage as emergent behaviour. It provides information on GPUs and the architecture. It explains in detail linear programming and their solvers.
- Chapter 3 introduces the proposed modelling approach, Optimal Constraint Behaviour Model (OCBM). It explains details of the model, including the novel addition of constraint hardness. The chapter explores the model with additional behaviour implementations, specifically luggage, people in social groups, and people and autonomous vehicles sharing space. The chapter also explores the performance of the implementation. This work advances Contribution C 1.
- Chapter 4 validates the accuracy of OCBM. It does so by testing the model and its behaviours against real data. This comparison includes observations from literature, and observations from performed data collection from CCTV at train station platforms. This chapter is part of contribution C 1.
- Chapter 5 details Randomised GPU Batch (RGB), a novel linear program solver on the GPU. It is specifically designed to be performant for 2D and low-dimension programs. This solver is an important aspect in optimising the performance of OCBM. The performance of the linear program solver is tested against other solvers. To maximise the performance gains of using RGB, OCBM is implemented on the GPU, and the details of this are explained, advancing contribution C 2

- Chapter 6 demonstrates the performant OCBM on the GPU by simulating many thousands of people, with their own complex behaviours, in real time. In addition, the performance of GPU OCBM is explored and compared to the previous CPU implementation to understand the changes in speed. This chapter contributes towards Contribution C 3.
- Chapter 7 provides concluding remarks and future directions of research leading on from this.

# Chapter 2

# Literature Review

This chapter provides context and background information regarding pedestrian crowd simulations relevant to the work proposed in this thesis. In particular, it explores how to simulate pedestrian crowds, pedestrian behaviours, constraint-based mathematics including linear programming, and optimising algorithms on GPU hardware. It will discuss which modelling approaches are suitable for simulating pedestrian crowds with both speed and precision, how easily extensible they might be, and which approaches interlink multiple behaviours and steering.

This literature review will begin by explaining how pedestrian simulations function with a brief overview. It will explore which approaches are most suitable for extension to implement the ideas proposed within the thesis. It will also explore a wide range of pedestrian behaviours that can be incorporated into models. The addition of modelling new behaviours allows simulating varied interactions and scenarios, which can be very important in examining anything but the most simple of crowd dynamics. Behaviours include evacuation behaviour, carrying luggage, and responding to announcements within a store.

This chapter will also discuss how to implement pedestrian models, including the various choices of possible software. An important facet of simulations is understanding how they compare to each other, and to real crowds. This will be explored in greater detail through the section regarding model validation. The chapter will also explain details on the mathematics of linear programmes and GPU architectures. These are relevant to exploring the characteristics and performance of approaches relevant to crowd simulation in general.

Section 2.1 provides an overview of pedestrian simulations and the suitability of various approaches. Section 2.2 goes into detail, examining specifically microscopic pedestrian modelling, which is the approach deemed most suitable for further study. It describes and compares small-scale aspects of pedestrian simulations. Section 2.3 describes literature around modelling human behaviours. Section 2.4 explains about model validation and how to test the suitability of models. Validation plays an important role in understanding how accurate simulations are. Section 2.5 explains linear programming, a key computational aspect of the work within the thesis which plays a role in the performance and optimisation of the final implementation. Section 2.6 provides information on GPU hardware and architecture, which is considered as a viable approach to improve the performance of constraint solving.

## 2.1 Introduction to Pedestrian Modelling and Simulation

A pedestrian crowd model is an approach to describe how people can be represented virtually. It takes the complex occurrences of real crowds, and attempts to simplify it into a set of rules. A pedestrian crowd simulation is an implementation of the crowd model. The simulation needs to be initialised: told who the people to simulate are, and in what environment it takes place. The simulation calculates how the crowd behaves and evolves over time. The choice of model can have a big impact on how the crowd evolves. The choice of model tends to be driven by what crowd behaviours are of interest or relevance to the specific situation. For example, interest in how a single person travels through an environment will typically use a different model to one interested in how a large crowd evacuates from a building. Models may vary in complexity and also performance.

Pedestrian crowd models are broadly categorised into the scale of how they consider individuals, whether on the person scale (microscopic), crowd scale (macroscopic), or as a more intangible data representation (data-driven) [121, 12]. Each approach has varying suitability when considering different scenarios.

Pedestrian crowd models are useful for a variety of cases, including applications in architectural design for predicting human behaviour in buildings which do not yet exist. In this specific case, they can be used to ensure the building will not suffer from crowding [25] or bottlenecks. Pedestrian simulations are used in safety studies, such as transport hubs and sport stadiums where there can be large numbers of people. They can be used to ensure that behaviours that can lead to unsafe situations like very high density (of 10 people per square metre) does not occur [124]. If unsafe situations do occur, then simulations can explore how to evacuate the people safely and efficiently [220, 89]. Pedestrian simulations are useful for creating immersion in virtual environments: having believable people move around virtual worlds helps with a sense of credibility of the world [202, 155]. Simulations within virtual environments also find application in games and films [174]. There are also applications in recent developments such as autonomous cars [165], or using pedestrian simulations with camera tracking to obtain more accurate tracking of people in crowds [35], where using a simulated prediction of how people move can help track how real people are moving.

### 2.1.1 Makeup of a Pedestrian Model

Pedestrian models aim to handle three distinct levels of behaviour. These are the strategic, the long-term, and the short-term [43]. The strategic level encompasses the concept of visiting a shop and returning home, or of looking at interesting events going on around [18]. The strategic level changes greatly depending on the location being simulated. People will behave very differently if entering a train station and attempting to board a train [119], compared to socially distancing at a supermarket [159]. Various disciplines such as psychology and sociology help understand behavioural complexity at the strategic level [42].

The long-term level of behaviour is usually concerned with the path planning of movement within simulation, i.e. it describes the behaviour of how a pedestrian navigates from point A to point B within a simulated environment. Knowledge of the environment is therefore required to know how to create a planned route or path to reach the destination. Such path planning involves decomposing the environment into an understandable representation for the model which can be solved by a routing algorithm [199]. Example decompositions include Voronoi shapes [195],

a regular sized repeating grid, and irregular triangles [144]. In general, solving a decomposed representation can be performed by routing algorithms such as A* or Dijkstra, regardless of decomposition shape. More complex path planning will take into account macroscopic crowd properties to affect the path planning. An example is navigating around a large crowd [72]. In this case, when there is a large group of people, the planning approach will account for the difficulty in navigating through the crowd of people, versus alternative options such as walking around the edge of the crowd.

The short-term level of behaviour is concerned with the near-distance and near-time scale of people. It involves objects close to people and timescales of a few seconds or less. It tends to be described by physics, biomechanics, and engineering. One example is by considering continuum approaches to model people as incomprehensible flows. If the individual people are given "intelligence" such as rules and heterogeneous, self-contained properties, then it can be described with Agent-based modelling (ABM) [137].

### 2.1.2 Categorisations of Pedestrian Models

Pedestrian models can be categorised by the different approaches they use. One such method of categorising is used within Meyers et al. [43]. Three prominent classifications they use are:

- Macro- vs micro- scopic
- Rule-based vs acceleration-based vs velocity-based
- Physics-based approaches and data-driven approaches

The first categorisation, microscopic or macroscopic, varies in how the model considers individual people. Macroscopic approaches consider properties of the whole crowd and lack a sense of individuals. In contrast microscopic models consider individual information (e.g. position and velocity) for each pedestrian, and the crowd properties are emergent properties that arise from individual interactions. The second categorisation regards how the model updates and moves pedestrians. Rule-based, describes pedestrian motion as a set of rules and is used by approaches such as cellular automata [29, 62, 8, 213, 116, 27, 28]. Acceleration-based, updates the locations of people by considering the forces applied on a person and the corresponding acceleration it causes. Velocity-based is similar to acceleration based, but acts on the velocity of the person to update the positioning. The third categorisation is a more generalised approach to the simulation. For a long time only physics-based approaches were used.

It is only in recent years that data-driven approaches have developed.The physics-based approach uses a set of basic rules which are created by analysing, understanding, and exploring physical and social considerations. The rules and parameters of the model are related to observable or understandable metrics. These methods can be separated by scale into macroscopic and microscopic scales. The approaches are concerned with outputting crowd-scale behaviour. The data-driven approach revolves around supervised deep learning methods. Supervised deep learning is categorised by the training of neural networks which have more than two hidden layers and are supervised to learn from the data. Very recent advances look to combine microscopic-scale physics based models with data-driven deep learning models, which constitute a hybrid approach. These show promise in lessening the respective weaknesses in the models by drawing on strengths of the other [161]. The

perceived weaknesses of physics-based approaches include the lack of accuracy of single-person trajectory. While the weaknesses of data-driven approaches include behaviour that is not physically possible, limited to single or few people, and difficulty in predicting scenarios which are different to the training data.

There are other hybrid models within pedestrian simulations. These include when a flow rate is imposed on individual agents (hybridising micro- and macroscopic approaches) [148], Pathfinder [198], and Society of Fire Protection Engineers (SFPE) models (which combine fire models with pedestrian models) [99]. This chapter is concerned only with the data-physics hybrid approach.

Any given pedestrian model can be categorised into the approaches highlighted. Table 2.1 shows the comparison of certain models under this classification. As an example, the well-known boids model [171] is a *microscopic, acceleration-based, physics* approach relevant to simulate a range of collective behaviours observed in animals and humans. This thesis aims to explore behaviours which affect individuals, such as luggage. This means that person-scale physical models are the most relevant. For this reason, microscopic models will be explored in further detail in the following section. This remainder of this section will briefly touch on the alternative categorisations.

| Paper | Macro/Micro | Rule/Acc/Vel | Physics/Data |
|---|---|---|---|
| RVO/ORCA [16] | micro | vel | physics |
| social forces [88] | micro | acc | physics |
| Cellular Automata [29] | micro | rule | physics |
| Footstep planner [189] | micro | acc | physics |
| Social LSTM [3] | micro | - | data |

TABLE 2.1: Comparison of various steering approaches

### 2.1.3 Macroscopic Pedestrian Simulation

Macroscopic Pedestrian Simulation considers the whole crowd at an aggregated level. Not dissimilar to fluid simulations, large crowd-scale rules are defined. This approach is useful for crowd-scale properties and long timescales. These long timescales are of the order of 10 seconds at a minimum, which is too coarse for people walking from one end of a room to another, but suitable when walking the length of a street or large building. This approach lacks person-scale detail and short-term time details. It lacks the concept of individual people, and is relevant when interested in the overall crowd dynamics. Because this approach is coarse and does not consider individual-scales, it is able to simulate much larger crowds with the same computational performance. There are many different types of macro-scale models, and more in-depth reviews are available [203, 125].

Early models by Henderson [91, 90] of modelling crowds of people similarly to fluid mechanics. It focuses entirely on the macroscopic crowd and does not have considerations of individuals. One model by [148] deals with very dense, large crowds. In such a situation, interpersonal distance reduces and individual freedom of movement disappears. They argue that due to this, simplifications can be made to simulate people. They model inter-agent dynamics and simplify people to behave as fluid particles. This model is able to capture some macroscopic properties, such as crowd vortices, and simulate many hundreds of thousands of people. The downside is that it lacks accuracy of individual motions. It also fails to simulate larger avoidance of people around a large crowd, and lane formation between two

head on crowds, though these can be improved upon by incorporating aspects of other models, they explain. "Lanes" occur during head on motion of two opposing motion crowds when people will follow behind other people walking in the same direction as them.

### 2.1.4 Data-Driven Pedestrian Simulation

Pedestrian simulations based on data-driven approaches have been growing rapidly in recent years. This is in part due to the machine-learning/deep-learning boom, in turn, driven by the ready availability of compute performance. Physics-based approaches have been maturing for many years, but only recently can hardware churn through the huge amounts of data necessary to extract results.

The data-driven approach focuses on deep learning methods. This uses large amounts of data which needs to be correctly sanitised and labelled. Data-driven approaches aim to find patterns within the underlying data, which it can do without an understanding of the physical rules which govern pedestrian motion. The data-driven approach tends to be focused on single person trajectories. A trajectory is the paths a person takes through the environment. Models are able to very accurately predict and model single person motion. Data-driven approaches also focus on low densities and within a single room or corridor-scale. Work on data-driven approaches rapidly increased after the work of social-LSTM from Alahi et al. [3], which learns general human movement and predicts their future trajectories. A more in-depth discussion on data-driven approaches can be found within [121]. They detail the literature around data-driven trajectory calculations and point to the interesting future direction of hybridising data-driven with physics-based approaches together.

There are downsides to the data-driven approach. These include data-related issues. It can struggle in regions that do not have much input data. Due to the 'black box' nature of the approach, there are no simple, physical variables which can be adjusted. This is in contrast to the physics approach which has a fundamental set of parameters to describe the pedestrian. It also struggles from overfitting data, which occurs when the model fits too closely to a particular set of data, reducing its ability to fit additional data or predict future observations reliably.

### 2.1.5 Data-Physics Hybrid Pedestrian Simulation

The data-physics-hybrid approach to pedestrian simulation is an emerging field of research. The idea of the hybrid approach is to simultaneously make use of the advantages of the data and physics approaches, and limit the associated downsides.

One way of doing so is to use physics-based models to extend the available data set on which to train models on. In this scenario the downside of limited observational data availability within data driven approaches is offset by the physical models ability to generate simulated data. Another use of the hybrid approach is to use physical based models to inform the predictions made by data-driven predictions to remove those that do not follow physical rules [175]. Another example is from Rasouli et al. who join 2d pedestrian benchmarks with 3D data sets to create an accurate pedestrian crossing prediction model [169].

A recent example of the hybrid approach is [7] which embeds the social forces model (see below in section 2.2.1), allowing for a relatively small training set, and acceptable results in areas it was not trained on. This method removes some of the black-box of machine learning and provides results in human-interpretive form.

## 2.2 Microscopic models

Microscopic models use rules at the scale of individual people. Individual based modelling is analogous with agent-based modelling where the rules are defined for the people (agents), who make their own independent decisions. In microscopic models the individual people appear to act with some level of intelligence. Microscopic models first developed in the form of acceleration-altering models, and later velocity-based models. Of particular note are the velocity obstacles family of models. Velocity obstacle models are velocity approaches which consider the region of available velocity space. The final microscopic approach is Cellular Automata (CA) which follows a discrete grid-like environment and are subject to a fixed set of rules, and consider only their nearest neighbours. There are also microscopic models which are unique or take a very different approach that they cannot be easily classified. These are presented at the end of this section.

The research of this thesis is on simulating pedestrian crowds with additional behaviours. Additional behaviours like groups and luggage are on a similar scale to the people themselves. As such macroscopic models, which struggle to record behaviours on the person-scale, are too coarse for use. The research is also concerned about how whole crowds interact with each other, which limits the effectiveness of data-driven approaches which tend to focus on single people. The microscopic approach, in contrast, is suitable at these tasks. Within microscopic models there can be acceleration (explored in section 2.2.1, and velocity (in section 2.2.2).

### 2.2.1 Acceleration-Based

Acceleration-based models are those which act with forces to navigate and move. Various forces can act on individual people, arising from a variety of sources, resulting in a combined force, which is integrated over time to move people around.

The most famous acceleration-based pedestrian model is the Helbing social forces model [88]. A very successful early model which demonstrated two phenomena, namely lane formation and oscillations at a bottleneck. Lane formations occur in corridors when pedestrians move in cross-flow (i.e. opposite directions), and extra steering rules when a colliding agent is within the line of sight. Lane formation is the phenomenon where people moving in the same direction follow roughly behind one another. This grouping increases the overall speed and reduces the amount of head-on potential collisions that people need to steer around. The advantages of this social force model is its light computation. Collision avoidance is proportional to the distance between agents, and the closer two people are, the more effect their interactions will have on each other.

The social forces model is possible to be improved upon and lends itself well to a GPU implementation [173, 112]. This is because, like with all microscopic models, there are numerous individual actors who all follow the same set of rules. The result is the ability to simulate many more people at a much faster rate, compared to using only the CPU. Applications to the GPU are not trivial and tend to require the whole simulation to be within the GPU. For the case of Karmakharm et al. [112] the long-term path planning was computed as a discritised grid with multiple layers for each possible exit point. This change was done to properly utilise the GPU hardware and allow the resulting model to handle many agents in real time.

The social forces model lacks accuracy compared to other models. Pedestrian motion is dependent on the distance to neighbours, and this can lead to abnormal

motion. Furthermore agents arrive at their destination in a slower time. Adjustments and extra forces can be included in social forces models to simulate extra properties. For example a body force is included for simulating evacuations where agents will be tightly packed to counter body compression [89], as well as a sliding friction force [127]. Such extensions can create more visually realistic behaviour compared to the standard social forces model in some scenarios.

Another force model, less suited for pedestrians is the Reynolds' Boids model [171]. The force rules are applied to flocking behaviours such as birds. For example, there is a force which all agents are attracted towards the perceived centre of the flock. Such forces do not create believable pedestrian motion, but was one of the first steps towards creating computational emergent macroscopic motion from rules applied to individuals.

An issue with acceleration-based approaches is that when a force suddenly applies or disappears, people can stop almost instantaneously. Force particles have inertia and take time to stop moving. This alternative approach can cause discrepancies in expected behaviour without additional factors. For further detail on this section, [40] provides a good explanation of the literature around social forces models.

### 2.2.2 Velocity-Based

Velocity-based models do not consider forces and acceleration, but rather the integral: velocity. These approaches tend to involve computing the relevant velocity for each person according to the various factors of the model.

A general framework for velocity based modelling of pedestrians is discussed in [139]. This framework makes only a few assumptions on the possible velocities, such as it being impossible to leapfrog over a person. [156] is an alternative velocity-based approach based on the bearing angle between agents. The bearing angle is between the vector of direction of motion and the vector connecting two agents. The angle varies greatly if the motion of the two agents is suitable, but remains close to constant if a collision is occurring. This approach to Velocity obstacles also uses only visual stimuli for intra-agent information exchange. There exists a set of models which search for the optimal velocity, which is dependent on the space in front of an agent. This approach is similar to vehicular dynamics and can produce various phenomena [200, 212].

A shortcoming of velocity-based approaches is the inherent inability to model stop-go processes seen in queues and similar. A similar phenomenon can be readily seen in cars at a traffic light when the light turns green. [201] adds noise into the velocity-based model to create stop-go motion and better match observations. To address these shortcomings, a velocity based approach [57] attempts to simulate these effects. It uses gradients to change the velocity vector. It is able to reproduce common phenomena like lane formation, and stop-and-go waves.

A popular velocity approach is velocity obstacles (VOs), which create limits in velocity-space to alter the possible velocities a person can travel at. A VO is the set of all velocities which will result in a collision between two actors. It is formed for the person of interest, and requires a target actor. It is calculated based on the current velocity of the other actor, so assumes the other actor will continue moving with the same velocity. If the person of interest moves with a velocity within the velocity obstacle it will eventually collide with the other target actor. Moving at a velocity outside of the velocity obstacle means the two actors will not collide.

Velocity obstacles function by examining the velocity and position of nearby moving objects to compute a collision-free trajectory. Velocity-space is analysed to determine what velocities can be taken which do not cause collisions. VO models lend themselves to parallelization since agents are updated simultaneously and navigate independently of one another with minimal explicit communication. It tends to be more computational and memory intensive than social forces models, but the large throughput capability of the GPU for such parallel tasks make it a very suitable technique for GPU implementation. Early models assumed that each person would take full responsibility for avoiding other people. Several variations include the reactive behaviour of other models [1, 117, 66]. One example is reciprocal velocity obstacles (RVO), where the assumption is that all other people will take half the responsibility for avoiding collisions [14, 81]. This model has been implemented on the GPU [26] and has shown credible speedup over the multi-core CPU implementation through use of hashing instead of naive nearest neighbour search. Group behaviour has also been included in VO models [86, 214] allowing people to be joined into groups. Such people attempt to remain close to other members of the group and aim for the same goal location. A further extension is Optimal Reciprocal Collision Avoidance (ORCA). ORCA provides sufficient conditions for collision-free motion by solving low-dimension linear programs. Freely available code libraries have been implemented for both single- and multi-core CPU [193]. VO techniques are suitable candidates for GPU implementation. The RVO model and implementation by Bleiweiss [26] shows notable performance improvements against multi-core CPU equivalent models. However, these methods must perform expensive calculations to find a suitable velocity. They tend to perform slower and are not guaranteed to find the best velocity.

Velocity object models were initially created for robotic navigation. Velocity object methods examine the available velocity space that will prevent collisions and select a velocity from the available set. Agents only need to know the position and current velocity of other nearby agents. No other communication is needed. This has the advantage that all information required is obtainable from sight, much like real people. Velocity obstacle methods have been applied to autonomous wheelchair driving through crowds [166] and warning drivers of impeding car collisions [185]. An advantage of these methods is predicting collisions by looking ahead in time and adjusting velocities sooner for smoother looking motion.

The original velocity object method by Fiorini and Shiller [64] suffered from oncoming agents entering a reciprocal dance. The reciprocal dance involves two agents arriving head on each other. To avoid collision they both select a velocity to the side. However, if both agents select the same side then they are going to collide again. They must select a new velocity to again avoid collision, which will be their initial velocity due to that being closest to the desired velocity of the agents. This is a repeat of the initial problem, so these agents are stuck in this situation of repeatedly selecting the same velocities and never moving.

Extensions to this model include accounting for car-like constraints [210], nonlinear motion [184], and group formation [170]. The latter attempts to simulate different group types such as tourist and friend groups. Various emergent behaviours were simulated in this model such as in low density, small groups form roughly a horizontal line so that they can talk easily to each other. In higher densities the group became a single-file line as this maximises the speed the group can move within the dense crowd.

To address the reciprocal dance issue, a new model, the Reciprocal Velocity Object (RVO) was created [14]. Both agents take half the responsibility to avoid collision. This prevents both agents from selecting their original velocity the second time round because that original velocity will be in an invalid region of velocity space. Such an invalid region will lead to a collision. This model requires both agents to use the same RVO model for steering due to the assumptions made in predicting the other agent's trajectory.

Other extensions to RVO add additional constraints, such as limiting acceleration (AVO [15]), rotating of non-radially-symmetric shapes (RRVO [70]). Most of these methods are made for linear equations of motion $\dot{p} = v$ for a change in position $p$ in time proportional to the velocity $v$. AVO has been further generalised to apply to any degree integrator [176] known as a continuous control obstacle. Other extensions by [9] allow for arbitrary homogeneous linear equations of motion. These have been gathered under a unification framework for general RVO where these different models are shown to be special cases of the general RVO [10]. This general RVO method allows for calculating a plethora of different equations of motion including linear and nonlinear, homogeneous and non-homogeneous types. It does not, however, extend to rotational shapes or account for a decoupling of the forwards vector with shape rotation.

### ORCA

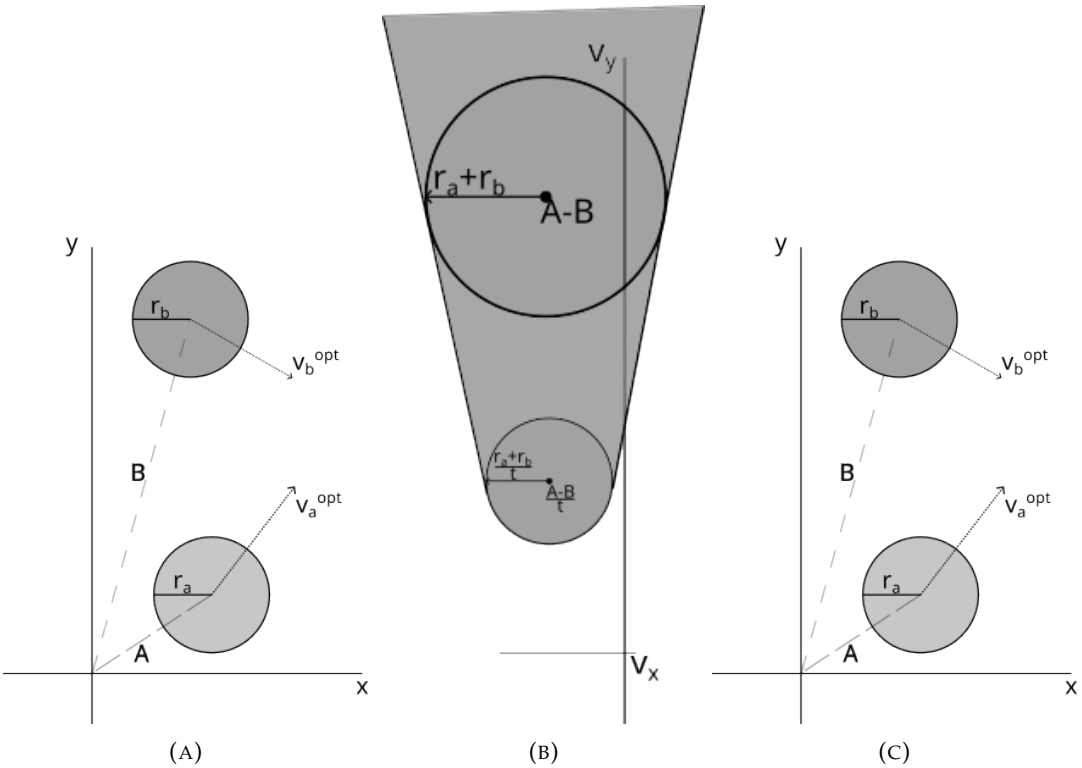
The Optimal Reciprocal Collision Avoidance (ORCA [16]) is a particular version of a VO. It allows multiple agents to navigate each other by creating a half-plane of allowed velocities for each neighbour. Once all half planes are calculated a velocity that most closely matches the desired velocity within the remaining free space is found using linear programming. These half-planes are conservative, so agents in tightly packed conditions are more limited than they are in a real representation.

The ORCA model functions when each person in the model has a start location and an end location they want to reach as quickly as possible, subject to an average speed and capped maximum speed. For each simulation iteration, each agent 'observes'properties of nearby people, namely radius, the current position and velocity. For each nearby agent a half-plane of restricted velocities is calculated (figure 2.1). By selecting a velocity not restricted by this half-plane, the two agents are guaranteed to not collide within time $\tau$, where $\tau$ is defined as the lookahead time, the amount of forward time planning people consider to avoid collisions. By considering all nearby agents, the set of half-planes creates a set of velocities that, if taken, do not collide with any nearby agents in time $\tau$. The agent then selects from the permissible velocities, the one closest to its desired velocity and goal.

It is possible that the generated set of half-planes does not contain any possible velocities. Such situations are caused by large densities of people. The solution is to select a velocity that least penetrates the set of half-planes induced by the other agents. In this case, there is no guarantee of collision-free motion.

The computation of velocity subject to the set of half-planes is done using linear programming. The problem for the linear program is defined with the constraints corresponding to the half-plane $\mathrm{ORCA}_{a|b}$ of velocities, attempting to minimise the difference of the suitable velocity from the desired velocity. Since each agent needs to find a new velocity, there is a linear problem corresponding to each agent, each iteration.

There are two main advantages of ORCA compared to other velocity models. The first is the ability for collision-free motion. This provides theoretically fewer




(A) (B) (C)

FIGURE 2.1: (A) A system of 2 people a and b with corresponding radius $r_a$ and $r_b$. (B) The associated velocity obstacle $VO_{a|b}$ in velocity space for a look-ahead period of time $\tau$ caused by the neighbour $b$ for $a$. (C) The vector of velocities $v_a^{opt} - v_b^{opt}$ lies within the velocity obstacle $VO_{a|b}$. The vector $u$ is the shortest vector to the edge of the obstacle from the vector of velocities. The corresponding half-plane $ORCA_{a|b}$ is in the direction of u, and intersects the point $v_a^{opt} + u$.

collisions, which is an important metric for crowd correctness. The second reason is the computational performance of ORCA implementations. ORCA achieves good performance for a velocity obstacle approach because it uses linear constraints for computing solutions. Velocity obstacles are decomposed to linear constraints, which can be computed relatively easily. This is in contrast to other VO methods which use complex shapes, which are more costly to compute. ORCA has been shown to display a variety of desirable properties. It exhibits self-organising behaviour [80], reproduces small-scale pedestrian interactions [78], and can match fundamental diagrams of experimental observations [50]. Performance-wise, it is computationally efficient [52] and numerically stable [51].

The ORCA model is advantageous compared to acceleration-based approaches because of the types of people it considers, and their importance in steering. An acceleration-based approach steers around other people proportionally to the distance between them. The closer two people are, the more important the forces affecting them are. The disadvantage of this is that fast, far away people are barely considered, even if they will be colliding shortly with the person. In contrast, ORCA steers around anyone who may collide with a person within a certain lookahead time. This approach leads to more believable emergent motion.

The popularity of the ORCA model has led to numerous extensions of it. Some of these adjust the shape of the agents to elliptical shapes [23]. Other extensions address behavioural aspects of ORCA. [216] addresses the least-effort principle applied

to ORCA. By having people navigate using least effort, they reduce their energy expenditure, and potentially more easily and swiftly arrive at their destination. [5] applies ORCA to agents with non-holonomic motion, such as cars, which are limited in how they move depending on the wheel position. [4] takes a similar idea and looks at ORCA for bicycles. [75] generalises the reciprocal requirement of ORCA into a variable reciprocity.

### 2.2.3 Complex Pedestrian Representations

Some pedestrian models focus on highly complex locomotion and as such typically focus on only one or a handful of people at any time, often performing computation offline. Some of these approaches draw on biomechanics to influence how a person can move. These models constrain the possible positions and velocities a person can take by considering factors like how much a leg can stretch and move. Due to the complex calculations involved these are usually done offline for interactions between small numbers of individuals.

Some groups have looked at modelling human physics and biomechanics to create motion and navigate objects [189, 20]. Agents are represented not as points or circles, as with other micro-simulation models, but as 5 circles, one for each shoulder, one for the body and 2 for the feet [110]. These shapes are compared for collision detection. Footstep locations based on the inverse pendulum (IP) model [126] are planned out for five or so steps. The motion of the five circles is predicted and compared with other agents to see if future collisions occur. Each agent's next few footsteps are calculated depending on the desired velocity and kinematic constraints. Any foreseeable collisions are re-calculated according to the agent's locomotion. This model is very computationally intensive and only runs tens of agents in real time. It creates a coupling of steering with locomotion, where locomotive constraints dictate the collision avoidance as well as vice versa. This is a useful feedback mechanism to account for locomotion constraints to affect steering. The resultant steering motion can look fake as not enough biological constraints are considered [189]. It is difficult to know how the model by Singh et al. computers steering compares to other models since the literature does not explain how the footsteps are recalculated when collisions are predicted to occur. It does allow for complex avoidance manoeuvres such as backstepping to allow an on-comer through the door. Potential future directions for this type of model include more biomechanical kinematics such as knee and ankle joints affecting energy cost, which affect the footstep output and produce more realistic looking motion. The advantage of this method is that the path taken looks natural and the timing of steps taken looks realistic. People can exist closer to each other because rather than one large circle to avoid it uses five smaller circles, which less conservatively estimates the shape of people. Agents in this model communicate between each other their planned motion, which can be more information than would be possible in a real life situation.

A different method uses whole body motion [160] and represents people in full 3D. The human can navigate complex environments and a few other agents. It has two parts: multi-agent planner for global and local motion, and high-degree-of-freedom (DOF) planner which minimises various energy constraints to create movement. A feedback loop returns whether the current path from the multi-agent planner is doable by the high-DOF and creates a new path if not. Obvious extensions are synthesising more natural looking motion, and heterogeneous agents. It is highly computational and works in real time for only a few agents, but they can navigate complex 3D environments as well as tricky 2D areas, by computing trajectories that

can satisfy kinematic, dynamic, and biomechanical constraints. This model lacks motion data for some movements, resulting in robotic motion. High-DoF models only look at the current time step, and could include further look-ahead for smoother motion.

These many degrees of freedom models are the start of a 2-way system between steering and locomotion. Biomechanics are used to influence the agent's steering ability. These are the most accurate models in this regard, though they remain computationally intensive and can commonly have visually incorrect looking movement when explicitly calculating body position as opposed to using motion capture animation on top of the agent location. These models were not created to simulate a whole crowd of interacting people, and hence are unsuitable for the proposed purpose of crowd-wide simulation.

### 2.2.4 Summary

The thesis is concerned about simulating a flexible modelling approach accounting for complex behaviours, complex motion, and real time simulation. For these reasons, microscopic approaches are the most suitable. They are able to easily describe differences at the individual level, and are suitable for high density crowds in small locations such as narrow corridors. Velocity based microscopic approaches are some of the more complex types of models. The complexity translates to useful model behaviour, such as accounting for people not according to their distance to them, but by how much of a potential collision they might have. In particular, ORCA addresses various concerns of previous velocity models, such as the reciprocal dance. ORCA uses linear constraints, which can be computationally lightweight compared to other RVO approaches, which is a desirable feature in the scope of this thesis.

## 2.3 Additional Behaviours

The previous sections have focused on the steering aspect of the pedestrian simulation. That is, how to avoid collisions between people and the environment. Up to now, there has been minimal assumption about who the people are, or their psychology or culture. People avoiding colliding with one another is a fundamental aspect of pedestrian motion, but does little when compared to real world pedestrians, who display a rich variety of differences between one another. One example of different behaviour is moving up or down stairs, walking at different speeds [182]. The aim of additional pedestrian behaviours within a model is to account for some of these real-world aspects within the simulation to better represent observable crowds.

Incorporating a behaviour into a model will affect the emergent behaviour. That's expected and desired, since the motion of a person in a group should be different when they are not in a group. However, joining multiple behaviours together can be difficult. Each behaviour changes some part of the simulation, and that can have unforeseen effects on other parts, such that the addition of extra behaviours makes none of the behaviours function as expected without special considerations. An example of this might be the social forces model [88], which has been applied to groups [134, 84] and evacuation [87], but needs a special, different consideration to combine the two together [217].

Approaches have been made to incorporate pedestrian behaviours together in a unified fashion. Hollmann [93] creates a framework which follows a holistic approach for modelling goal-driven, cognitive individuals. Behaviours for this model

are goal-driven: behaviours which alter the goals a person has. This is in contrast to the behaviours examined in this thesis where behaviours alter the short-term motion of a person. For a behaviour to be incorporated into the framework of Hollmann it requires the behaviour to incorporate the base functionalities of the framework. Hollmann's approach is based on well-established cognitive architecture to provide complex decision making intelligence to virtual people. The rules-based system it uses has many of the same strengths and weaknesses regarding collision avoidance and short-term motion as cellular automata. The performance of this work is targeted at offline functionality, and focuses on computation rather than speed of execution. The model uses a large amount of memory for each person in order to give each person a more complex psychological model. The model incorporates multiple behaviours together, but these behaviours are not able to overlap

### 2.3.1 Groups

People moving in groups is an important part of real crowds. People in groups can make up a large percentage of the total population [187]. Large group sizes tend to be less common. In the study of large group sizes it has been found that people will split off into smaller sized groups [46].

With respect to pedestrian modelling research, most models do not incorporate group behaviour with their models. This is probably because of separation of concerns. Such research is usually examining a specific aspect, so wants to not be affected by other behaviours, such as groups, even if they are an important consideration for real crowds.

Having groups of people in crowds is an important part of making believable simulations. However, the various approaches taken by the film and game industry do not usually disclose how they simulate their group crowds, and it is hard to figure out through visual inspection alone. For all the importance of having groups of people in crowds, there does not appear to be a standardised or cohesive approach taken by different models. For a more in-depth overview of pedestrian groups and the literature around it, [151] provides a comprehensive review.

Some research into groups can be categorised into a microscopic approach. This looks at the individuals with a group, and the effect it has. An interesting empirical finding of Moussaïd et al. [145] was that groups tend to form with certain shapes, as well as move slower than individual people. They find that group shapes deform when faced with increasing density. A social group will tend to create a "V" shape in order to maximise the ease of communication between the members. At high density, the group becomes a single-file line in order to maximise the ease of movement. They conclude with a mathematical equation to describe group shape and behaviour to reflect observed data.

He et al. [86] uses a least effort principle to define person-person avoidance as well as group-group steering. The method chooses one agent within each group as the leader. A downside of the work of He et al. is how conservative avoidance of other groups is. The convex hull created by an opposing group is determined by the most extreme clockwise and anticlockwise agents. For group shapes which are much more ovular than circular this can create unusual behaviour in avoidance, as the agent makes the simplification that the group size is circular.

The macroscopic properties of groups are factors that are present throughout the crowd. This includes values like the fundamental diagram of a group, which is the speed of the group, as a function of the density of groups around them. Within the research, the macroscopic effects of groups are less clear, or more controversial.

This can be because different factors in the experiments had large effects, such as the makeup of the people, or the culture of the people.

The fundamental diagram is altered when incorporating groups of people. The particular changes can be found from Mussaid et. al. [145, 146] examining the average speed as a function of group size. [97] found that in high density scenarios (with densities between 1.25 and 4 people/$m^2$) the person-level scale is affected by the existence of groups, but the variation of moving in small groups, compared to as individuals, is found to be less than the experimental noise. This comes at odds with [48] who predict a lower flow for small group sizes. [96] provides an interesting meta-analysis on egress time with groups to assess whether through the various experiments in literature, some overarching results can be drawn. They ultimately conclude that there is too much doubt between findings and confidence for the various experiments to say whether groups have any effect or perhaps no effect. Though the examination was for uniform motion through a corridor, Hu [97] found that the speed-density effects were smaller than experimental variations. Of interest were dynamic differences that people in groups move slower and perform more severe avoidance of others by maintaining more distance to other non-group people. Within the simulation these results are part of the input specification to the model, rather than a property that arises from the interacting model. As such, the results of the paper should be used when aiming to accurately reflect real pedestrian behaviour as input into simulation models used.

### 2.3.2 Luggage

Various experiments have been carried out in examining real pedestrian behaviour with luggage. Many have looked into the natural walking speed of people with luggage without obstacles or other people in the way, known as free motion [98]. This is for densities sufficiently low that people are hardly affected by other people, so can walk at their usual pace unimpeded. This generally occurs for densities below 0.5 people/$m^2$ [32]. Luggage can be categorised by size where “small" luggage is items such as briefcases and backpacks. Medium items are small trailing luggage or large carry bags, and large items are luggage of a similar size to people, or two pieces of medium luggage. Ye et al. [215] found the impact of different sized luggage on mean walking speed for free motion. They report that small luggage reduces mean walking speed by 2-3%, medium by 5-8% [32], large by 10-14%. Davis et al. [54] found that luggage size affects both walking speed and space requirements when examining airport terminals. Wu et al. [211] found that people with large luggage move on average of 1.25m/s and occupy space of 0.3 $m^2$/pedestrian. In contrast, individual people without luggage move around 1.55 ± 0.18 [218].

One way of understanding pedestrian models is by the fundamental diagram, a relationship between density and the speed of people [149, 104]. Each diagram is sensitive to the environment, such as corridor width and direction of movement. Zhang et al. [218] collate the fundamental diagram for various measurements for uni- and bi-directional flow through corridors to obtain aggregated averages for people without luggage, while Shah et al. [182] create a fundamental diagram for crowds in which a certain proportions of people have luggage. Shah et al. find that crowds containing people with luggage tend to move slower compared to those without at higher densities.

### 2.3.3 Autonomous Vehicles

The ethics and safety of autonomous vehicles is an important subject of current studies [71, 133, 140, 6, 74]. There is also examination into understanding how users respond to autonomous vehicles, and therefore how such vehicles should be programmed. A recent survey and overview of this is provided by Rasouli and Tsotsos [168]. Schwarting et al. use a variety of psychological metrics which model the altruism and selfishness of people to predict their interactions towards autonomous vehicles [179]. Bonnefon et al. examined the idea of pedestrian-first safely compared to passenger-first safety of autonomous vehicles. That is, given a scenario which will cause harm to either the pedestrians or passengers, whose safety will be prioritised. They found people would prefer others to buy pedestrian-first safe vehicles, but would prefer to ride in passenger-first safe vehicles, which provides a moral, ethical, and utilitarian dilemma as to the choice of algorithms used in autonomous vehicles [31]. Pettersson and Karlsson [162] find that different ways of presenting information on the same subject can yield different types of data. This means that the larger variety of methods of presenting and interacting with information about autonomous vehicles, the more understanding of the users can be found.

Work into simulating autonomous vehicles has been carried out with agent-based models. Boesch and Ciari [30] present an agent-based simulation model for autonomous vehicles, called MATsim, with the aim of it being simple enough for usage for those interested. This tool is used in a variety of applications, such as examining an autonomous taxi service [95], providing a theoretical examination in which to target future experimental research. Zhang et al. [219] try to predict the effect autonomous vehicles will have on city parking through use of an agent based model, and estimate that up to 90% of parking demand could be eliminated. [167] create a framework for pedestrians with autonomous vehicles. This recent work is based on literature to understand how people might behave around autonomous vehicles, and using the framework to predict new scenarios and explore a wider range of environments.

### 2.3.4 Summary

The behaviours explored in this section, groups, luggage, and autonomous vehicles, are behaviours that affect people at the individual scale. Pedestrian models most suitable for simulating individual scale behaviours are microscopic approaches. This is because microscopic approaches account for individuality and heterogeneity within the crowd. This means that behaviours can be described by how they affect a single person. This approach is one which makes logical sense, and also one that can be easily tested through experiments.

An issue with including pedestrian behaviour in a model is that it alters the behaviour of the people. If the model has been tuned to act a certain way, the inclusion of behaviour will result in the model behaving differently. Additionally, behaviours add computational performance, since more calculations must be performed. Ensuring the simulation runs in real time can be a challenge with the additional complexity of simulating behaviours. For this reason the software implementation is important.

## 2.4 Simulation of Pedestrian Models

The previous sections have explored pedestrian crowd models. This has involved understanding how to represent real crowds and behaviours through technical descriptions. To make use of pedestrian models, they must be simulated. This section discusses the important aspects of simulations. This includes how to implement the model, which concerns the process of getting something that can perform the computation and simulation of a crowd. Simulating a pedestrian model involves using a piece of software. There are various pieces of software available which are designed specifically for pedestrian simulations. Their purpose and benefit will be explored. An important part of model development is understanding how accurate and precise it is. Understanding the accuracy of a model can be found through validation. Potential processes of validating a model is also explored in this section.

### 2.4.1 Software/Frameworks

There are a wide range of open source and commercial software tools for pedestrian simulation, each has various pros and cons. It is useful to know about the tools which can aid in research and software development.

Menge [49] is a modular framework for pedestrian crowds. It has somewhat stringent installation requirements, but is useful for testing different models with the same design. Steerbench [190, 191] is a benchmark framework that aims to provide a numerical value to how valid a model is. It contains a suite of various scenarios that measures three main metrics: number of collisions, time efficiency and effort efficiency, as well numerous others. The benchmark results should only be used as comparisons between tested models due to results being presented as a value. Extra test cases can be constructed and incorporated into the framework to examine other cases, such as cluttered environments. The values the benchmark provides are a combination of the metrics, weighted by user choice. The values are not validated against real data, but do follow the accepted idea that efficient behaviours are natural.

A range of tools exist with a focus on steering behaviours. Steerbug [109] provides a set of time-varying-constraints in real time to aid debugging and understanding of steering models in real-time. It is aimed at specifying and detecting steering behaviours. This is suitable during development of models to avoid undesired behaviour such as circular motion, collisions, and wall-hugging. It is also suitable during model comparisons to see whether similar behaviours occur. A downside is that this method depends on user-driven evaluation. Users must specify behaviour to examine, which may not be related to the real-world accuracy of the model, only the apparent realism. Steerfit [22], another steering based software tool, uses a method of tuning parameters of steering algorithms to maximise desired properties, e.g. minimise turbulence at bottlenecks and reduce building evacuation time. It achieves this by employing different benchmarks to find optimal values across a range of scenarios. This also provides a method of examining the range of behaviour of models relative to its internal parameters. This allows understanding if models can be applicable being the scope it is introduced for. This method is not as easily extendable from a software point of view, but does theoretically allow for inclusions of other scenarios. Exodus [157] is targeted at evacuations. It is able to handle a variety of complex environments. It can simulate and analyse the movement of people during evacuations. It makes use of validated models, and is used

for predicting and understanding the effectiveness and safety of evacuation strategies. The model includes group dynamics as part of the complex behaviours that the virtual people demonstrate. It is a useful tool in analysing pedestrian evacuations, and can be extended to apply to novel pedestrian behaviours beyond evacuation, such as the work by Hollmann for modelling the pedestrian usage-cycle of a station [93]. This work provides continual perception input to simulated people, who adapt their itinerary as the environment changes.

There are data-driven approaches to quantify how accurately simulated individuals correspond to the real-world counterparts [129]. The goal is to evaluate how typical each individual motion and behaviour is, while previously mentioned methods identify if a set of behaviours appears in the simulation. A requirement for this method is real-life data with individual motion tracking. If this requirement is satisfied then it can numerically answer "how does the simulation compare to a real crowd". It measures density, proximity, and flock-measure to create numerical values.

A combination of Steerbug and the data-driven evaluation is provided in [36]. By examining real data, it can detect potentially erroneous behaviours. As with all data-driven approaches, a downside is that examined data must be similar to obtained real-world data. Provided data exists, this method can be more suitable and accurate to detect undesired behaviour than Steerbug, which does not use real-world evaluation. Guy et al. [79] provide a method of comparing real-world data with simulations. It measures how much deviation the simulation contains to the real data using an entropy metric. Given a state of a crowd $x_k$, how close does the simulator come to predicting the subsequent crowd state $x_{k+1}$? An alternative paper and metric is density [130], which works well for comparison between simulated and real large dense crowds. If real data is used, these two methods provide a statistical way of quantifying how accurately a model lines up with reality. A downside to this statistical analysis is that though it may result in crowds looking realistic, individual behaviours may not be.

SteerPlex [21] provides a method of estimating the complexity of a scenario. It is able to analyse the simulation environment to provide a metric of expected complexity. This can be an interesting framework to use, to see whether more complex scenarios require more complex steering methods to solve more realistically. Steersuite [188] is a CPU-based C++ framework that comes with various simple test cases and metric extraction tools that makes it useful to compare between pedestrian models. It also comes with a simple visualiser to display people as their computational 2D circles, but in a more immersive 3D space. It has a few pedestrian models built-in and the testing suite aims at providing a way of comparing and contrasting the behaviours of different models. By extension it can also provide the performance comparison of different models by timing model-specific sections of code.

### 2.4.2 Validation

Validation is an important step for understanding how accurately a model represents a physical system. Validation helps ascertain that a predictive model can match empirical observations. Validation can also be performed by contrasting against other similar models (i.e. cross-validation) to understand the differences between them. This section will provide an overview of some pedestrian simulation tools which aid in constructing, comparing different models, and understanding behaviour. It also briefly talks about the fundamental diagram, an important and often used picture for describing models and observations of pedestrian flow.

The use of collecting observational data is an important step in validating models. [63] collects studies of pedestrian behaviours and explores research gaps in the data. They conclude that there are a large number of issues with pedestrian behaviour data, including the difficulty in collecting data for a wide range of complex scenarios, of the challenges of collecting data of high risk setups, and the lack of comparison between different approaches studying the same behaviours.

#### Fundamental Diagram

One way of understanding pedestrian models is by the fundamental diagram, a relationship between density and the speed of people [149] [104]. Each diagram is sensitive to the environment, such as corridor width and direction of movement. [218] collate the fundamental diagram for various measurements for uni- and bidirectional flow through corridors to obtain aggregated averages for people. Due to its importance and capability to describe a pedestrian flow through unidirectional corridors, fundamental diagrams are part of the criteria proposed by [123] for creating validated pedestrian simulations. [149] created a model based on density-dependent filters to better match observed fundamental diagrams. The model is shown to generate smoother trajectories with fewer collisions, and can be applied to various popular steering models. [218] perform experimental experiments to find the fundamental diagrams for uni- and bidirectional flow in straight corridors. Their findings follow approximate agreement to other literature, but with fluctuations expected due to the complex interactions that happen when attempting to measure such values. [181] explores the quantitative validation of the flow through bottleneck environments and presents the findings and observations from literature, such as [207], as fundamental diagrams. Parisi [158] examines the running bulls event in Spain, which has large, dense crowds attempting to run as fast as they can. They find data on the very high density region of the fundamental diagram that is usually unobtainable due to the amount of people needed to be packed together. They find that as density increases, the average velocity of people also increases, at odds with the general literature on pedestrian systems. The findings suggest rather than a single speed–density relation, there exist many curves that correspond to a broad range of desired speeds for the running bulls event.

An example of a fundamental diagram is shown in figure 2.2. It shows a demonstration of what a fundamental diagram tends to look like as people move down a corridor in one direction. This diagram shows how the flow of people changes with density. The critical density is the density which permits the most amount of people to flow through. At the critical density people are moving slower than they would in free flow, with no one else around. This happens because the speed slowdown is counteracted by having more people. Above the critical density is considered congestion, where fewer people flow, and people are packed densely together. If there is no slowdown due to neighbours, the fundamental diagram would follow the dotted line entitled Free Speed.

## 2.5 Linear Programming

Section 2.1 and 2.2 examined the various approaches to pedestrian modelling and concluded that constraint based modelling using velocity obstacles, and in particular ORCA, represents a powerful approach to simulation which balances accuracy

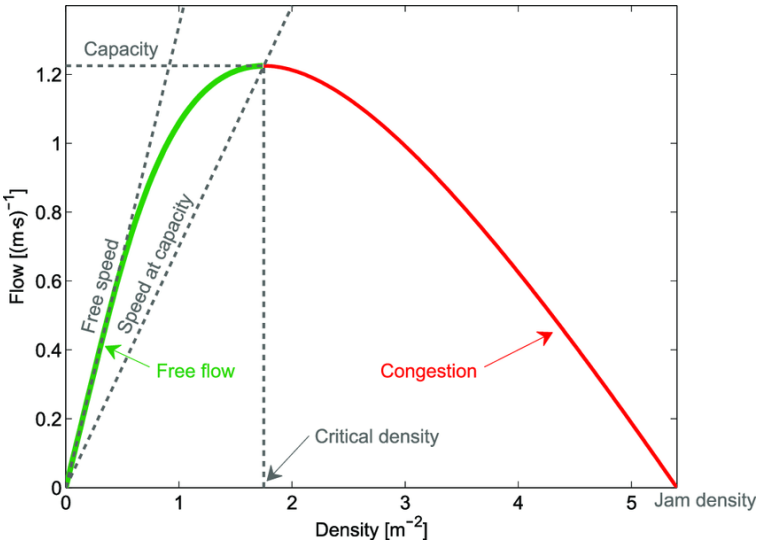



FIGURE 2.2: An example fundamental diagram for uniform flow in a corridor. Image from [61]

and performance. The fundamental approach to ORCA modelling is the use of linear programming and as such this section will explain the mathematics involved in linear programming.

### 2.5.1 Linear Programming Algorithm

The following explanation has appeared from part of the paper [37]. Linear programming is the problem of maximising an objective function subject to linear constraints. The objective function to maximise is represented as

$$\max \boldsymbol{c}^T \boldsymbol{x} \tag{2.1}$$

where $\boldsymbol{x} = (x_1, ..., x_n)$ is the set of values to be determined, and $\boldsymbol{c} = (c_1, ..., c_n)$ is the objective function coefficients. $n$ is the dimensionality of the problem. This is subject to the linear constraints

$$\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \tag{2.2}$$

where $\boldsymbol{b} = (b_1, ..., b_m)$ is a constant for each constraint, the maximum possible value of the equation, $m$ is the number of constraints of the problem, and $\boldsymbol{A}$ is a matrix of known constraints of size $n \times m$. This set of constraint vectors creates a convex polytope if there is a feasible solution. A linear programming problem is infeasible if there exists no solution for $\boldsymbol{x}$ that satisfies all of the constraints.

Various algorithms are suitable for large dimension problems, the most common of which is the simplex algorithm. In order to apply the simplex algorithm, the linear programming problem must be rewritten in its standard form. In standard form, comparisons are replaced by equalities with a slack variable.

### 2.5.2 Incremental Linear Programming

Incremental linear programming is a linear programming method that is conceptually simple and preferable with respect to performance for low dimensional problems [150]. Incremental linear programming works by considering each constraint

incrementally and calculating the intermediate objective function for each added constraint. It requires each step to have a unique and well-defined solution. To ensure this, up to two additional constraints per dimension are added , $\boldsymbol{x} \leq M$ and $\boldsymbol{x} \geq -M$. These ensure a finite solution and $M$ is taken as very large so as not to affect the optimal solution.

Two outcomes can occur to the intermediate optimal solution when incrementally considering a constraint: (1) if the optimal solution is already satisfied by the new constraint, no change occurs to the intermediate optimal solution; (2) if the intermediate optimal solution does not satisfy the new constraint, the optimal solution will exist on a point on the new constraint intersecting a previous constraint. In the case of (2), the algorithm to find the location of the new optimal solution is a set of $i-1$ $(n-1)$-dimensional linear programs, where $i$ is the current number of incrementally considered constraints, $0 \leq i \leq m$. When considering two-dimensional problems ($n = 2$), a set of one-dimensional LPs must be solved. It has been previously proven that one-dimensional LPs can be solved in linear time [17] — the time to find a new intermediate optimal solution is proportional to the total number of vertices/constraints currently considered, $O(i)$.

The set of $(i-1)$ 1D LPs can be parameterised by a variable $\hat{\boldsymbol{u}}$, where $\hat{\boldsymbol{u}}$ is parallel to the added constraint $A_i x = b_i$, labelled $l$. $\sigma(h, l)$ is the $\hat{\boldsymbol{u}}$-coordinate of the intersection point of line $\boldsymbol{l}$ and the considered constraint $\boldsymbol{A}_h \boldsymbol{x} = b_h$, where $0 \leq h \leq i$, i.e. $h$ indexes over the previously considered constraints. If there is no intersection then either the constraint at index $h$ can be ignored or the linear programming problem is infeasible. For each remaining constraint of the set, two values should be remembered, depending on whether $h$ is bounded to the left or right:

$$u_{\text{left}} = \max_{h=(1,\ldots,i-1)} \{\sigma(h, l) : l \cap h \text{ is bounded to the left}\} \tag{2.3}$$

$$u_{\text{right}} = \min_{h=(1,\ldots,i-1)} \{\sigma(h, l) : l \cap h \text{ is bounded to the right}\} \tag{2.4}$$

where $u_{\text{left}}$ represents the leftmost valid point on the line, if the line was horizontal, and, similarly, $u_{\text{right}}$ represents the rightmost valid point on the line. The program is infeasible if $u_{\text{left}} \geq u_{\text{right}}$, otherwise the solution is either $u_{\text{left}}$ or $u_{\text{right}}$ depending on the objective function.

Calculation of a new optimal solution is only required when the next constraint to consider renders the current optimal solution as infeasible. A worst case input set would require a re-computation of the solution for each constraint. In this set, each constraint renders the previous optimum solution invalid. If the order of consideration of the worst case input set was reversed, it would create a scenario where only the first constraint would require recomputing the solution. As such, the order of consideration of the input set is important. There is no simple strategy to optimally organise the order of constraint considerations. To achieve the best *expected* runtime, the order of consideration should be selected *randomly*. For a single LP calculated serially the expected run time is $O(m)$.

### 2.5.3 Solving Linear Programs Without a solution

There can be linear programs that do not have a valid solution. This occurs when the set of constraints result in no permissible value. For the case of the ORCA pedestrian steering model, when there is a program without a valid solution, there can be no guarantee of collision-free motion.

Models involving linear programs require a method of handling the case of no valid solution. Two trivial possibilities are to return either 0, or $P$, the point to minimise the distance towards. In the case of steering models like ORCA, $P$ is the desired velocity, the velocity a person wants to travel at, where there are no other obstacles or people.

The ORCA model provides an alternative possibility [16]. This is for the case of linear programs which attempt to minimise(/maximise) the distance to a point, $P$, in 2D. Each constraint is projected into 3D space, and it is tilted to move away from $P$ in the projected 3rd dimension. There is always a valid solution to this 3D program. This also means that at sufficiently large $Z >> 0$, there is always a valid solution to the 2D program at $z = Z$.

The advantage of this approach is that the 3D linear program can be decomposed into a set of 2D programs. Fast 2D linear program solvers can still be used.

Pseudocode for the ORCA approach to finding a value for a program without a solution is provided in 2.1

LISTING 2.1: Finding a velocity for a program without a solution

```
1  // [c] are the set of constraints which make up a program
2  // [c(z)] are the set of constraints shifted in z away from the desired
       point of solution. Each constraint is shifted according to its hardness
       .
3  // z-step is the amount in z to iterate each attempt
4  // P is the point to minimise the distance to
5
6  // Function to check if there is a solution to a 2D linear program.
7  function 2D_linear_program_has_solution()
8  // Function to find the optimal solution to a 2D linear program. Returns
       the value that is closest to p
9  solve_2D_linear_program( [c(z)], p )
10
11 z = 0
12 while (2D_linear_program_has_solution( [c(z)] ) == false)
13   z += z-step
14
15 solve_2D_linear_program( [c(z)], p )
```

## 2.6 Performance and GPU

The literature review has considered approaches for pedestrian simulation comparing the balance of accuracy and performance. Micro-simulation approaches are highly flexible and as such software tools tend to focus on steering based approaches. Linear programming has been highlighted as a crucial component in velocity based approaches to micro-simulation. A significant consideration of pedestrian modelling tools, of micro-simulation in general and of batch linear programming is the computational cost. As such it is necessary to consider approaches to accelerate the implementations using data parallelism such as that readily available in desktop GPUs. This section will help explain the implementation on GPU hardware, by explaining GPU architecture and its peculiarities.

### 2.6.1 GPU Parallelism

GPUs are built as high-throughput devices which are designed to maximise the throughput of the pipeline rather than minimise the latency of individual operations [68], as is the case with CPUs. GPUs achieve this through switching groups

of executable units on demand when the appropriate resources are available. For example, when a group of executable units is stalled due to a memory request, another group of threads are context-switched so that computation can be performed. Context-switching hides the memory latency and enables high memory bandwidth. To achieve good utilisation of the GPU device, two factors are required to hide the latency: compute utilisation (high arithmetic intensity) and memory access patterns (reduce the number of total memory movements through the pipeline).

GPUs programs require many threads which execute the same set of instructions (a kernel) on different regions of data (data parallelism). At the execution level, threads are grouped into batches of 32 threads, known as a warp. If some threads within the warp do not require the computation, they are masked out. The worst case scenario for GPU performance is when only one thread requires a certain computation, leaving the remaining threads masked and effectively idle. Any branching paths of execution within a warp causes all threads to execute all paths, with the appropriate threads stalled (or masked out from performing instructions). In general, any aspect of divergent computation should be minimised to improve compute utilisation and maximise performance.

Optimising memory access can be understood through the use of memory fetches. Memory is fetched through 32 byte level two (L2) cache lines. If a byte of data is required, the corresponding 32 byte cache line where the data resides is transferred from memory. It is therefore important to utilise as much data from as few cache lines as possible. In the worst case scenario, each thread within a warp issues a transfer request to move a unique cache line (i.e. a scattered read). The transfer of many cache lines requires greater bandwidth and increases the latency of total memory movement, reducing the performance of the code.

### 2.6.2 GPU Acceleration of Pedestrian Simulations

GPUs have been used to accelerate pedestrian simulations through a variety of different implementations. [115] developed a general agent-based framework on the GPU, FLAME, which is aimed at simulating large population sizes of agents. This framework has been applied to pedestrian simulations in [112]. It simulated acceleration-based pedestrians and handled global navigation through a group map of force fields. The force fields are calculated for the whole environment, and there is a different field for each possible exit. This approach is computationally lightweight and permits hundreds of thousands of agents in real time. [135] focus on maximising performance, being one of the first implementations to achieve millions of people simulated in real time. They also use a force-based approach, which is computationally light weight on the GPU. [147] compare forces-based and cellular automata approaches and obtain performance improvements of approximately 4 times by exploiting GPGPU.

Cellular Automata models are an attractive choice for high performance because the discretized space is computationally efficient even without GPU parallelism [177]. [120] explore the use of multiple GPU devices for a pedestrian simulation. This approach uses cellular automata and calculates the forces applied to people to calculate how they navigate through the grid. The computation of the navigation grid is complex, and this is performed on other GPUs while one GPU calculates the updates for each agent.

[222] uses an agent-based approach that combines cellular automata and acceleration-based. They design routines that are parallelised and performant for the GPU and achieve speed over 18 times against the single-core CPU equivalent.

### 2.6.3 Linear Programming on the GPU

The following explanation has appeared from part of the paper [37].

Literature on the use of GPU LP solvers can be separated into three main topics: early solvers before GPU computational APIs, LP solvers aimed at solving single LPs efficiently, and LP solvers specialised in solving multiple problems simultaneously.

Research into the use of GPUs for improving linear programming performance began with the start of GPUs as programming units. Early examples (such as [73, 107, 106]) showed limited performance improvements over serial algorithms for problems larger than 800 dimensions by 800 constraints. Such early models struggled with limited device memory. With hardware advances larger problems can now be tackled and larger speed-ups can be obtained. The advent of dedicated GPU computation programming languages, such as NVIDIA CUDA [153], has made it easier to develop efficient LP models on the GPU.

A large proportion of relevant papers examine high-dimensional problems using simplex algorithms, undoubtedly due to the popularity and efficiency of the model. Hall [83] provides an overview of early multicore simplex algorithms, concluding that speed-ups can be obtained by parallelising for various problem types, excluding large sparse LPs. The trend of applying the simplex algorithms to GPUs has continued, with speed-ups occurring for many different simplex algorithms including the regular simplex algorithm [128, 114] and the revised simplex [24, 194]. Ploskas and Samaras [163] showed that the Primal-Dual exterior point simplex algorithm is more efficient than the more standard Revised Simplex algorithm on GPU hardware. They achieved this through minimising CPU-GPU memory transfer, since memory transfer dominates the runtime of such large problems. They achieved speed-up for all problems tested compared to the simplex-GPU algorithm.

The choice of pivot rule plays an important part in simplex algorithms. Choosing a poor rule can slow down performance and lead to no optimal solution being found. Ploskas and Samaras tested the effects of pivot rules on GPU hardware [164] and found that GPU versions perform better than the CPU equivalents for problems larger than 500 dimensions and constraints, for dense constraints. Lalami et al. [128] deals with efficient memory transfer through page-locked host memory which provides higher memory bandwidth. In this case, the overhead of data transfer to the device is hidden through asynchronous transfer and computation by staging the problem into smaller units of work. They demonstrate a speed-up of around 12 times for problems larger than 2000 dimensions by 2000 constraints for the regular simplex algorithm, and a speed-up of 2.6 times for problems of 500 dimensions by 500 constraints. With respect to GPU implementations of non-simplex algorithms, an early implementation by Smith et al. [192] demonstrates that the matrix-free interior point algorithm shows some performance speed-up for large sparse matrices. For problems larger than around 16,000 dimensions by 16,000 constraints, the GPU implementation outperformed the CPU multicore equivalent. This was due to the efficiency of computational operations required.

An observation of the previous literature suggests there exists a small size limit at which LPs should only be computed on the CPU. This is due to the limited parallelism available for smaller problem sizes. Below this amount CPU implementations are equivalent or better performing than GPU equivalents. CPLEX is a CPU optimisation software package containing the functionality to solve LPs [47]. It is able to solve problems using different algorithms including dual simplex, primal simplex and barrier method. It uses multithreading to solve models, but this efficiency decreases up to 4-8 threads after which increasing thread count will not significantly

change execution time [100]. GLPK [138] and CLP [44] are open-source serial simplex solver methods. The performance of different CPU LP solvers is extensively tested [143, 69, 101] but can still remain challenging to find the most efficient method for a given problem.

In order to expose greater parallelism for small LPs, many small LPs should be computed simultaneously. Gurung and Ray [77] examines an algorithm for solving numerous dense LPs simultaneously using the simplex algorithm on the GPU. By considering many LPs at once, asynchronous memory transfer can occur simultaneously with computation of results to increase performance. Also the use of many LPs ensures the computational cores on the device are all being utilised. They show a speed-up over equivalent CPU algorithms for square LPs as small as 5 dimensions by 5 constraints for 100 batches. Multiple concurrent streams are active in the algorithm. Such streams allow overlapping memory transfer and kernel computation. Thus, rather than copying all the data across from CPU to GPU, then solving, then copying data back to CPU, the batch LPs are split into smaller groupings. Once data has been transferred to the GPU for one of these smaller groups, the kernel can perform the required computation. Simultaneously, the information for another group can be copied across to the GPU. By splitting tasks into these smaller groups, total device utilisation is increased, increasing performance. The current implementation limits the size of feasible LPs to 511 dimensions by 511 constraints [76].

## 2.7 Summary and Discussion

This literature review has explored pedestrian simulations and steering models. Data driven approaches are powerful but are unable to ensure constraints of physical behaviour are enforced. Another downside within this thesis is because data-driven approaches struggle to handle crowd-sized numbers of people, and the scope of the research is concerned with large numbers of people in crowds. When considering the applicability of macroscopic models towards the thesis research, there are various issues that arise in macroscopic models. One issue is that they lack detail at the person scale. It is difficult to account for differences between people without first understanding the impact they have on the macroscopic crowd as a whole. Macroscopic crowds can simulate huge crowds in real time, but this comes at the cost of lacking detail. A hybrid approach to exploring the thesis research appears to be a potential route forwards. It can handle numerous people, and can handle people displaying complex behaviours. The downside of this approach is the computational power required, which makes it difficult to run in real time. Since the research in the thesis is interested in heterogeneous agents displaying a wide variety of behaviours, microscopic approaches are the most suitable. In particular, the velocity obstacle approaches provide a good balance between performance and fidelity.

The review also explored pedestrian behaviours. It has explored the science regarding three particular behaviours: groups, luggage, and autonomous vehicles. These behaviours influence and alter the motion of individual people, and as such can be considered person-scale behaviours. Microscopic models that can account for changes to individuals are a good choice for implementing these behaviours as they naturally allow behaviour to be described at the individual level. Other approaches to describing pedestrian behaviour with micro-simulation are possible. Behaviours can be accounted for by macroscopic crowd-wide scale. A common difference for this approach is that the crowd behaviour is a controlled variable within the simulation. In other words, the approach will dictate how the crowd should function

as the effect of the behaviour takes place. An example is that a crowd behaviour can take the effect of altering the speed-density relation as the proportion of people in crowds have luggage. This is in contrast to the microscopic versions, where the speed-density relation is not directly altered, but rather by changing the behaviour of individuals, the interactions between people alter the emergent speed-density relation.

The implementation of multiple behaviours in a model can result in incorrect motion for the model, as explained in Duives et al. [59]. This occurs when the behaviour makes changes to the calculations within the simulation in ways that were not initially considered. Cellular automata are noted by Duives et al. as being one of the few types of models which are able to handle additional behaviours easily. This is because cellular automata are rule-based, which lends itself well to handling additional behaviours in the form of rules. Cellular automata have the major drawback of using discrete space. This limits the accuracy of high density simulations, as well as limiting the level of details of results which can be obtained. ORCA is a powerful modelling approach because it combines complexity with computational performance. The use of linear constraints to find velocities makes it a performance VO approach, and it shows accurate motion through predicting collisions occurring in the future. Combining behaviours with steering approaches other than ORCA causes issues that affect the behaviour of the steering, which can result in more collisions. ORCA can avoid this by combining behaviours with the computation of collision avoidance together in one calculation, the details of which are explored in the following chapter.

The review has explored key aspects in the speed and performance of the constraint-based microscopic-scale model that will be used throughout the thesis. Linear programming is a fundamental field of mathematics that is used for ensuring simulated people can steer around each other. GPU architecture is important for implementing GPU accelerated algorithms, and plays a part in creating the high-performance simulation later in the thesis.

The literature review can express the following main findings from literature.

- Performance vs accuracy must find a compromise when considering crowd scale simulations in real time.
- The addition of behaviours to a steering model can impact the accuracy of collision response.
- ORCA presents an accurate collision response model with the potential for extension points if behaviour can be combined with the collision avoidance calculation, through expressing behaviours as constraints.
- Linear programming is an essential computation within the ORCA model. There have been performance improvements to linear program solvers, but these have not focused on solving large numbers of linear programs simultaneously.
- The GPU is a feasible candidate for accelerating simulations

These findings will be used within the thesis to direct, and answer the research questions. The thesis will address some of the limitations in the state-of-the-art models, and also address these main findings.

# Chapter 3

# Constraint-Based Behaviours Modelling

## 3.1 Introduction

There is a commonly occurring problem in pedestrian simulations. This problem is the difficulty in incorporating pedestrian behaviours while maintaining accurate functionality of the rest of the model, such as collision avoidance. Chapter 2 reviewed the state of the art approaches to pedestrian simulations. It was concluded that microscopic simulation approaches are the most appropriate for representing a wide range of person-scale behaviours, including collision avoidance and more general interactions. ORCA is a promising candidate for collision avoidance due to its performance in using constraints, its believability, and complexity in looking ahead to potential future collisions.

Microscopic collision Avoidance methods and behaviour methods both change the motion of people. Collision avoidance finds a motion that takes a person towards their goal. Behavioural computations find motion that exhibit analogues to real behaviour, such as moving together in a social group, or stopping to watch something interesting happen in the nearby environment. A commonly used method [49] is to calculate the steering to produce a resulting motion that avoids local collisions, then further alter this motion to account for behaviours. The issue with this resulting motion is that while it complies with behaviours, it deviates from the collision-avoiding motion and possibly results in collision [59]. This can be particularly problematic in high densities or large crowds.

This chapter aims to address the flexibility and accuracy trade off of previous state of the art approaches by extending the flexibility of previous constraint based approaches by incorporating behavioural aspects as constraints themselves. This proposed solution aims to tightly couple the collision response with behavioural modelling to provide accurate simulation with the ability to add a complex and diverse set of behavioural characteristics. To balance the significance of collision and behaviour on simulated behaviour the proposed modelling approach provides a simple way of controlling the impact of behaviours and functionality using a novel property of constraints explained within the chapter, known as constraint hardness. The hardness of a behaviour determines how strictly it is adhered to, allowing rigid

behaviour without impacting other facets of the model functions. The formulation proposed is referred to as the Optimal Constraint Behavioural Model (or OCBM).

The thesis contribution of this chapter is part of contribution C 1 outlined in the introduction. This chapter is concerned with the description and performance evaluation of OCBM. The specific contributions of this chapter, which aim to tackle this contribution, will be:

1. Implementing a constraint-based simulation framework. This framework describes pedestrian motion and action as mathematical constraints.

2. Describing and implementing hard and soft constraints. The hardness of a constraint controls how strictly the associated behaviour is adhered to. This permits the addition of new behaviours and functions without impacting already implemented behaviour.

3. Demonstrating the flexibility of constraint-based framework through models of new behaviour. Three particular behaviours will be added to the collision avoidance model and the behaviour of the model will be explored.

4. Evaluating the performance of constraint based modelling. Experiments will be performed which examine how different simulation parameters alters the performance of the implemented framework.

A key component of the OCBM is the concept of "hardness" of constraints. Functionality of the model, both behaviour and steering, is described mathematically using constraints on the pedestrian motion. The "hardness" is a mathematical means of describing how much a behaviour/steering constraint should be adhered to. An example is using hard steering combined with soft behaviours. This will result in motion that prioritises avoiding collisions while also demonstrating additional behavioural motion when it does not impact potential collisions.

The OCBM approach can be used for any number of general pedestrian behaviours. The requirement for using a behaviour within this novel model is that the behaviour has a representation that constrains the motion of people. Within this chapter three particular behaviours are examined: groups, luggage, and autonomous vehicles. These three behaviours are either commonly observable in many types of pedestrian crowds, or are of recent interest due to technological advances. The choice of three behaviours is to limit the scope and also numerous enough to explore various facets of pedestrian behaviours. The key requirement is having a method of converting behaviours into desired velocity or mathematical constraints. Though three particular behaviours were selected within this chapter, the framework provided is extensible and applicable to other potential pedestrian behaviours.

Section 3.2 explains the implementation of the novel modelling approach, OCBM, and how constraint hardness is used. Section 3.3 show how OCBM can be applied to steering as well as pedestrian behaviours, notably: people in groups, people with luggage, and interacting with autonomous vehicles. Section 3.4 explains the implementation of the model into code using the SteerSuite library [188], and assesses the behaviours chosen through experiments. Section 3.5 examines how the implementation performs with regards to timing and speed of computation through experiments. Finally section 3.6 discusses the implications of the results and assesses how the chapter has managed to answer the contributions of the thesis.

## 3.2 A Framework for Hard and Soft Constraints

This section explains the proposed OCBM constraint-based modelling approach and its implementation detail. It explains the key aspect, which is the softness/hardness of constraints to allow combining numerous behaviours. The modelling approach will be tested in the subsequent sections which will apply the hard and soft constraints approach to a range of pedestrian crowd behaviours.

The idea of using constraints on the velocity space has been previously used in the ORCA family of steering models [56]. OCBM increases the number of constraints considered by also creating constraints due to behaviours. The details of how to construct constraints for behaviours is addressed in the following subsections.

### 3.2.1 Linear Programming and Constraints

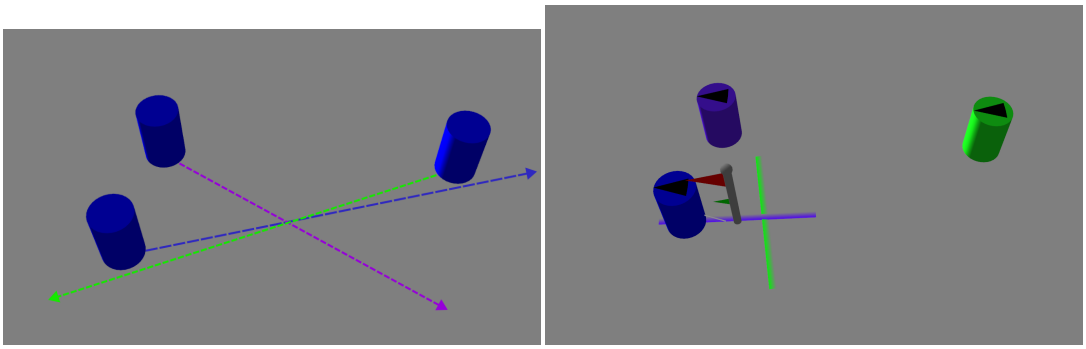
A linear program is a problem which aims to minimise a function given a limited subset of space. Within the context of pedestrian simulation each person may be described as having a subset of velocity space. This space is the velocity the person can travel at. It is bound by a circle whose radius is the maximum speed. More constraints are applied to the space depending on behaviours and neighbouring people. The function to minimise is the difference to the person's desired velocity. If the person's desired velocity is within an invalid region of velocity space, the linear program finds the closest valid velocity to it.

Section 2.5 in the previous chapter explains linear programming in greater detail. Linear programming is relevant to state-of-the-art because various RVO models, including ORCA, make use of the method for obstacle avoidance. The OCBM approach uses the ORCA steering model as the fundamental steering choice and expands on it by extending the flexibility beyond simple collision response. The details of the original ORCA approach are described in the literature review section 2.2.2. Any steering model that functions by limiting velocity constraints could be a suitable replacement within OCBM for handling steering. These include the extensions to ORCA like elliptical agents [23].

Figure 3.1 shows a simple setup of three people approaching each other. Figure 3.1b shows the ORCA half-plane constraints formed for the leftmost blue agent. Also visualised is the left blue agent's desired velocity (small flag) and actual velocity (large flag), which in this situation are the same. In this situation there is a large valid solution to the linear problem formed, and the chosen velocity is the one that is closest to the desired velocity, which here is the same value.

The resulting velocity space formed by the half-plane of steering and behaviours is the set of velocities a person can choose to avoid collision. It is possible that there is no valid solution. In this case a method should be chosen to attempt to select the next-best option. If there is no valid solution, then various methods can be chosen to compute a possible velocity. The method chosen by the ORCA paper [16, 17] recreates the linear problem by gradually extending the constraints away, until a velocity or region becomes valid. Such an alteration no longer provides guarantee of collision avoidance, but due to the look-ahead aspect of the model it may still be possible. This solution should be the choice which least penetrates the velocity space of other agents in a collision.

Linear programming problems having no solutions in the standard ORCA model are found most commonly in very dense scenarios. Within a physical environment this would typically represent a case where people are closely surrounded by others, and have limited choices in how they can move. In the OCBM approach careful



(A) (B)

FIGURE 3.1: Graphics of 3 people approaching each other using the ORCA model of collision response. The graphical representation is from a computational experiment implemented and rendered in SteerSuite. (A), three people heading towards each other. Their target goals are shown by dotted lines. If there are no other obstacles, they would follow this line exactly. (B) the same simulation frame, superimposed with velocity computation on the leftmost agent. It includes a coloured half-plane for each other agent (purple and green), and two flags, one large for the current velocity and one small for the desired velocity. These velocities are the same value in this case.

consideration has been given to the impact of adding additional behaviours which result in numerous additional constraints. The addition of constraints has the potential to create a scenario where it is not possible to solve the linear program for what may visually appear to be quite a simple setup. The OCBM uses the same approach of gradually moving constraints until a valid solution is found. An important inclusion is a "hardness" parameter in order to account for psychological effects. A harder constraint is one which is adhered to more strongly. This can be justified by considering a social group of people approaching an obstacle. A person, depending on their psychology, may choose to navigate around the obstacle by separating from the social group (a weak constraint), or stay close to their group at the detriment to their movement speed (a hard constraint).

### 3.2.2 A Combined Model of Soft and Hard Constraints

People are noted to adhere to behaviours differently. This can be from person to person, or between cultures. In order to include this concept within the OCBM approach, this chapter proposes to introduce the novel idea of giving linear constraints a 'hardness'. Adding a "hardness" to constraints is proposed to address problems that arise in linear programs with numerous constraints. This parameter affects scenarios where motion for a person cannot be found that permits a collision-free trajectory and adheres to behaviours.

Hard and soft constraints address these issues by weighing the constraints caused by different behaviours differently. So constraints caused by less important behaviours will be "softened" which in effect creates a prioritisation of behavioural components. Each type of behaviour is provided a weight (equivalent to the "hardness") which determines how much alteration should be made for that type of behaviour. Hardness must be greater than 0 with no maximum value. Softer constraints are closer to 0. This hardness value only comes into effect when an agent

has no permissible velocities when accounting for all constraints and as such constraints are considered equal under normal conditions.

Provided there is an available, non-zero region within the linear problem created by the numerous neighbouring constraints, there is a guarantee of collision-free motion. There are however situations where this is not possible. Such situations tend to occur when there are lots of constraints, caused by numerous neighbouring agents, such as in dense scenarios, or through lots of additional behaviour constraints.

Behaviour constraints typically have a different priority for adherence compared to collision avoidance. If considering group behaviours, one group may be willing to split apart to navigate a difficult section (e.g. friends within a crowd), whereas another group may be more willing to stand still and not advance towards the goal in order to best stick together (e.g a family group). These scenarios arise when the opportune scenario is not available, i.e. when there is no valid solution to the linear problem. These tend to more often occur in high density or large crowds. In this case, the constraints created by different behaviours are adjusted in different amounts. This phenomenon is captured by the mathematical description of the hardness of the constraint. A soft constraint is one which adjusts more easily and can potentially be discarded entirely. A hard constraint, in contrast, will adjust less or even not at all.

The hardness of constraints can be provided according to the behaviour that generates it, or at an individual level driven perhaps by psychology of the individual. One example is that people will avoid colliding or getting too close to autonomous vehicles, which can be modelled as a very hard constraint. At the individual level some people may be more prone to dropping their luggage due to psychological reasons or physical ones. It is proposed that these specific examples and a wider range of phenomena can be modelled by the hardness of the constraints.

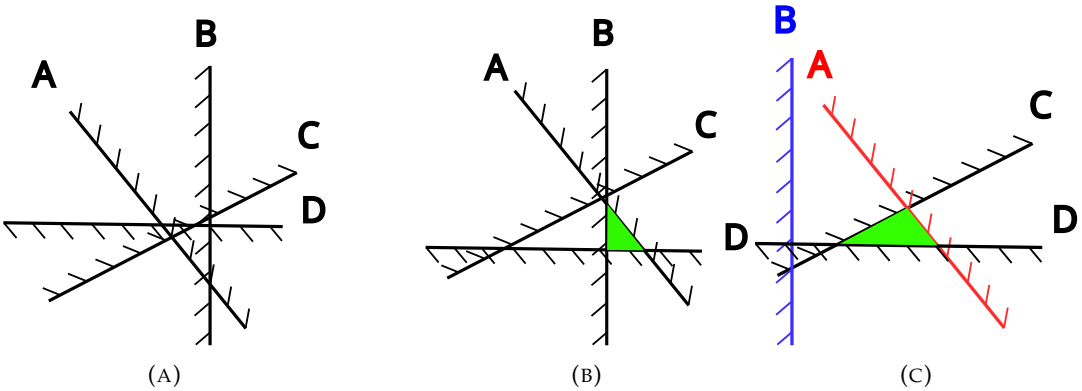
#### Implementation

It has been established that each person has psychological differences which may result in them responding differently for any given behaviour [142, 204]. This is computationally represented as each person containing a set of parameters specifying how soft or hard various behaviour constraints are for them. This allows heterogeneous alterations for behaviour models. Other agents are not able to usually see and observe what their neighbours' soft constraints are. They make no assumptions nor observations about this (similarly to how agents do not know the goal locations of observable neighbours).

In a situation where a valid solution exists to the constructed linear problem, the agent can move collision-free and adhere to all behaviours. In the case of no valid solution then the existence of these different hardness constraints allows a priority of what should be most strictly adhered to. The "harder" a constraint the more rigidly it will be adhered to.

To solve the scenario for no valid solution, the approach taken by ORCA can be used as a basis (see Section 2.5.3 which provides more detail on this approach). The ORCA approach is useful because it makes an attempt to find a suitable velocity by computing the velocity which least penetrates the velocity space of the constraints. OCBM extends this by changing how important it is for a constraint to be adjusted, e.g. a hard constraint is one which should not be adjusted much.

There are alternative choices for the case of no valid solution. Alternative examples include softening constraints which are caused by further away neighbours, which can ensure collision free motion between slow, near neighbours but increase




(A) (B) (C)

FIGURE 3.2: A linear problem with no valid solution. (A) shows an example problem with no solution. The invalid side of a half-plane has a chequered pattern. To find a feasible region the constraints are shifted until some region is valid (green) (B). (C) Show the same scenario but with one soft (blue) and one hard (red) constraint. The soft constraint has moved a lot and the hard constraint has almost stayed in the same location. It produces a different valid region of solution (green).

the severity against high moving further away neighbours; or simply stopping motion if no valid solution can be found. The least-penetrating approach is chosen because it avoids the chance of seriously colliding with another object. Future work remains to explore how alternative approaches may be suitable for specific scenarios.

A modeller must be careful when describing behaviour as the addition of hard behavioural constraints have the potential to impact on steering, and will usually result in "worse" statistics of the steering behaviour. An example is measuring collision-penetration depth. This will usually be worse in the presence of harder constraints because other behaviours may need to be balanced or even take precedence over steering. Mathematically, an unweighted constraint has a weight value ($w$) of 1. A soft constraint is one with a weighting of $0 < w < 1$, and will move more than other constraints. A hard constraint is $w > 1$ and moves proportionally less than an unweighted constraint.

### Example

Figure 3.2 shows the scenario of a linear problem without a solution (figure 3.2a). It shows 4 half-plane constraints. The chequered side of each line is the unfeasible region. There is no area that is feasible for all 4 constraints. Figure 3.2b shows the case if all constraints have an equal weighting ($w = 1$). The green triangular region is the resulting valid velocity. All four constraints have shifted an equal amount of 1 unit. In the case of soft and hard constraints. Figure 3.2c shows the different resulting velocity region. Constraint A is hard ($w = 5$) and B is soft ($w = 0.2$). The hard (A, red) constraint has barely moved, only 0.2 units. In contrast, the soft (B, blue) constraint has adjusted more than any other constraint, at 5 units. The remaining constraints (C, D) shift the same amount as before.

To describe the OCBM more formally, let $d_{A|B}(h_B\mathbf{v})$ be the distance of velocity $\mathbf{v}$ to the edge of the half-plane formed by $B$, either from ORCA or behaviour, scaled by the hardness. $h_B$ is the hardness of the constraint $B$. The new resulting velocity is

| Behaviour | Weight $w$ |
|---|---|
| Steering | 1 |
| Groups | 0.7 |
| Luggage | 2 |
| Avoiding Autonomous Vehicles | 10 |

TABLE 3.1: Suggested weights for different behaviours in the OCBM framework.

one that minimises the maximum distance to any of the scaled half-planes induced by the behaviours:

$$\mathbf{v}_A^{new} = \text{argmin}_{\mathbf{v} \epsilon D(\mathbf{0}, {v_A}^{max})} \max_{B \neq A} d_{A|B}(h_B \mathbf{v}) \tag{3.1}$$

A low hardness $h_B < 1$ means that $d_{A|B}(h_B \mathbf{v})$ will require a large $\mathbf{v}$ to equal the same value. This results in the constraint being moved a larger distance.

Table 3.1 provides example reference weights for the hardness of the different behaviours. These are example values to provide guidance on potential values. These values are tested visually, and have not been verified against real data, and should be adjusted acording to the specific scenario of interest.

## 3.3 Modelling Behaviours with Hard and Soft Constraints

One reason to have different hardness between constraints is because the constraints are generated by different processes. This section will explain the implementation of behaviours as mathematical constraints. It will also apply the softness of constraints to these behaviours to create a more complex pedestrian simulation model.

Three behaviours in particular are considered to demonstrate the flexibility of OCBM. These are groups, luggage, and autonomous vehicles. These three are chosen because they can play an important role in crowd dynamics. Groups are ubiquitous in crowds. Luggage and autonomous vehicles are less common, but people notably change their dynamics when carrying luggage or moving in the same space as autonomous vehicles. Although these three behaviours are chosen the OCBM approach is not limited to these and is generally applicable to a wide range of behaviours.

This section attempts to describe these behaviours as mathematical constraints which can then be used as part of the OCBM model, similar to how ORCA constraints describe steering. Converting these behaviours to constraints is something novel proposed in this chapter. By creating a set of constraints for a behaviour, these can be combined with other behaviours, and ORCA steering, to create a more complex linear program, the solution to which provides the velocity the person takes. This resulting velocity takes into account all the behaviours and steering motion together. This tackles the problem outlined at the start of this chapter, which is the difficulty in finding pedestrian motion which accounts for both steering and behaviours.

The following subsections will tackle each of the three behaviours in turn, following the order of Groups (subsection 3.3.1), Luggage (subsection 3.3.2), and Autonomous vehicles (subsection 3.3.3).

### 3.3.1 Group Behaviour

Groups are social groupings of people who walk together and have the same end location as other members of the group. These groups are between 2 and 6 people in size [2]. Friends or family walking together in groups is the behaviour explored here. Group behaviour is an important behaviour in crowds due to the prevalent proportion of people that move about in groups [187]. These can vary from couples and small families, to large tour groups. The psychological behaviour can vary between different sized groups. Despite this group behaviour is infrequently considered within pedestrian modelling and simulation, and pedestrian models with group behaviours are designed specifically for the task [170].

The groups of interest within the context of this chapter are those behaving similarly to small groups of friends [46]. Such groups have a desire to stay together, but are flexible in the group shape (i.e. relative placement of people within the group), and are willing to adjust their relative positions as they navigate the environment. In previous literature it has been suggested that large sized groups can be split into smaller groups [103]. This occurs due to the limited ability of communicating with many people while walking. Because of this, large sized groups can be decomposed into multiple smaller groups consisting of between 2-7 people.

The work in this section was inspired in part by the paper of [214] and [86] (discussed in detail in literature review, section 2.3.1). In their work they compute macro-level proxemic behaviours of groups. Their model creates spontaneous groups which gather agents with similar motion together. This allows for large-scale avoidance of other agents moving in other directions. Their work is less about groups of people moving together, and more about grouping agents together when their motion/goals are similar, and splitting from them when appropriate. This is useful in examples such as bi-flow along a corridor. There are naturally two types of groups that form, one in either direction, and the model cleanly allows the two different directions of motion to pass each other without much head-on collision, which is visibly observed in the standard ORCA model. He et al.'s model performs proxemic calculations to alter the desired velocity $v_{des}$ which is fed into the ORCA model to perform collision avoidance. In contrast, the OCBM approach both alters the desired velocity directly and adds to the ORCA constraints, which will in-turn result in adjusted resultant velocity.

#### Implementation

Two main objectives for the group behaviour of this implementation are to:

1. create observable group shapes. Such shapes are a 'v' pattern which are known to provide easy communication between members, and a straight line when navigating dense environments.

2. aim to always stay together and avoid splitting or letting obstacles in between.

The most important aspect of the implementation of group behaviours is decomposing group behaviour into mathematical constraints that limit the motion of people. These constraints can then be given a varying hardness per person which equals a person's willingness to maintain group formation.

In addition to constraining the motion of people, this implementation of group behaviour also affects the desired velocity of people in groups $v_{des}$. This is implemented to further encourage people to move together in social groups. This influences where a person wants to move. It can be described using the equation

$$v_{comb} = (1 - u) * v_{des} + u * v_{group} \tag{3.2}$$

where $v_{comb}$ is the resulting velocity a person travels at. $v_{des}$ is the desired velocity of the agent, provided by the navigation part of the simulation and is the velocity desired to reach their objective given no other factors, $v_{group}$ is the velocity of the agent needed for it to maintain its structure within the group. $0 \leq u \leq 1$ is a weight factor which determines the strength of group cohesion. $u = 0$ ignore velocity-related group effects. $u = 1$ ignores the individual velocity, and can result in the group not moving towards their goal. This approach has similarities to other heuristic approaches to group behaviour. One such example is the cohesion part of the boids model [171]. Both this approach and the boids cohesion seek to maintain group togetherness so members do not disperse too far, by altering the velocity of the agents.

We define a variable, referred to as the "ability to communicate" of the agent. The individual value, as well as the relative value to the others in the group helps determine where an agent positions themselves within a group. An individual with high communication within a group will tend to be more towards the centre of the group. An individual with low communication will be towards the sides (relative to the direction of motion) and further forwards. This provides a method for creating varying group distributions and formations. A group with high communication between members will result in a more tight knit group, who tend to walk closer together and less often permit non-member agents to pass between them. In contrast, a low communication group will be more split, and more easily adjust shape when navigating crowded spaces and high density crowds. The value $u$ when determining the velocity is calculated as a normalised sum of the communication of all individuals of the group.

A person has an equilibrium position within the group shape relative to the group centre. They move towards this position with velocity $v_{group}$. The location this velocity points towards can be found by summing over all the locations of members of the group weighted by their communication and finding the average point. This is the centre of the group. A person wants to be a certain amount away from this centre according to $c_{group}/(c * n)$, for individual communication $c$ and number of members in the group $n$. This dimensionless value is then scaled by the number of people in the group by $n^{1/2}$ so the "group area" scales linearly with the number of people. This resulting value $s = scale * c_{group}/(c * n^{1/2})$ is scaled with physical value $scale$ relating to the size of agents. For use in this thesis it is scaled by 0.8 to create close-knit inter-group distances suitable for navigating through large dense crowds.

Within the simulation all members in a group have the same goal and desired speed. This is because people want to walk together within a group. They can have different shape radius to account for physical differences between group members. $u$ is a scaling parameter that adjusts the velocity of people. While an agent is within the group it has no effect and $u = 0$. An agent is considered outside the group if their distance $d$ to the centre is greater than $s$. In this case $u$ scales between 0 to 1 up to the maximum amount which is reached when the distance $d$ is $2s$. The value of $u$ is summarised as follows:

$$\begin{aligned} &u = 0 \text{ for } d < s \\ &u = (d - s)/s \text{ for } s < d < 2s \\ &u = 1 \text{ for } d > 2s \end{aligned} \tag{3.3}$$

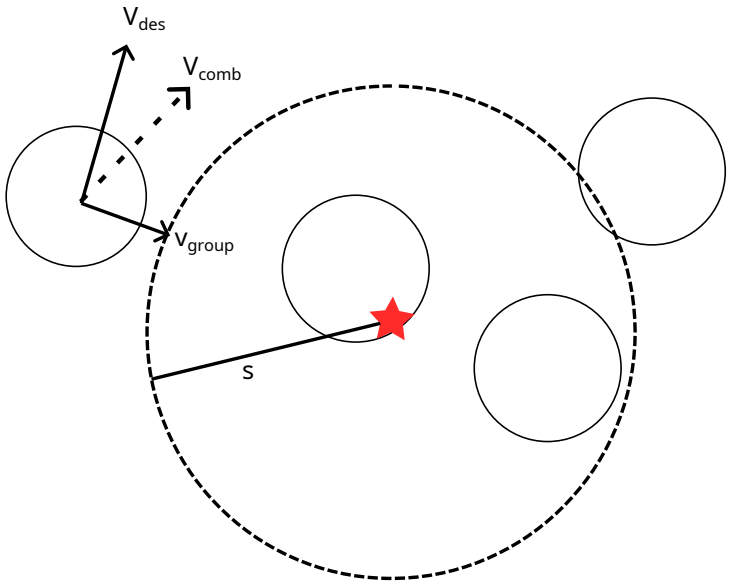



FIGURE 3.3: Velocity for an agent in a group. The red star shows the group communicative centre. The left agent is beyond the equilibrium group position, which is denoted by the dotted circle of radius $s$, so has a velocity component to return to within the area

The velocity calculation is visualised in figure 3.3. It shows the group communicative centre (red star), and the group equilibrium circle (dotted line) of radius $s$. The leftmost agent is outside the equilibrium circle. It has a velocity according to equation 3.2 with a $u$ value according to equation 3.3 as it is within $s < d < 2s$ of the group centre.

Groups should maintain a communicative shape when possible. This is commonly observed as a 'v' shape, where people towards the edge of the group (relative to the direction of motion) are further ahead. A set of constraints is added to any member of a group in order to mimic this observed behaviour. These constraints aim to maintain group shape where possible, but be flexible and permit changing group shapes where necessary. It is two constraints in the shape of a 'v' orientated on the communicative group centre. Figure 3.4 shows this diagrammatically. The crossing point of the constraints intersects the vector joining the group centre to the group goal. It is offset by an amount $f$ and each constraint makes an angle $\theta$ to the line of intersection. These constraints are assigned a hardness for the scenario being simulated. The softer these constraints, the more easily the group shape will deform when other people and obstacles are nearby.

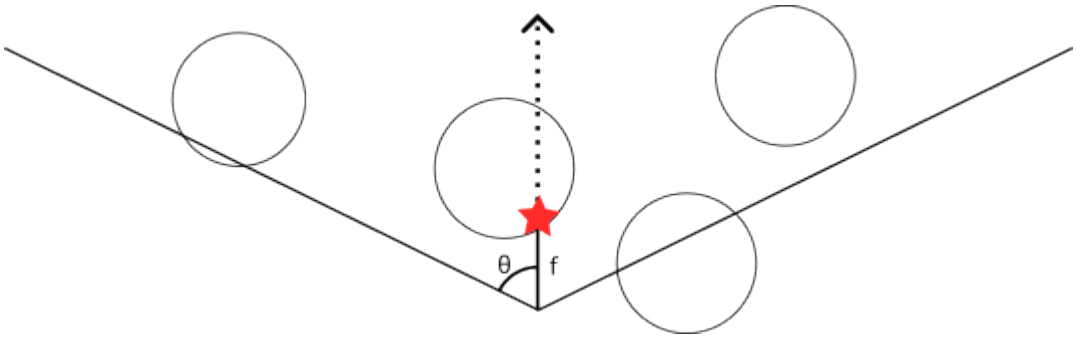



FIGURE 3.4: A group of 4 agents from a computational experiment designed using the OCBM approach and implemented and visualised in SteerSuite. The red star shows the group communicative centre. The dashed line is the vector towards the goal location of the group. Two soft constraints are shown (straight black lines) whose purpose is to maintain a group communicative "v" shape. All agents within this group have these two constraints, in addition to any others due to other behaviour or steering.

### 3.3.2 Luggage

Bags and luggage come in various shapes and forms, and people have different ways of travelling with luggage. This thesis concerns itself with medium-sized trailing luggage, which is commonly of the form of wheeled airport hand luggage. This luggage is carried by a handle, and wheeled along the floor behind a person.

People with luggage can be an important phenomenon to consider when simulating crowds. They can make up a large proportion of people at transport terminals, and have different behaviour to people without luggage. People with luggage move differently, such as slower speed at higher densities [182]. A difficulty is that in scenarios where people travel with luggage (transport terminals, public transport), there are usually additional effects occurring. While there are frequently other behavioural aspects in play, such as looking at timetables at transport terminals, the use of luggage is an important one and is considered in isolation as an example.

The Legion model [13, 19] is able to create people with luggage. In the Legion model, people with luggage are simplified into a singular larger person. The parameters of this singular person with luggage is altered, such as their size, and their movement speed. The simplification of using a single larger person rather than two separate objects is not something that should have much impact on the crowd at low densities, but is potentially an important factor at higher densities. Legion increases the radius of a person by 0.2m for medium sized luggage, and recorded that speed changes (on open flat terrain) from around $1.5m/s$ (UK commuters) to $1.1m/s$ (tourist). Legion documentation does not specify what determines a tourist or if luggage was part of the measurements. It does, however, suggest that people with luggage should occupy more space overall, and move more slowly.

#### Implementation

First, we consider a number of general observations about trailing luggage.

- It will spend most of the time behind the person.
- It cannot go more than a certain distance away from the person, equal to the person's arm length and luggage handle length.

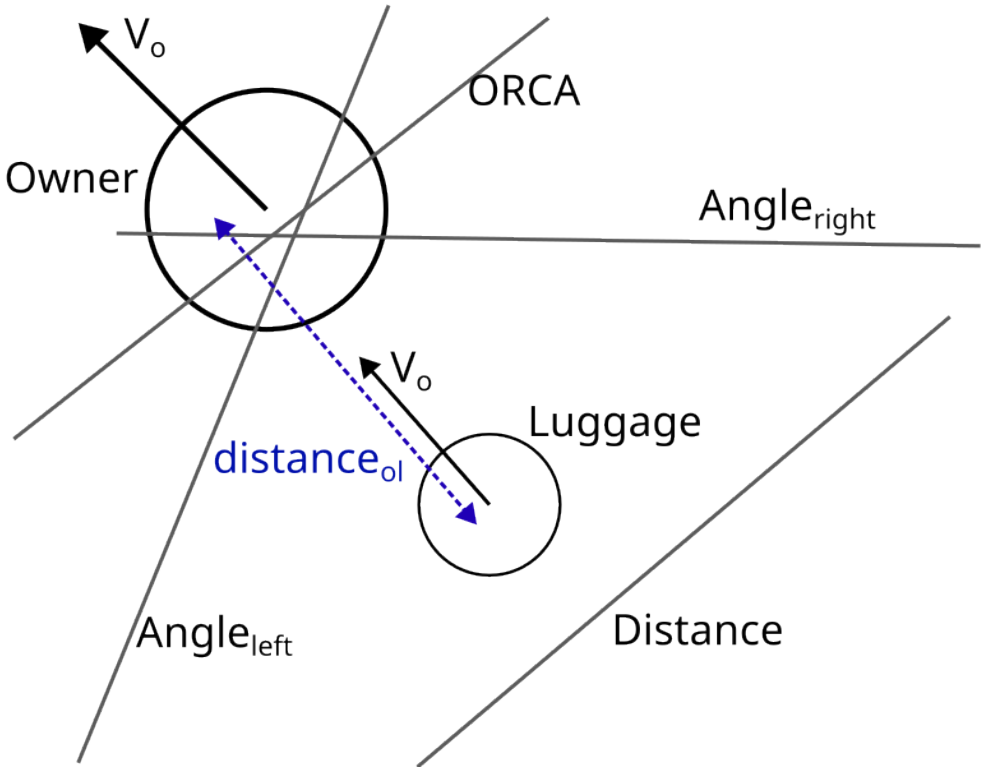



FIGURE 3.5: Constraints caused by owner $O$ on luggage $l$. Owner is moving with velocity $V_o$. Luggage also has a desired velocity of $V_o$, but is subject to change if the constraints do not permit it. Four constraints are generated, one to prevent the luggage and owner colliding (ORCA), one to prevent the luggage moving too far away (Distance), and two to prevent the luggage moving too far in angle (Angle left/right)

- It will, in its natural position, without being near other objects, be directly behind the person. However, when moving through crowds, the luggage can move some degrees from the back angle to aid in avoidance.

This behaviour contains two kinds of agents, one representing people and one representing luggage. Both agents follow the collision rules for regular ORCA agents, but both have novel extensions. Each luggage agent has a knowledge of its owner 'person' agent. Similarly, if a person agent owns a piece of luggage, it knows which particular one it is. It is tracked by assigning a unique ID to each agent, and storing the other agent's ID in the intrinsic information of this agent.

Within the simulation the luggage is given a similar level of intelligence to human agents. This allows luggage to avoid collisions in a similar way to people. In a simulation it helps maintain a consistent level of computation between all agents. It could be possible, if there was need for a distributed computation, to have the owner perform the calculations for the luggage and manoeuvre it accordingly.

Figure 3.5 shows the constraints acting on a luggage due to its owner. These are to simulate the above properties of maintaining a certain distance and angle to the owner. The luggage is labelled as B while the owner is O. The constraints are:

1. One standard ORCA constraint $ORCA_{o|l}$.
2. One half-plane constraint on the opposite side $Distance_{o|l}$. This sets the maximum distance between the luggage and person, and is larger than $r_o + r_l$.

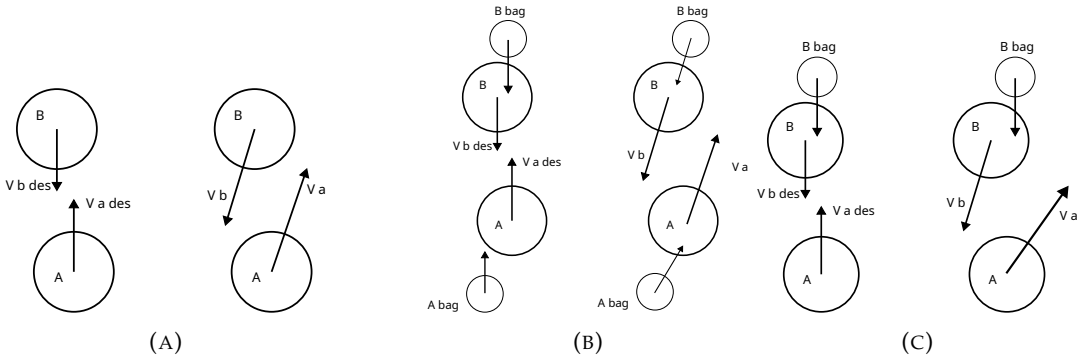



FIGURE 3.6: The different possible individual interactions. (a) person-person interaction. Each takes equal responsibility to avoid the other. (b) luggage-luggage interaction. Both people and bags equally avoid each other. (c) person-luggage interaction. A similar setup to (a) but with a luggage with agent B. Both people avoid each other, but person A moves out the way of the luggage and the luggage adjust its velocity only a small amount.

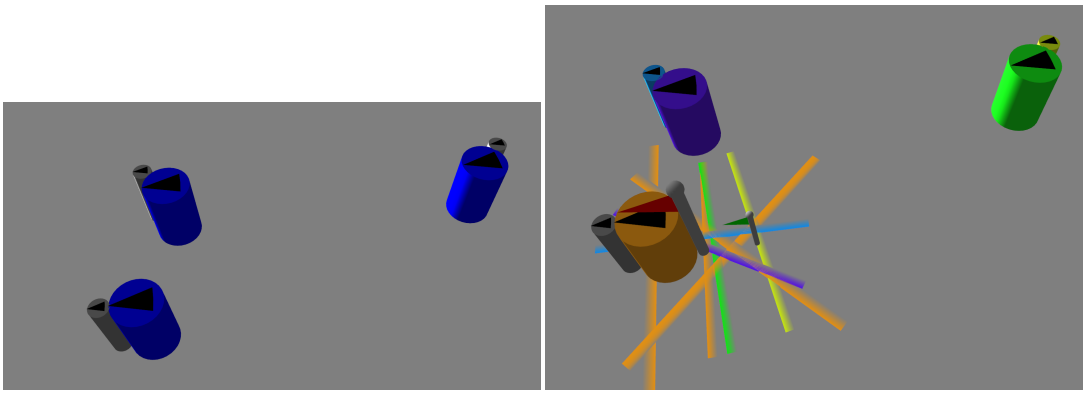
3. Two constraints from the direction of velocity of the owner, both tangential to the velocity vector. These prevent the luggage from moving too far from the angle the person is moving at.

Unlike regular ORCA constraints, which are reciprocal in how people avoid each other (each person takes half responsibility $u = 0.5$), the bags are completely responsible for avoidance with its owner. This is to simulate the fact that people do not avoid collision with their own luggage, but rather move the luggage out of their way when required.

Different individual interactions between bags and people are shown in figure 3.6. For a luggage-luggage interaction, the result is a regular ORCA line constraint, and within the simulation each takes half responsibility for avoidance ($u = 0.5$). For person-luggage interactions, the person takes greater responsibility for avoiding luggage ($u > 0.5$). This is because of the assumption that the luggage is less mobile, hence less likely to move compared to a person.

A potential scenario occurs where a luggage and its owner become separated. This can be observed in large dense crowds, such as people rapidly transferring through busy train terminals with luggage. To account for this within the model, if the owner of a luggage travels too far from the luggage then the luggage can no longer move. In this case the luggage is set to no longer move or steer, as its owner no longer has the capacity to control the luggage. A behavioural rule is added for the owner to immediately make their way back towards the luggage. This behaviour takes priority above any other behavioural rules the owner may have. All other agents need to take full responsibility to avoid such a stationary piece of luggage ($u = 1$).

An example scenario of three people, each with luggage, is visualised in figure 3.7. Figure 3.7b shows the constraints formed on the left-most luggage (grey coloured). The colour of the lines correspond to the agent that they originate from. There is one ORCA constraint formed by each other agent, as well as numerous luggage-related constraints due to the owner (orange). The orange lines are caused by the orange owner including ORCA steering and because it is the owner. The green line is caused by the associated green person to avoid collision with it. The



(A) (B)

FIGURE 3.7: Graphics of luggage. (A), three people heading towards each other, each with a small trailing luggage. (B) the same simulation frame, superimposed with constraints on the leftmost agent (luggage). Here the desired velocity is within an invalid region, shown by the small flag. The closest velocity suitable is shown by the large red flag. The colour of the lines is determined by which agent is associated with them. For example, the orange lines are due to the orange owner.

desired velocity of the luggage is in an invalid region (small flag), and the closest suitable velocity is shown with the large flag.

### 3.3.3 Autonomous Vehicles

This section presents simulating mixed model environments containing both people and autonomous vehicles sharing the same space. The focus of autonomous vehicles here is a shared walking space of people and small personal autonomous vehicles. It is based on the 'LUTZ Pathfinder' pod developed by Transport Systems Catapult [209]. These are different to versions by companies like Google and Uber in that they are specially designed two-seater 'pods'. Experimental trials conducted have been focused on the acceptance of such vehicles, rather than understanding the behaviours of interactions between them.

Understanding the interactions of people with autonomous vehicles is an important ongoing development. Worldwide, countries are adopting and trialling various forms of autonomous vehicles. In the UK, it has been forecast that a market will account for £28bn in the UK alone by 2035, 3% of a global market of £903bn [45].

There is a broad number of domains involving person-vehicle interactions, such as self-driving cars, or personal pods to travel short distances. Each of these domains has their own dynamics and specialities that need to be understood. People will react differently to high speed self-driving cars compared to slow moving pods designed to share the same walking space.

Autonomous vehicles, like cars, have non-holonomic motion. This means a car has two control directions ($v$, $\theta$ (the rotation of the car)), yet has three variables in configuration space ($x$, $y$, $\theta$). Extensions to the ORCA model have been examined in literature which model non-holonomic agents [5, 10]. In order to maintain simplicity of the steering model, the autonomous vehicles in this thesis follow the same rules of motion as regular people (i.e. holonomic motion).

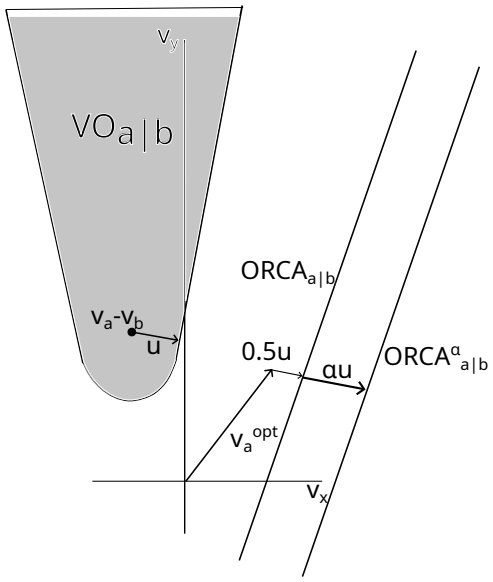



FIGURE 3.8: The use of $u$ in calculating the half-plane ORCA constraint. Two possibilities are shown, $f = 0.5$, and $f = \alpha$ ($> 0.5$). $VO_{A|B}$ is the velocity obstacle formed by agent B on A. $u$ is the vector needed to adjust by to avoid collision. This image uses the same velocity obstacle for two agents as previously demonstrated in figure 2.1b

**Implementation**

The autonomous vehicles and the people interacting with them are given some assumptions here. With regards to the vehicles themselves,

1. The model assumes that the autonomous vehicles are more willing to slow down when approaching a crowd, rather than weave between people

2. The autonomous cars use the ORCA steering method for avoiding collisions

And for people interacting with autonomous vehicles,

1. People will avoid colliding with vehicles with greater effort than any other behaviour.

The importance of avoiding autonomous vehicles is reflected in the calculation of the half-plane $ORCA_{A|B}$ being shifted by $0.5u$. Because the agent takes (at least) half of the responsibility for avoidance, it will assume the other agent will take the remaining half, and hence collision will be avoided between the two agents. A variable parameter $f_{A|B}$, that represents the fractional responsibility that agent A will take to avoid agent B is introduced. This variable is used when calculating the position of the half-plane $ORCA_{A|B} = v_{opt} + f_{A|B}u$. For the standard case of people-people interaction, $f = 0.5$.

Because of the different nature of movement of people and vehicles, the avoidance taken between vehicles and people will not be reciprocal. As such, the weighting of $u$ in Figure 3.8 will be different. Since colliding with an autonomous vehicle could provide greater risk of health and damage, it is assumed people will tend to naturally take more responsibility in avoiding collisions with vehicles (the amount of avoidance is $f_{person|car} > 0.5$). These resulting constraints will be hard constraints so that if collision free motion cannot be guaranteed, people will still prioritise avoiding vehicles foremost.

The vehicles themselves can move in a variety of ways. Because of the reciprocal nature of collision avoidance, it simplifies the simulation if the autonomous vehicles also follow the same steering rules. This ensures that collision-free motion can occur and people can move safely within the environment. When a vehicle must avoid colliding with another car, they will reciprocally avoid each other and each take half the responsibility, i.e. $f_{car|car} = 0.5$.

To ensure guaranteed collision-free motion, $f_{A|B} + f_{B|A} >= 1$. If an agent knows how much another will avoid collisions, they can know how much they should avoid collision. If a person knows that vehicles will not attempt to avoid collisions (i.e. $f_{car|person} = 0$), then people know they are entirely responsible for getting out of the way ($f_{person|car} = 1$). These values are selected before the simulation starts, which will vary depending on how fast and dangerous the people perceive the autonomous vehicles to be, and assume every agent is aware of how much they should avoid others.

## 3.4 Software Implementation and Experimentation of Constraint-Model Behaviour

The previous sections have explained specific examples of behaviours and how they are implemented within OCBM as hard and soft constraints. This section will explore how each of the behavioural models from the previous section behaves within the context of a computational experiment. This section will describe the software implementation of the constraint-behaviour model with the hardness of constraints, and explore its behaviour. It will perform computational experiments to test the proposed behaviours, testing each behaviour individually. The behaviours will be tested in the same order as they were introduced: i.e. groups, luggage, and autonomous vehicles respectively.

Testing the model and behaviour performance will provide an idea of how the various models function. This helps understand whether the framework is able to reasonably model the behaviour by using constraints, which is part of the contribution of the chapter.

### 3.4.1 Software Implementation of Modelling Behaviours

The OCBM approach and associated models were implemented in the SteerSuite software framework [188]. This framework is designed for creating and evaluating pedestrian models. It provides a modular way to create and extend common behaviours, such as collision avoidance. SteerSuite comes equipped with tools to extract parameters from people, and the model as a whole. It tracks variables of individual people, such as position and velocity, and calculates useful values, such as the time taken for a person to reach its goal. The tools of Steersuite were extended to track pedestrian behaviour within a certain area only. This is useful to measure particular locations without accounting for edge effects. By tracking velocity of individual people, and the local densities of those people within a specified area, fundamental diagrams of the simulation can be constructed. Fundamental diagrams often use charts to display information of pedestrian motion within scenarios. More detail is provided in the literature in section 2.4.2. Fundamental diagrams plot the velocity (or flow) against density of people. The shape of the curve depends on many factors including the environment, the mobility of the people, and even the culture. It provides a simple way of describing a scenario visually.

The visualisation of models using autonomous pods are obtained within the Unreal Engine. The Unreal Engine comes built in with many powerful features for graphically realistic 3D rendering of virtual objects. Simulations are still executed within SteerSuite. The simulation results from SteerSuite are fed (in real-time) into Unreal to set the values of the people and vehicles. Unreal manages the animation of people, and the visual appearance of the objects within the simulation. It is used here to better distinguish the autonomous vehicles from people, as well as provide the model the autonomous vehicles are based on.

All the experiments are configured using a common approach. Multiple associated start and end regions are chosen, such that agents are spawned in a start region with a target in an associated end region. Random spawn locations are chosen so that there is no overlap with other people within a certain time period based on person size and speed. Within a simulation, each agent has a goal location to aim for. The agent's velocity is in the direction of the goal location, scaled to the movement speed. Once an agent reaches the goal location they are removed from the simulation. Once all agents have reached their goal the simulation is ended.

The following experiments assess the functionality of the three behaviours as part of the OCBM. The objective of these experiments are to demonstrate how OCBM and the description of the behaviours as constraints translates to emergent crowd observables.

### 3.4.2 Assessing Group Dynamics

In order to test the proposed constraint modelling of groups, with the addition of hard constraints, a demonstration of the functionality will be explored experimentally. The example scenario chosen is a bi-flow corridor. This scenario is conceptually simple but can demonstrate different behaviours between different implementations.

A bi-flow corridor is constructed like a regular corridor, with parallel walls along two sides. People move in bi-flow, that is to say, two different directions of flows. For a corridor aligned left-to-right, the two flows are left-to-right, and right-to-left. This results in head-on motion between the two flows. There is an observable phenomenon in bi-flow corridors known as lane formation. This is where people moving in the same direction tend to follow another person moving in the same direction. This occurs because it speeds up the flow of the individuals, and the crowd as a whole. This is observable in physical crowds, and is a common emergent behaviour tested of microscopic pedestrian models. The ORCA steering model is able to demonstrate this phenomenon [59].

The group shape and dynamics of the group itself can be measured by exploring the angle between members of a group. These angles can provide insight into the positions people within a group make, and how the group shape changes in different scenarios. This angle is measured as the angle between normal of the group centre to the goal, and the vector joining the two agents. This can be observed within bi-flow corridor as well. A small angle means that people within a group are walking side-by-side to each other. This is approximately the desired shape a group wants to take, because it is a useful shape for communicating between each other easily. It is, however, difficult to navigate dense environments in this way. A large angle means that agents are walking in a lane formation, with one behind another. This shape is more suited for moving though dense crowds, and hampers the ability for people to communicate so is not the natural shape they tend to. The resulting angle is an average of each agent's angle to its nearest neighbour. If two agents are both

closest to each other, the angle is recorded only once to avoid double counting the same value.

### Experimental Configuration

The cross flow corridor experiment is simulated in a virtual corridor 6m wide. The corridor is 20m long, but only the central 10m of the corridor is observed. People are spawned into the simulation at either side of the corridor, in the final 2m length. The diagram of this setup is shown in figure 3.9.

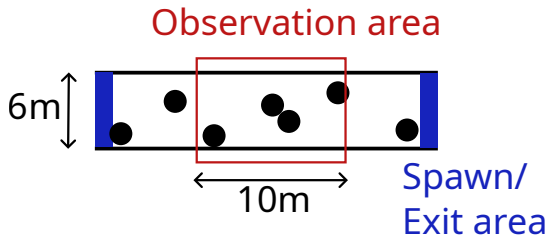


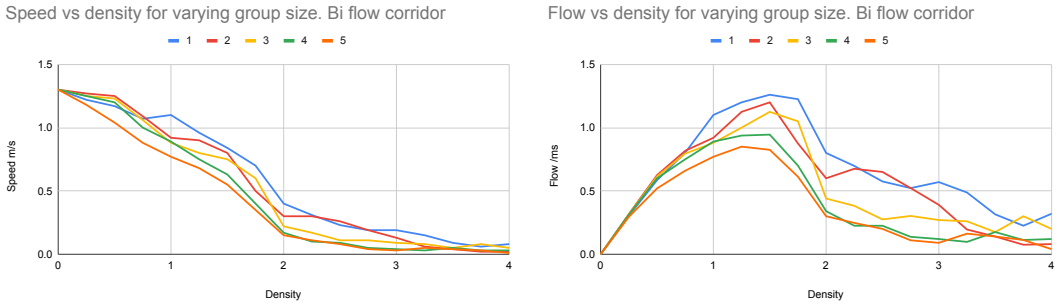

FIGURE 3.9: The biflow corridor setup. The red rectangle, 6m by 10m, shows the area in which measurements are recorded. People are spawned within one of the two blue rectangles and navigate to the opposite blue rectangle.

For each simulation a fixed group size is selected. Groups can only be of the specified size and all agents must be part of a group. Multiple simulations are run to attain repeat measurements over different densities, and each run uses a different random seed. Group sizes of up to 5 are chosen. A group size of 1 is equivalent to no group behaviour.

### Experimental Results

Figure 3.10 shows the mean speed-density and flow-density diagram for varying group sizes. Each data point corresponds to the mean of 500 measurements. As can be seen the increase in group size reduces the speed of moving through the corridor. This occurs because the constraints caused by being part of a group tend to result in slower motion. The more people within a group, the more severe the impact of the additional constraints and rules. Note, however, that for low density without many other agents, the people move at the same speed regardless of how large the group size is. This is in contrast to certain literature [97], but can be set and adjusted as input parameters to the model if such behaviour should be desired. Similarly, from figure 3.10b there is a general trend that when groups contain more people the flow will be similarly reduced.

Figure 3.11 shows how the group shape changes with density in the bi-flow corridor. The angle is divided by $\pi/2$. An angle of 0 ($/\ (\pi/2)$) means two people in the group are walking side-by-side. An angle of 1 ($/(\pi/2)$) means the two people in the group are walking in front of the other. The figure shows approximately an angle of $\pi/2$ for low density, which is in keeping with the 'v' shape of the group formation. No matter the group size, this shape is chosen for low density. As density increases, the angle increases. With a larger angle, people are walking more in front of other members of the group, resulting in a more line/snake like group formation. This behaviour is similar to lane formation, which is a commonly observed phenomenon in bi-directional crowds. For smaller groups there is a more obvious shift between walking side-by-side and one-after-another, occurring around a density of $2.5/m^2$.




(A) (B)

FIGURE 3.10: Bi-directional flow with varying group size (see legend) for density against (a) speed and (b) flow. Each data point is the mean from 500 observations.

For larger groups the members are more spread out, by virtue of being more numerous, so the average of the angle over density transitions more smoothly. Above a density of $3/m^2$ all tested group sizes are walking more front-to-back, rather than side-to-side, as the the angle is greater than 0.5 ($/\pi/2$).

### 3.4.3 Assessing Luggage Dynamics

#### Experimental Configuration

In order to test the behaviour of luggage dynamics two experiments are considered. The first takes place in a bi-flow corridor. The second in a four-way corridor. The bi-flow corridor results in head-on movement and helps understand scenarios where the most likely interactions are moving in the opposite direction to other people. In addition, a four-way corridor experiment is also examined. It is the intersection of two bi-flow corridors with each other at 90°. This setup is chosen to explore the interactions for both side collisions and head-on collisions and helps understand scenarios where collisions between agents can occur at any angle. These different collision types are useful to explore due to the asymmetry of people with luggage. A person with trailing luggage has a different cross-section when viewed from the front compared to the side (where the trailing luggage extends the cross-section). People in the four-way corridor setup spawn at any of the four corridor edges, and move to the opposite end of the same corridor. Figure 3.12 shows the diagrammatic setup of the four-way corridor.

The experiments seek to compare not only the impact of luggage on the simulation, but also the method of modelling luggage. The proposed method introduced within this chapter simulates people and luggage as separate entities. An alternative approach is to simulate a singular larger person which accounts for both the person and their luggage [19]. In order to compare and contrast the approaches, the experiments are repeated for various pedestrian shapes. See 3.13 for a visual distinction between them. A singular agent of size $r_{small}$ is the default behaviour as if luggage was not accounted for. A singular agent of size $r_{small} + r_{bag}$ aims to simulate the alternative approach. These agents have a larger-than-average size to account for the luggage within themselves. The two separate agents $r_{small}$ and $r_{bag}$ are for the novel proposed behaviour.

The experiments use varying agent types:

- The OCBM version with luggage

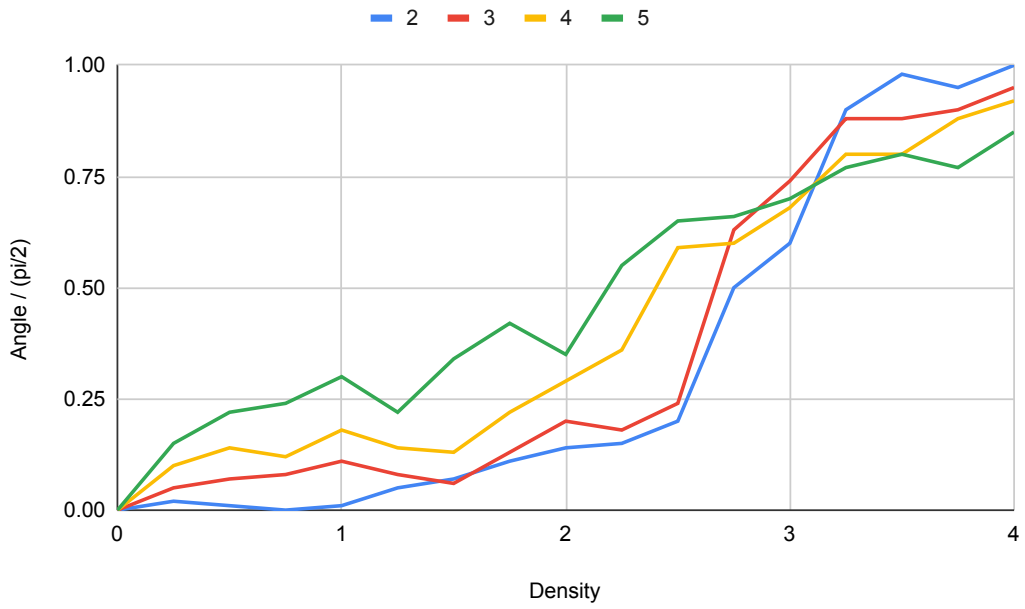



FIGURE 3.11: Angle within groups vs density, for varying group sizes between 2 to 5

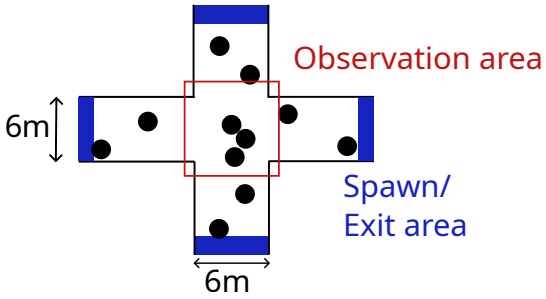



FIGURE 3.12: The four-way corridor setup. The red square encapsulates the recording area where measurements are taken. People spawn in one of the blue rectangles and navigate towards the opposite side of corridor

- A singular person of radius 0.5$m$
- A singular person of radius 0.6$m$
- A singular person of radius 0.7$m$

The OCBM version with luggage, labelled "constraints" uses 0.5m agents with 0.2m luggage, and is the novel proposed behaviour. The standard ORCA model is used for modelling the other types of people, which is equivalent to the OCBM without additional behaviours. People of size 0.5m are tested for simulating behaviour without luggage. An intermediary "large" agent is also tested, at 0.6m radius. This value is used as a similar area of circle to a small person with a luggage (area $\pi * 0.559^2$), and rounding the 0.559 to 0.6m radius. This value is used as an intermediary to show the transition of values between 0.5 to 0.7m. People of 0.7m radius are also tested, chosen as a combination of 0.5 + 0.2m. 0.7m people do not have a separate luggage and are modelled as a singular shape which encompasses

the person and luggage. Simulations are repeated to obtain 500 observations for the bi-flow corridor, and 200 for the four-way experiment.

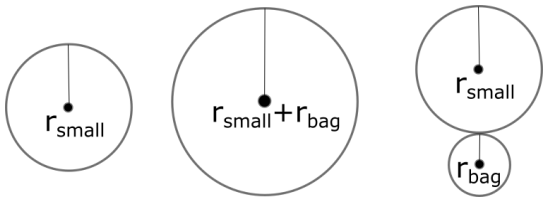



FIGURE 3.13: View of regular sized person, large person, and person with luggage. A large person is approximated by $r_{small} + r_{bag}$ to encapsulate both the person and the luggage within the same circle.

### Experimental Results

**Bi-Flow Corridor** Figure 3.14a shows how the average person's speed changes over varying density for the single bi-flow corridor. Four sets of results are illustrated. The data labelled as "Constraints" is the data of OCBM with luggage introduced previously. Data labelled as "large" represents simulating large ORCA people of various radii.

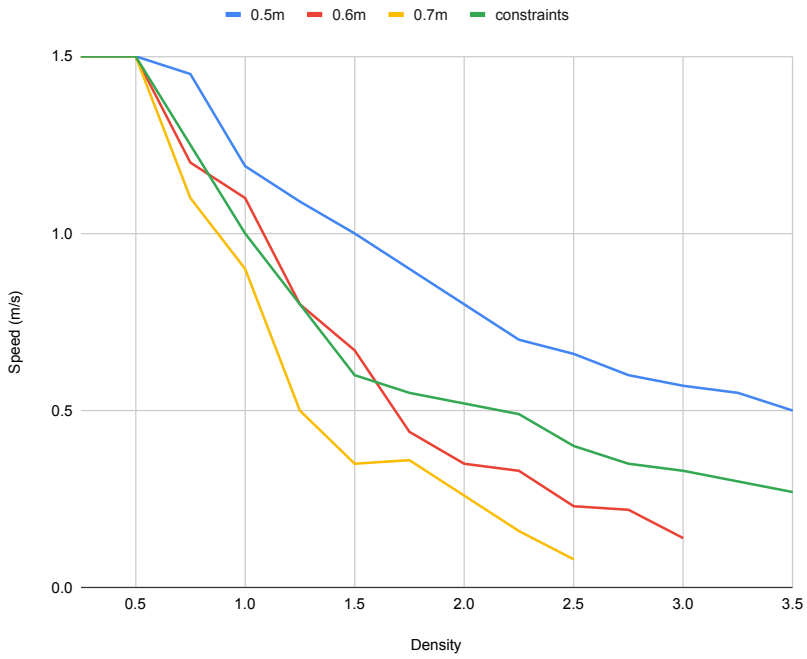
The similarity of the curves suggest that the behaviour of singular larger people is similar to those of singular people of radius $0.5m$. The curve of OCBM with luggage does not correlate to the other curves. This shows that the two different ways of representing people produce different emergent outcomes. Since people with bags, when viewed head-on, are of radius 0.5m, due to the luggage being hidden partially or mostly behind them, this results in faster speed and flow through the corridor compared to simulating people of larger size.

Figure 3.14b shows the fundamental diagram of the bi-flow corridor for flow rate, which is a measure of how many people pass through a location. The OCBM model ("Constraints") shows faster flow behaviour compared to simply simulating larger people, i.e. the flow rate is larger for increased density meaning that the model better captures the ability for people to pass by each other in dense crowds.

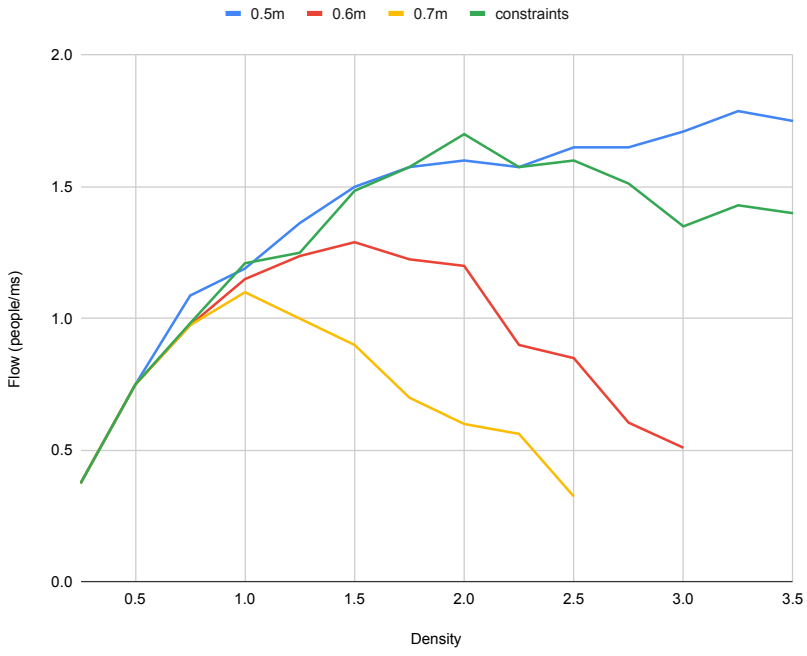
The addition of luggage should not affect the speed of people's movement [98, 58]. As such the speed of people at low densities should be the same. Changes in simulations are due to statistical fluctuations. Both the separately modelled luggage and larger people effectively cause a reduced density due to luggage taking up space but not being associated with person density.

A possible reason for the difference in results between different agent types is thought to be caused by the frontal cross-section of the person. Larger people have more frontal cross-section (0.6 or 0.7m compared to 0.5m) which reduces the number of people that can pass each other in the corridor. This makes it harder for larger sized people to move through the corridor, and these people move more slowly for lower density. This is compared to the OCBM luggage model which behaves more similarly to people without luggage, albeit with shifted density due to the additional headway due to the presence of luggage.

These differences can be seen as the OCBM luggage model follows a similar shape to people without luggage, but shifted for reduced density. In comparison, larger people display a different speed-density relationship, which can be attributed to the larger frontal cross-section.

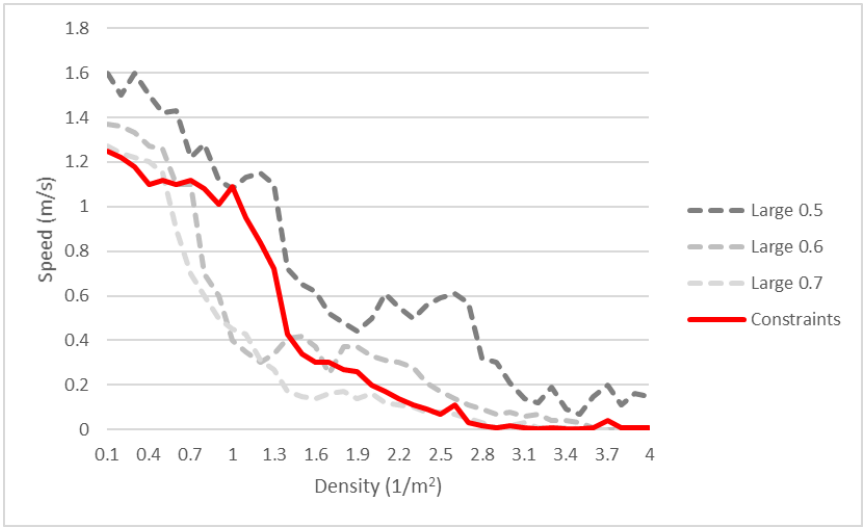



(A)

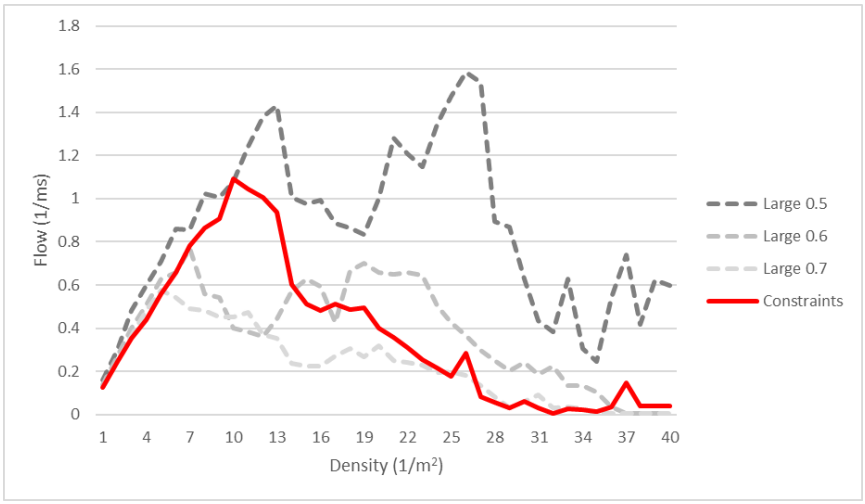



(B)

FIGURE 3.14: Various simulation models and experimental data in bi-directional paths: (a) Shows the speed-density relationship, (b) shows the flow-density relationship. Each data point is the mean of 500 observations.




(A)




(B)

FIGURE 3.15: Experimental data in cross-directional paths: (a) Shows the speed-density relationship, (b) shows the flow-density relationship. Each data point is the mean of 200 observations.

**Four-Way Corridor** The next experiment examines the four-way corridor, and the results are shown in Figure 3.15. This experiment shows that for small densities ($< 1.6$ people$/m^2$) the speed of the people in the constraint model is faster than larger people. However, when the density rises the speed of people matches better to those of larger people of size 0.7m. This means that the proposed model transitions from a speed-density pattern similar to people without luggage at lower density to a pattern similar to large people at higher density. This can be understood as for low density the separate luggage provides little impact on the interaction and flow. For higher density the sideways cross-section of the agent with luggage is very similar to an equivalent larger agent, and a similar speed-density behaviour can be observed.

**Results summary** The results show that the model of people with bags, compared to considering single larger agents, shows different behaviours when examining the speed/flow-density relationships. For scenarios which involve head-on collisions, in the bi-flow corridor experiment, the proposed model uses a reduced head-on

cross-section compared to a larger agent, and this results in larger flow since people and luggage take up less corridor cross-sectional space. In the four-way corridor experiment, where crossing agents approach the extended longitudinal (side-on) cross-section of agents travelling across their path, for high density, there is less difference between the “large person" and OCBM luggage (i.e. “constraint") models. This can be understood in that the longitudinal cross section is the same size, and this fact is more important than the reduced head-on cross section. Within lower density the ability to manoeuvre luggage allows for better flow compared to larger agents.

These results show the effects on speed and flow for varying densities, when considering different representations of people. The model shows a transition of speed-density behaviour between normal-sized people without luggage to large people with increasing density when side-on collisions are the dominant interaction. Therefore, for scenarios where head on flows are a common occurrence, it is important to consider luggage as a separate entity rather than an artificially enlarged person size.

### 3.4.4 Visually Assessing Autonomous Vehicles

The behaviour of interacting with autonomous vehicles is a complex and developing field. The OCBM with autonomous vehicles is tested by using it as a tool for visual validation by experts. The visual running of the model in real-time allowed experts in the field of developing autonomous vehicles to easily view and explore the interactions. With this they could evaluate if the simulation appeared to exhibit movement that they expect. This visualisation behaviour was shown to engineers at the Transport Systems Catapult[1], a company involved in the development of autonomous vehicles.

Two visual experiments are created. The first experiment is a 2-way crossing, and the second experiment is a four-way crossing. Both experiments are in a large open space (without corridors) of people interacting with autonomous vehicles. The 2-way crossing has people moving left-right or right-left, while the 4-way crossing has people moving in cardinal directions. The 3D model of the vehicle is that of the “Pod", used by the Transport Systems Catapult [34].

The avoidance responsibility values are set such that people are entirely responsible for avoiding collisions with the autonomous vehicles. This is mathematically represented as $f_{person|car} = 1$, $f_{car|person} = 0$. To demonstrate the heterogeneous nature of the agent’s individuality each agent is assigned a colour.

The 2-way crossing experiment has the two crowds attempting to pass amongst each other to reach their destination. Figure 3.16 shows the visual result of this. An inset is provided which shows the eye-level view at the same instant. In addition, the bird’s eye view shows the arrow of motion of each person. The arrow is colour-coordinated with the colour assigned to the associated person. During the simulation people leave a larger space for autonomous vehicles than they do for other people. This is due to the additional responsibility people take to avoid colliding with cars.

The second experiment, the 4-way crossing, is visualised in figure 3.17. Each crowd must navigate directly across the environment, causing avoidance of not only head-on agents, but also side-on. Subfigure 3.17a again shows a birds eye view with people’s colour assigned to the arrow. Most arrows can be seen to be pointing in one of the four cardinal directions. Subfigure 3.17b shows the simulation at a different

[1]Part of the funding provider for this thesis grant “Accelerating Scientific Discovery with Accelerated Computing" (grant number EP/N018869/1)

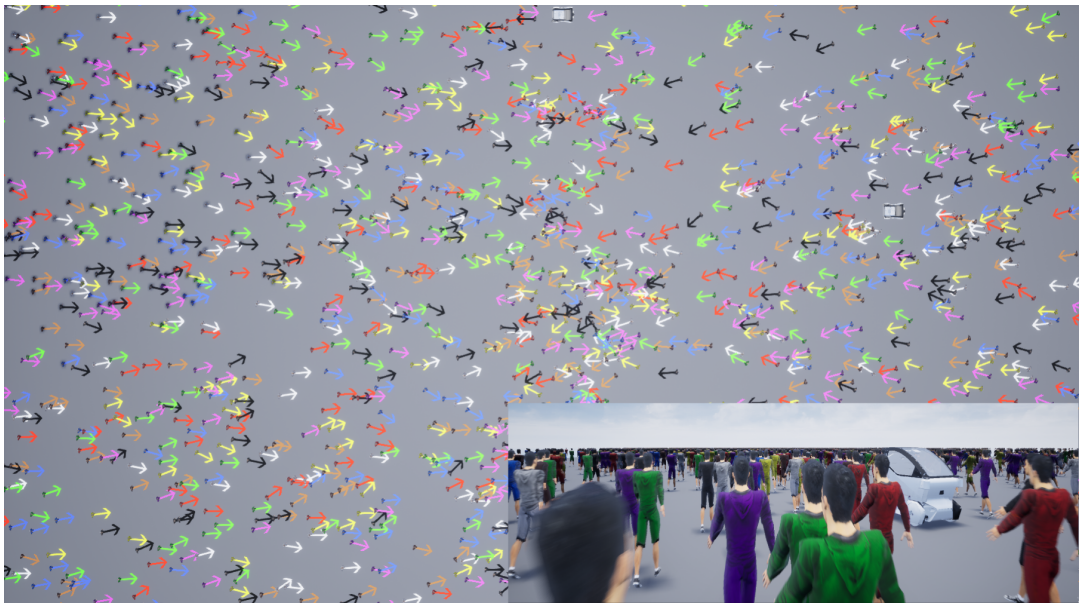


FIGURE 3.16: Visualisation of 2,500 agents in Unreal Engine. Two crowds navigate past each other, one heading from left to right and the other heading from right to left. Inset: an eye-level view of the simulation, taken at the same time

moment without arrows of motion. Pods can be more clearly seen. A circular region around each pod exists where people do not enter often. This radius is related to the lookahead time. Subfigure 3.17c is an eye-level view close to a pod. This view provides another view of the distance where people avoid walking close to pods.

During these experiments the autonomous cars are able to move steadily in their intended direction. There are cases where clusters of people form where the pod is aiming to move. In this case the pod slows down until the density of the cluster reduces and people can more easily move out the way.

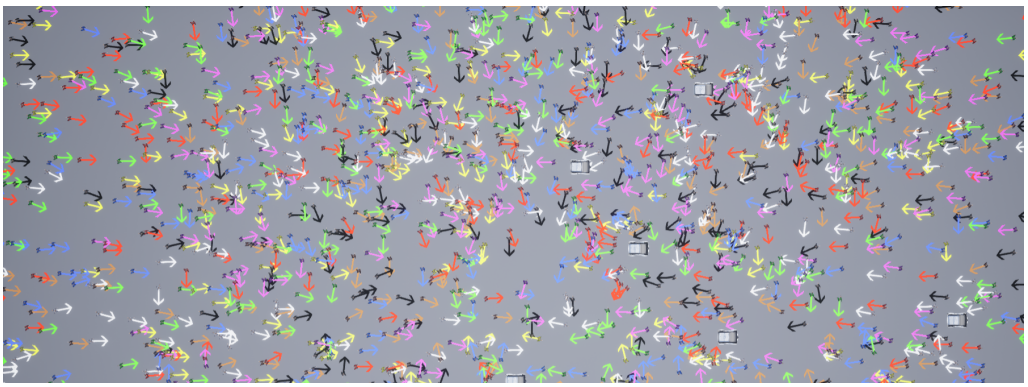
These experiments were shown to engineers working on autonomous vehicles at the Transport Systems Catapult who appreciated the visual clarity provided by the experiments and thought the interactions appeared plausible in their opinions.

### 3.4.5 Discussion

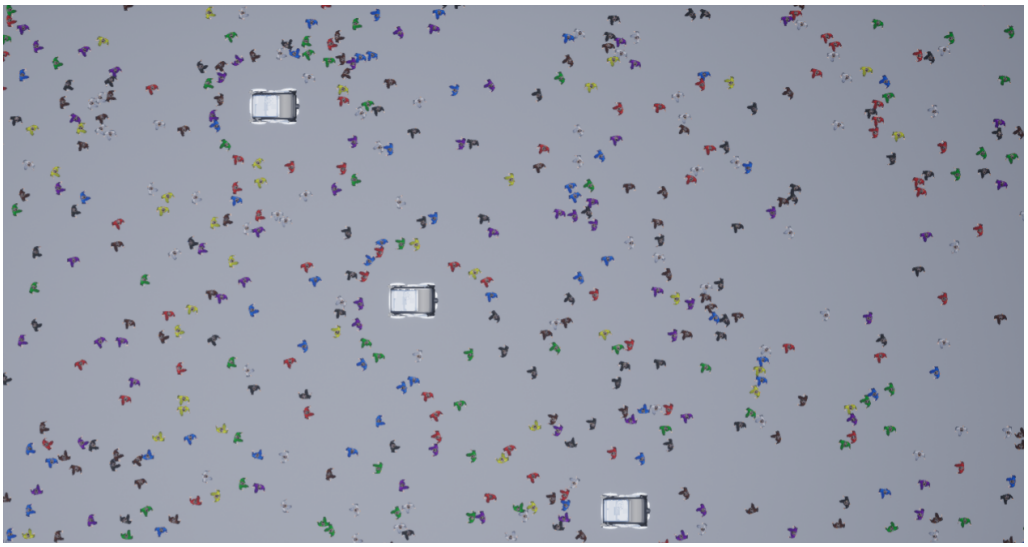
The experiments of this section have explored the behaviour of the OCBM applied to three behaviours. Each behaviour has been individually tested and explored. The results have demonstrated that OCBM can simulate different behaviours and display different emergent behaviours depending on the behaviours used.

Groups were tested with sizes up to 5 people within a group. It was observed that the larger the group, the slower the crowd speed. The groups are able to display a transition from walking in a 'v' shape to a column at high density. Luggage was modelled using two separate entities (owner and luggage), rather than a singular larger agent. It was shown that experimentally this can result in different emergent behaviour. The results make no attempt to suggest one is more realistic to another.

The results of the luggage experiments have demonstrated differences in emergent behaviour between the OCBM with luggage approach, and to singular people. These differences are noticeable in the four-way corridor, and it is theorised this is due to the difference in cross-sections of the different agent types tested. It can be concluded because of this that any scenario that involves interactions other than the



(A)



(B)

(C)

FIGURE 3.17: Visualisation of 2,500 agents in Unreal. Four crowds navigate past each other, two heading between left and right and two between top and bottom. a) Shows a zoomed-out view, with arrows showing the motion of each agent. b) shows a zoomed-in section of the above image. c) shows an eye-level view of the simulation

head-on direction might benefit from the OCBM approach with people and luggage modelled separately. This is because the head-on directions have similar cross sections, but any other direction will have a different cross-section. Future work would require human experimentation to validate either approach further.

This section has shown how the OCBM behaves, but it has not provided insight into how the underlying simulation implementation performs. This can be understood through examining metrics of the simulations and performance timing of the code, which will be examined in the next section.

## 3.5 Performance

Performance is important to understand to know what domain the OCBM can be applied to. If the code can run faster than real-time (i.e. the code runs faster than the simulated time) then it can be suitable for real-time applications like interactive visualisation. It is also an important first step in creating optimised, performant code, by highlighting potential areas for performance improvement.

In order to understand the performance of the OCBM implementations, performance experiments will be carried out. The experiments will explore the speed of code execution. It will do so by examining how simulation parameters affect the speed, as well as how important different parts of the code are to the time taken. The faster the code runs, the quicker the same simulation will complete. This section can be highly affected by particular factors like compute hardware. All the experiments are carried out without visualisation nor rendering, so that only the time for computation is examined.

Section 3.4.1 previously explained how the model was implemented into code. Because of the modular framework of SteerSuite, it is easy to isolate individual routines to measure the time taken to compute them. The hardware tested was an Intel I7-4790k CPU with 16GB RAM. The code was run on a single core.

### 3.5.1 Experimental Performance Configuration

#### Scaling by Population Size

Perhaps the most fundamental performance metric is a measure of how long a simulation step takes with varying numbers of people. Although this can vary depending on the experimental configuration of the hardware it provides insight into the limit of the maximum number of people that can be simulated. Such a factor is important for real-time simulation and interactive visualisations. More people in the simulation should result in more time required, and this experiment aims to quantise how much it increases. A model scaling experiment is conducted to understand simulation run time of the OCBM for larger crowds. This experiment is set up by increasing the scenario size proportionally to the number of agents, and setting up goal locations to keep the density between the scenarios as consistent as possible, in this case around 1 person/$m^2$. Average density was tracked for each iteration of the simulation over all agents, and those iterations that had density varying more than 10% from $1.0 + -0.1$ were ignored. Maintaining a consistent density is important to ensure only the change in population is affecting the results. In this experiment no additional constraints are modelled, only steering. All people have the same parameters (except for goal direction). There are no environment obstacles. The experiments were performed on an I7-4790K Intel CPU with 24 GB of RAM and was consistent for all performance experiments.

### Examining Interaction Density

Each person in the simulation has a limit on the number of other people they will consider avoiding. This is partially due to computational optimisations, to reduce the number of calculations, and also due to psychology: only people near to a person are of interest for collision avoidance. Increasing the number of considered neighbours past a point will not affect the simulation results, but will slow down the computation. Too few neighbours will result in unrealistic movement and potential collisions. This similarly applies to behaviours like other bags.

This experiment aims to find how much time is required to consider additional constraints. It also seeks to answer whether constraints created from different sources (steering, luggage) cause different changes. For this, agents are packed incredibly tightly. The number of neighbours they consider is varied by increasing the lookahead radius ($\tau$) and the maximum number of neighbours. The maximum number of neighbours is the only value directly having an effect. The lookahead radius ($\tau$) is changed because there is a limit to how many agents can be within the nearby radius, and altering only the radius value has little impact by itself.

This experiment ignores the calculation to find a feasible velocity if there is no valid solution to the initially constructed linear problem. This is because performance of this calculation can vary greatly depending on whether a feasible solution can be quickly found or not. This means that people are not behaving correctly. However, this is a reasonable approximation of performance. This is because the experiment measures the impact of constraints on the performance in the case of a valid solution. This is a much more common scenario, and the results of this experiment will be applicable to those scenarios.

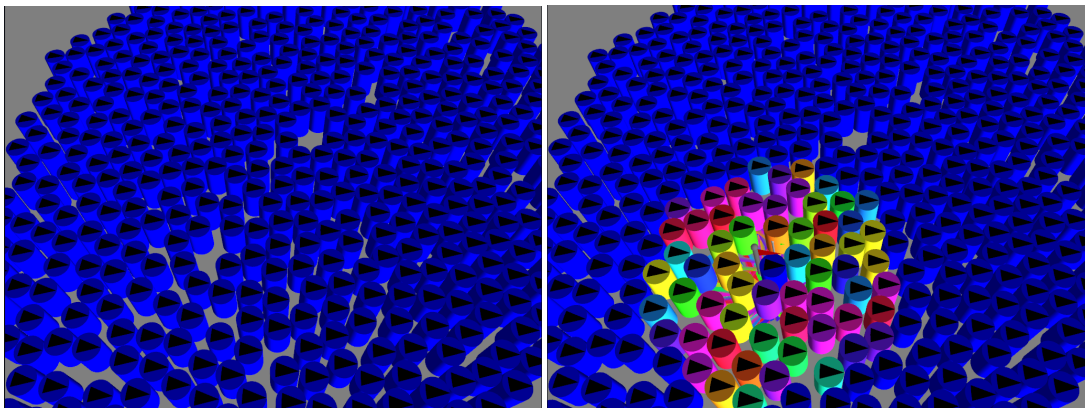
An example setup is visualised in figure 3.18. It shows a large number of people tightly packed in a crowd. The scenario is close to the maximum possible packing density. Figure 3.18b shows the neighbours (coloured) for a blue agent of interest, in the central part of the image. A visual way to think of this experiment is that it changes the number of coloured agents, and records the time taken.

This experiment uses an open environment without obstacles. It spawns between 100-500 people at random locations who all have the same goal location. This results in everyone moving to the same spot. People do not leave the simulation if they reach the location. Recording outputs does not happen for 60 seconds to ensure a stable configuration of crowding is formed. Recording happens for 120 seconds. If group behaviour is enabled, everyone is in a group of 3. If luggage is enabled, everyone has a piece of luggage.

### Examining Model Complexity

This experiment explores how the performance changes when including the additional group and luggage behaviours. The effect can be observed by timing the individual routines and observing differences. In order to separate out effects of density and considered neighbours (which are measured in the previous experiment), three distinct densities are maintained for the test. The densities are chosen as low, medium, and high density with approximate values of 0.2, 1.0 and 2.0 people / $ms^2$ respectively. In order to separate out the effect that more people in the simulation increases the time, the resulting time will be divided by the number of people in the simulation to obtain a time that is independent of the number of people simulated.

This experiment uses an open environment without obstacles, and controls the spawning rate to achieve desired density. Between 100-500 people are randomly



(A) (B)

FIGURE 3.18: Graphics of large numbers of people. (B) a blue agent in the middle of the coloured agents is selected, all the alternatively coloured agents are those considered as neighbours to blue. In this figure 60 neighbours are highlighted in different colours.

simulated each time. People are spawned in one of two regions, east or west, and have a goal location in the other. This results in a cross flow. The density can be controlled by a combination of changing the number of agents spawning in each region, and changing the size of the spawning/goal areas. All people are in a group size of 3, have luggage, and an additional 5% of the population are autonomous vehicles (between 5-25 agents).

The routines are separated into particular functions. They are "constraints", "lp2", "lp3", "move", "groups", "bags", "autonomous":

1. "constraints" is the routine for examining nearby neighbours and generating ORCA steering constraints.
2. "lp2" is the routine for solving the linear program formed by all the constraints. It is possible that there is no feasible solution to this problem. In which case the routine "lp3" is called.
3. "lp3" is called when there is no feasible solution to the linear program. In this case there is no velocity that satisfies collision avoidance and behaviours. lp3 accounts for soft and hard constraints, and shifts them until an acceptable solution is found.
4. "move" is the routine for updating the position of the agent each step.
5. "groups" is the routine for constructing constraints related to group behaviour.
6. "bags" is the routine for constructing constraints related to luggage behaviour.
7. "autonomous" is the routine for constructing constraints related to autonomous behaviour.

### 3.5.2 Performance Results

#### Scaling by Population Size

The first experiment shows how the performance scales with the number of people. The results are shown in figure 3.19. A linear trend line of best fit is plotted. It shows

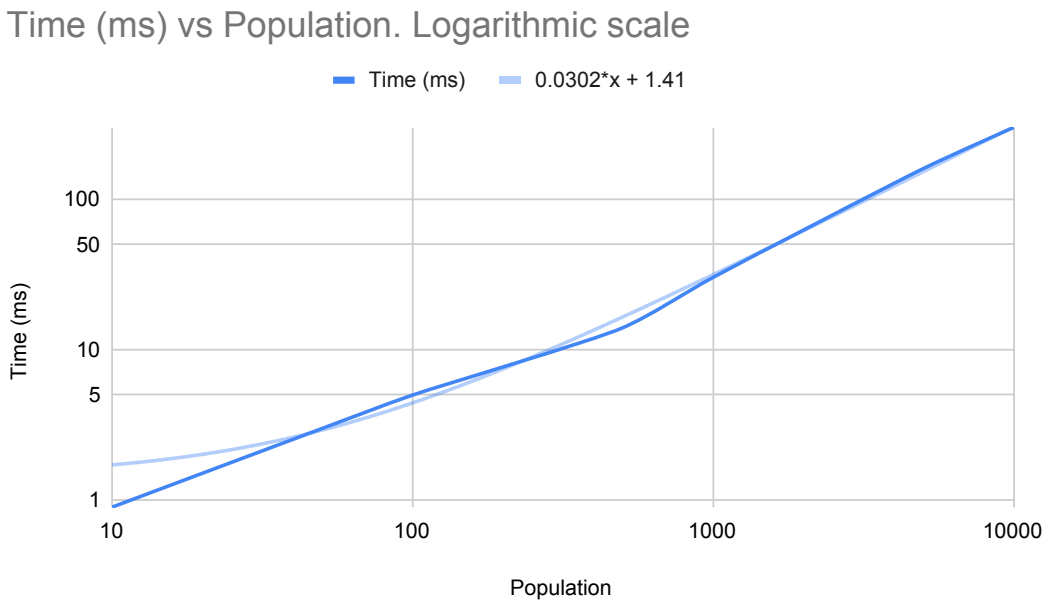



FIGURE 3.19: Plot of population vs frame time in milliseconds. People are at a density of around 1 person/$m^2$ in a large open space without other obstacles.

an approximately linear increase in time taken as the number of agents increases. Each agent adds an additional 3 microseconds per iteration. This shows that the model performance is approximately linearly dependent on the number of agents. For this set-up a maximum limit of 400 agents are required for simulating faster than 60 frames a second, and around 1000 for 30 frames a second. This provides an approximate limit of 400-1000 people for interactive real-time simulations. Each data point is the mean of 1000 seeded runs. The standard deviations are all within 2% of the corresponding mean time.

### Examining Interaction Density

Figure 3.20 shows the scenario of incredibly dense agents and the varying of the number of neighbours considered. As can be seen, the number of considered agents has an approximately linear impact on the time taken, with each additional neighbour adding in 7 $\mu$s for a computation frame. Each data point is the mean of 1000 seeded runs, and the standard deviation is less than 4% of the corresponding mean.

The complexity of computation per agent can be extended by increasing the number of neighbours they consider. Doing so increases the number of constraints each person can consider, provided there are more neighbours than this limiting value. This is representative of a more complex model (i.e. additional behaviours) with more constraints. This is because a constraint is treated the same in the code regardless of whether it is due to a neighbour or caused by a behaviour.

This shows that time increases for more constraints. It does not increase differently when due to steering, luggage, or groups. This means that the speed for solving the linear problem depends only on the number of constraints, not the behaviours that caused the constraints.

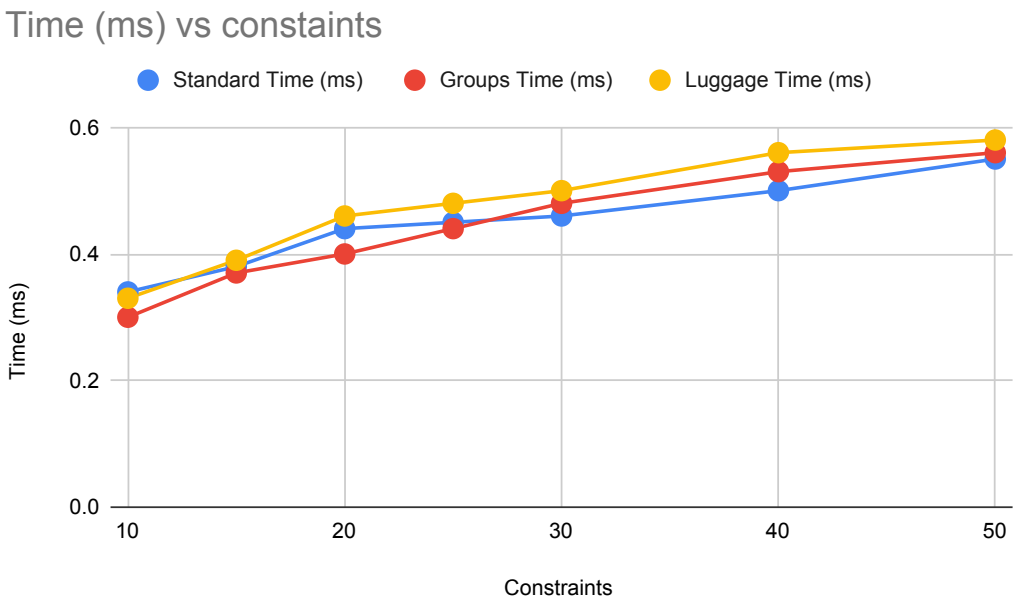



FIGURE 3.20: Plot of time taken per frame against number of neighbours considered by a single person.

### Examining Model Complexity

Figure 3.21 shows how the various sub-routines of the ORCA method change the simulation time. It shows that for complex considerations of a single agent, the largest time taken is with the linear program solver. For all three density types the linear program solver is where the code spends the longest time (the combination of lp2 + lp3) and this is exaggerated at higher densities. This is because 1) there are more neighbours at higher densities, so more complex problems, and 2) more neighbours/constraints increases the chances of not having a feasible solution which further increases the time taken in the linear program by running the lp3 routine. The additional constraint behaviours (groups, bags and autonomous vehicles) take roughly the same amount of time regardless of density due in part to the relative fluctuations being small compared to the total frame time. The results are obtained from a single run, averaged over the recorded run time of 50 seconds, which corresponds to 1500 data points.

The linear program routines take longer the higher the density, because there are more constraints to solve. In addition, with more constraints, the lp3 routine is more likely to run. The previous experiment showed that more constraints increase the time taken, regardless of how they originated. It can be assumed that either higher density or simulating additional behaviours adds computational burden.

## 3.5.3 Discussion

The experiments within this chapter have explored the effects that simulation parameters and behaviours have on the time taken. It has shown that each additional constraint considered per agent has a performance impact, but the origin of the constraint does not have an effect. This means that, when the number of constraints considered by a person is the same, it takes the same average amount of time to solve. This will be a useful observation for work later in this thesis, as improving

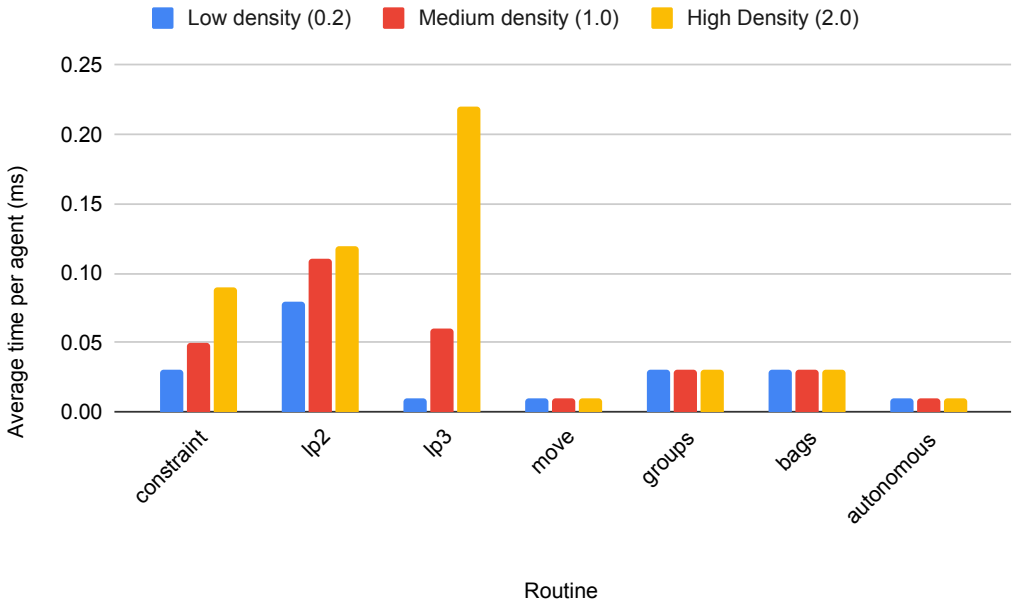



FIGURE 3.21: Timing of various sub-routines of ORCA (routine vs time taken). Multiple lines/bars show varying density

the performance of the constraint solver can be assumed to be effective regardless of the behaviours used in the model.

The most computationally slow routines are those which solve the linear programs. When there is no feasible solution, the routine to find the best candidate solution is itself a linear program. Thus, improvements to the 2D linear program solver should have the largest impact on performance improvements.

## 3.6 Summary

This chapter has presented a novel constraint based solving method, OCBM, to join steering avoidance with additional pedestrian behaviours. This addresses a limitation in the literature of a continuous-space model which can handle additional behaviours without detriment to the collision avoidance. It has shown that by using soft and hard constraints certain behaviours can be de-emphasised or encouraged while ensuring collisions are minimised. The chapter has broken down behaviours for group interactions, luggage, and autonomous vehicles into mathematical constraints that can be combined with existing steering methods (such as ORCA), which generate constraints from steering, into a single larger linear programming problem. Solving this generated problem provides motion that accounts for both steering and pedestrian behaviour together. The performance of the model has been explored and it has highlighted that the most efficient improvements in performance will be with the linear program solver.

The OCBM can be applied to more psychological behaviours than the three described within this chapter, which were groups, luggage, and autonomous vehicles. A potential future extension could be to incorporate 3D pedestrian navigation as velocity constraints. Various steering models exist for 3D space, including RVO and ORCA related ones [33]. Being able to account for the 3D shape of people, it will be possible to more accurately consider highly dense environments where people

will overlap others when viewed from above, but not collide within 3D space. Such an example is train carriages, where some people may duck under others who are leaning over to reach their luggage.

A future research direction would be to explore the effects of modelling autonomous vehicles as rectangular models. A rectangular shape more closely encompasses the shape of autonomous vehicles and cars. This could be explored by extending the ORCA steering approach to consider rectangular agents. This approach will be similar to other shape-altering extensions, such as elliptical agents [23], but applied to rectangles instead. The inclusion of non-holonomic behaviour, which occurs because autonomous vehicles cannot suddenly change direction, would be pertinent to include within this future study also [5].

The OCBM with separate luggage demonstrated different behaviour to simply simulating a singular larger agent. The next step in this research direction is to perform experiments on human behaviour which specifically target cross-flows. Understanding how real people behave in such a scenario will help provide guidance on the luggage behaviour model. This can help determine if one approach is more suitable for cross-flow scenarios.

There are performance limitations to the OCBM implementation. It is limited to simulating 400 people in real time. This limits the scope and size of scenarios the implementation can be applied to. It has been found through performance examination that constraints caused by different behaviours and steering have a similar impact on performance, and that the linear program solver is what the implementation spends the longest time computing.

In the OCBM model it is possible to construct a linear program which does not have a feasible solution. The choice of formulation of handling this is to find a velocity which least-penetrates the problem constraints. This minimises the chance of serious collisions with other people and breaking away from expected behaviour. Alternative approaches can be used, and these may be more relevant to certain scenarios of psychology of certain people. Future work could be to compare the emergent crowd behaviour of different types of solving invalid solutions, and comparing these to physical crowd experiments to understand the different formulations.

Having introduced and explained the framework implementation, the next chapter will consider the real world accuracy and suitability of the framework and behaviours with reference to real world data.

# Chapter 4

# Model Accuracy

The previous chapter introduced and explored a constraint-based framework, OCBM, for pedestrian simulation. The framework was able to be extended with new behaviours by describing them as constraints. The functionality of the different behaviours can be controlled through the hardness of the constraints. The flexibility of the approach was demonstrated by describing a range of group behaviours which were evaluated through computational experimentation. The performance of the approach was also evaluated through example experiments.

The applicability of OCBM for accurately simulating scenarios is unknown. An accurate model is able to reproduce real behaviour. Therefore, whether OCBM can replicate real crowds is uncertain. Knowing for what domains the model can replicate is important in understanding where the model can be suitably applied, and also perhaps provide insight into the underlying science of the crowd.

This chapter will investigate the model's behaviour with comparison to real world observations and experiments in order to undertake validation of the model and the OCBM approach. This will explore and explain how OCBM compares to experimental observations. By comparing the model to observed experiments it can be ascertained how well the model reflects the real world. Performing validation on the model will aid in understanding its usefulness and applicability. It will also provide an idea of the suitability of the simulation model for graphics and immersion.

Validation will be undertaken by using OCBM to simulate crowds and extract emergent crowd properties. These properties are an outcome of the interactions of the people in the microscopic simulation. These emergent properties will be compared to observations from literature and from obtained CCTV data of crowds. A model which is validated against observations is one which can simulate properties expected from experimental observations. Therefore if the simulation can similarly match emergent properties of observations, the model can be considered validated for the scenario covered by the experimental observation.

The process of validation involves targeting a physical scenario. This type of validation is a form of face validation that uses human intuition, as opposed to formal statistical analysis [118]. It will visually compare the results to draw validations and conclusions. This could be a type of environment like a corridor, or a train station. Observations of crowds at the chosen scenario are obtained, either from previous literature, or by recording and measuring crowds of people. The parameters of the

observations are used as model inputs for a simulation. These are quantities like environment shape, and number of people. Emergent crowd properties are extracted from the simulation, such as fundamental diagrams (see literature review, section 2.4.2 for more information), and these are compared to observation. Because the parameters being compared are emergent, they cannot be directly controlled in the model, they arise due to the complex interactions of people. If a Model can take information of individuals and match emergent observations at the crowd scale, it would suggest that the model captures important fundamental behaviour which is present in real crowds. This leads to the idea that the model is validated and is able to reproduce real behaviour because it captures the fundamentals of what is incorporated within such real behaviour. In addition to the visual validation approach used within this chapter, mathematical methods for comparing graphs is also possible. This could involve the Euclidean Relative Difference and the Euclidean Projection Coefficient to provide a mathematical measure of similarity [136, 55].

The data which will be used for validation is emergent properties. These are properties of the crowd, rather than of any one individual. An example is the fundamental diagram, a relationship of average crowd speed for a given density. Another property, one specific to train station platforms, is the emergent property boarding/alighting time: the time it takes for the people on the platform to board the train, and vice-versa. The data will include CCTV footage extracted at train station platforms. Quantities like platform shape, number of people, and the time it takes for people to board and alight trains are recorded. Also used are observations from literature [58, 183].

An important validated model is Legion [13]. Calibration and validation studies have been performed on Legion, such as by Berrou et al. [19]. This study explores a large range of data from various scenarios for validation. Berrou et al. explore the correctness of behaviour of people in locations like transport terminals, but does not explore Legion with regards to evacuation behaviour. It is an example of a broadly validated simulator software. The specific data used for validation is not shared, but it is understood that tailored experiments were carried out which validate properties of Legion such as pedestrian size, speed, and properties with luggage at a variety of different environment types.

This chapter will explore how valid OCBM is in two chosen situations. These are with luggage, and at a platform-train interface (PTI). Out of the behaviours examined previously (groups, luggage, autonomous vehicles), luggage is the simplest to isolate and study. The interactions of people in groups or with people and vehicles is vast, complex, and difficult to specify. Luggage, on the other hand, particularly trailing luggage, is more specific and the properties of people with luggage compared to people without is easier to distinguish. The PTI is chosen as it is a more complex and encompassing scenario, which draws on multiple different behaviours in order to properly simulate. This scenario can be used to demonstrate a model that uses multiple behaviours together. The proposed constraint-based framework is designed to handle multiple behaviours together without impacting the integrity of the steering, or the actions between the behaviours. Thus the PTI is an interesting case to affirm if the framework is functional for a multi-behaviour case.

The chapter will make the following novel contributions:

1. Validating the framework against experimental tests of trailing luggage.

2. Validating the framework against a multi-behaviour scenario at the PTI. The addition of multiple behaviours together will demonstrate the extensibility of

the model while maintaining the basic crowd functionality of collision avoidance

This chapter is structured into three main sections. The first, section 4.1, attempts to validate the OCBM with luggage behaviour against observations from literature. Section 4.2 is concerned about validating OCBM at the PTI. Final discussion remarks are provided in 4.3.

## 4.1 Experimental Validation - Luggage

In order to ensure behaviour is correct it is important to first consider validation. In this case, validation of the luggage model in OCBM. In particular, luggage refers to trailing baggage that is wheeled behind the owner. The implementation of luggage was explained in section 3.3.2, and its behaviour was explored in section 3.4.3.

People with trailing luggage can be an important factor in crowds, especially when they constitute a large proportion of the crowd, such as in transport terminals. Within the OCBM bag model, luggage is treated as a smaller agent which tries to steer around other people and bags within the scenario. People and luggage interact differently to each other according to a set of rules. These rules are expressed as a set of constraints which aim to maintain the expected position that luggage has when trailing behind its owner. An experimental evaluation will be conducted to test how well this modelling approach compares to observed data. This will address an area of future work highlighted in the previous chapter, in that it will compare the effects of modelling trailing luggage as a separate entity instead of accounting for a person with luggage as a singular larger agent. It is anticipated that this will provide justification for the approach of modelling baggage separately.

### 4.1.1 Literature

An initial step in validation is to find specific literature or suitable scenarios for testing with empirical data. For the case of luggage two scenarios are chosen: a bi-flow corridor, and a double-exit room. The bi-flow corridor is one of the simplest scenarios for the simulation that can provide meaningful comparison. In contrast, the double-exit room is a more dynamic scenario with some specific behaviours within, which help test the flexibility of the behaviour. A double-exit room is a large room where one side has two possible exits. People start in a waiting area within the middle of the room, then move towards one of the two exit locations. Figure 4.2 shows a specific double-exit, and visually explains how the simulation progresses.

Dong [58] et al. describes a specific bidirectional pedestrian flow experiment considering the influence of trailing luggage. Their work tests the effect of everyone with luggage, and only one side of the flow with luggage. The results conclude that the more luggage present in the experiment, the slower people will move at the same density. Their finding suggests that there is a converging point at large density of 0.5m/s that people will move at. Their work is useful to compare against because the experiment tests against a varying proportion of the crowd with luggage. Similarly [218] describes an experiment of people moving in a bidirectional pedestrian flow corridor under regular circumstances without luggage. This can be used as a baseline in validating the OCBM baggage model. They created a fundamental diagram for bidirectional pedestrian flow motion. This is for regular people without luggage or other behaviour. One conclusion they draw is that pedestrian flow with

separated lanes should not be interpreted as two unidirectional flows, as interactions between opposing streams are still relevant.

Recent work by [183] explores the double-exit room for varying levels of luggage. They find that unencumbered people crowd around the exits. However, with the addition of luggage, people form a more direct line to the exit, preferring to form a queue-like pattern. [218] have found a fundamental diagram for bidirectional pedestrian flow motion. This is for regular people without luggage or other behaviour. One conclusion they draw is that pedestrian flow with separated lanes should not be interpreted as two unidirectional flows, as interactions between opposing streams are still relevant.

Shi et al. [183] performed experimental observations of people in a double-exit room. They tested the effects of including luggage. They observe that people tend to crowd around an exit. However, people with luggage tended to queue and form a more direct line to the exit, even if that meant they were further away from the threshold. They also found that people in their experiment preferred to leave through the right-hand door if without luggage, but equally if with luggage.

### 4.1.2 Experimental layout and hypothesis

Three particular experiments will be performed, which seek to validate how people with luggage move within the OCBM baggage model. Validation will be undertaken by examining the speed in which people move depending on their density. These experiments are labelled: biflow 1, biflow 2, and double exit. The first two experiments, biflow 1 and 2, will alter the density the people are in, and the third, double exit, will alter who can choose which exit.

The experiments use varying agent types, these are the same that were defined in the previous chapter in section 3.4.3: (i) The OCBM version with luggage, labelled as "constraints", (ii) A singular person of radius $0.5m$, (iii) A singular person of radius $0.6m$, (iv) A singular person of radius $0.7m$.

#### Biflow 1

This experiment is designed as a single corridor, with people flowing in both directions. The corridor layout is the same as described in section 3.4.2 and a diagram of the set up is repeated here in figure 4.1. The experiment has 50% of the people carrying luggage all on one side. This setup mirrors the work of Dong [58] in order to compare the simulation model to their findings.

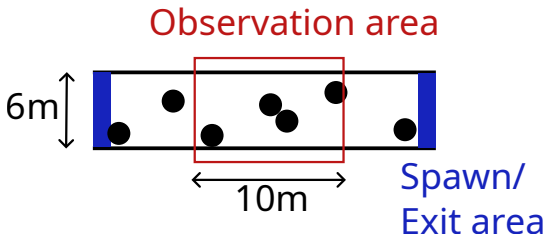



FIGURE 4.1: The biflow corridor setup. Same set up as the previous figure 3.9. The red rectangle, 6m by 10m, shows the area in which measurements are recorded. People are spawned within one of the two blue rectangles and navigate to the opposite blue rectangle.

This experiment will test which of varying agent types most closely matches the observations of Dong [58]. It will determine if the findings are more accurate when

simulating people and luggage as separate people, or as individual agents. This scenario mostly has head-on interactions and moving within the corridor in streams.

#### Biflow 2

This experiment uses the same environment as the previous experiment, biflow 1. This experiment has everyone carrying luggage moving in both directions. This is the same experiment and setup from the previous chapter in section 3.4.3, with the same results. The results presented for this experiment also include the observations of Dong [58] for comparison. This experiment together with the previous, biflow 1, provide insight into how the proportion of people with luggage changes the crowd behaviour.

#### Double-exit

A computational experiment is set up to mirror the experimental work of [183]. A graphical representation of the set up is shown in figure 4.2. People are spawned uniformly distributed within a 4.8m square. A wall is 6m away, with two symmetric exits at 2.4m from the origin. These exits are 1m wide. A measurement area of $2m^2$ is positioned in front of each exit. Within these areas the speed and density parameters of each person are extracted. The experiment is repeated for a fixed number of people and varying the proportion of luggage. The proportions of people with luggage is either 10%, 30%, or 50%. Each experiment enforced how many people exit through each door by assigning a door before running each simulation.

Shi et al. [183] note how many people move through each door and how many of those have luggage. These values are used within the simulation to ensure the same number and types of people exit through the appropriate door. The starting positions of these people are randomised within the waiting area for each simulation. This double-exit experiment is hypothesised to test the side-on interactions between people, as people will be steering through the cross-flow of the other door to get from the waiting area to their exit.

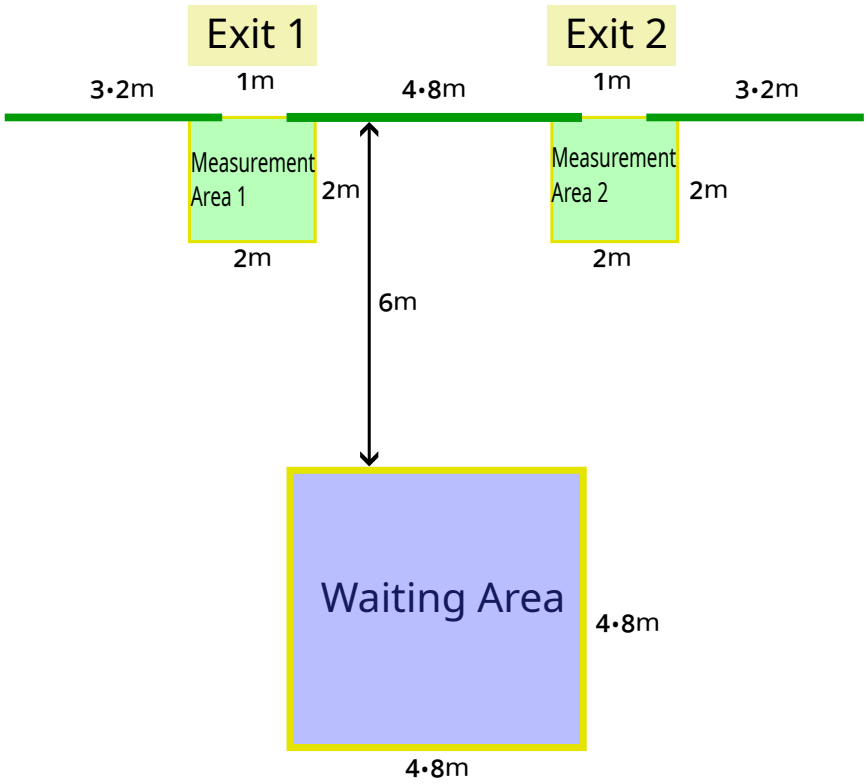


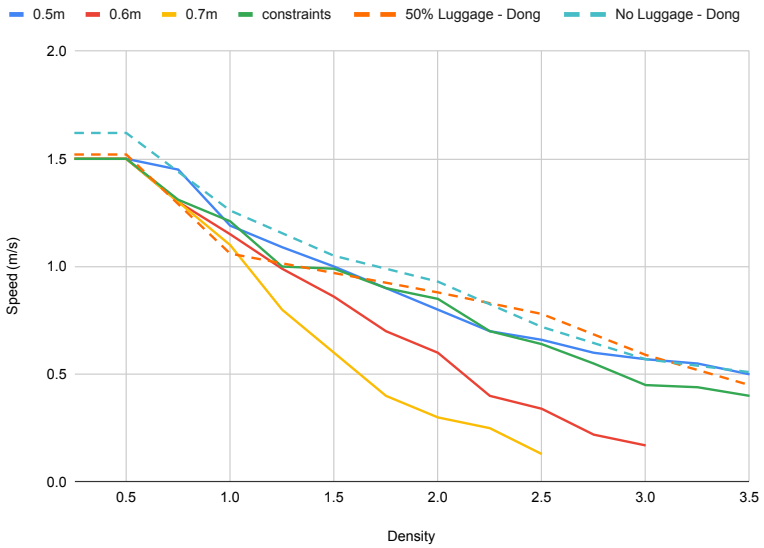

FIGURE 4.2: Set up of the double-exit experiment. People are initialised within the blue waiting area. Each person is assigned an exit door, and possibly assigned luggage. Once the simulation begins the people move towards their assigned exit. The speed people travel at is measured within the recording areas.

### 4.1.3 Experimental Results and Discussion

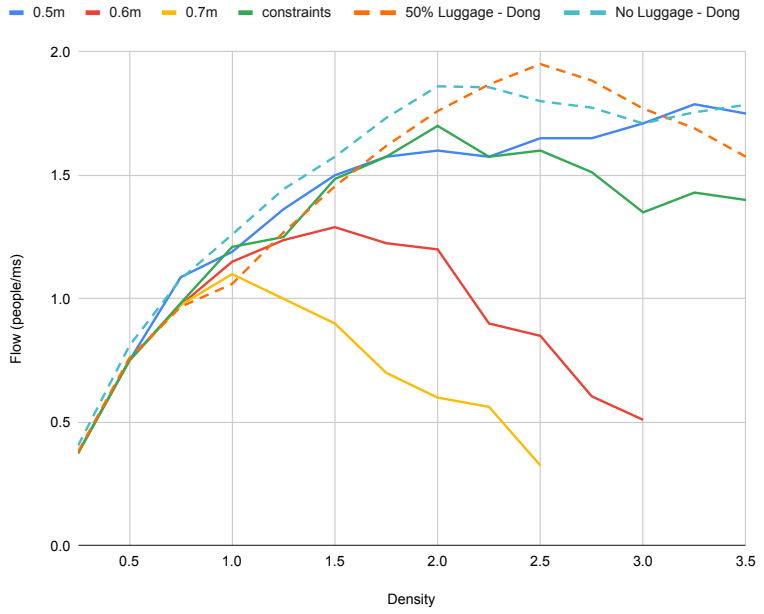
**Biflow 1**

The first experiment's results are shown in figure 4.3. This experiment has everyone on the left side with luggage, and people on the right without. The figure shows the work of Dong [58] as a baseline for the case of no luggage. The results from their paper are re-interpreted here (as in, values were read from their published results and converted into approximate numerical form, which was then plotted). The result from Dong [58] for the case of one side with luggage is also shown. For the cases of 0.6 and 0.7m radius people, half the people are simulated with the larger radius on one side, and half are simulated as the regular size of 0.5m on the opposite side. Each data point for the simulation results within the graph is calculated from the average of 2000 data points.

Experiment 1 shows that larger singular agents of 0.6m and 0.7m are not able to reflect experimental observation. The experimental findings of Dong [58] show the speed-density relation does not vary much between 0% and 50% of the population with luggage. A similar result is observed for OCBM and the agents of 0.5m size (i.e. no luggage). This shows a difference in the results of modelling a singular larger person compared to two separate agents. It suggests that the modelling of the separate luggage more closely follows experimental findings, especially at higher densities. The experimental results of Dong for 50% of people with luggage and 0% with luggage are similar. They are close enough to be within the bounds of




(A)




(B)

FIGURE 4.3: Fundamental diagram for various simulation models and experimental data in bi-directional paths for 50% of the population with luggage: (a) Shows the speed-density relationship, (b) shows the flow-density relationship

experimental fluctuations. The implication of this is that the effects of 50% of the population with luggage is less impactful than other more important factors.

The larger simulated people (0.6m and 0.7m) have similar speeds for low velocities, but the speed noticeably reduces compared to observational results at larger densities. These larger agents are not plotted along the whole density (x) axis because beyond the values shown, it is noticeable that issues such as crowding and standstill motion occurs. These effects skew the results, and they also reach the packing limit, where it is not physically possible to fit more people of that size at that density.

The velocity at low density varies between the simulations and experimental findings because of the assumed velocity of people in free flow. In the simulations it is set constant at 1.5m/s whereas Dong [58] finds 1.6m/s. This can be remedied by adjusting the simulation parameters to make people move at a higher freeflow speed. This parameterisation was not undertaken to maintain the same simulation parameters through the different experiments unless otherwise noted. Maintaining the same parameters for all experiments is desirable to ensure that only the difference in model/agent type is affecting the results. Experimental analysis suggests that people with luggage move at a similar speed to those without luggage in free motion, as shown by [98]. The value of free flow speed can be seen to be between 0.9 to 1.6 m/s depending on the context [60].

The findings of experiment 1 suggests that the cross-sectional length of a simulated person is an important consideration. This is because both the 0.5m and OCBM similarly match the results of Dong [58]. A factor that links the 0.5m and OCBM approaches is the horizontal cross-section. Ensuring a cross sectional length of 0.5m radius is less important at lower densities, where there is sufficient space along the width of the corridor. At higher densities the larger size of 0.6m and 0.7m people results in unrealistic behaviour caused by reducing the number of people that can fit side-to-side along the corridor width.

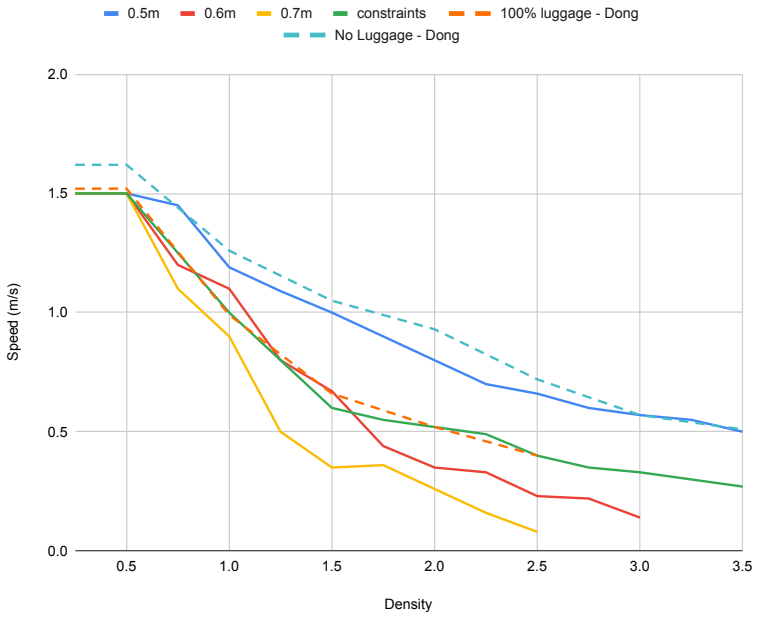
This can be reasoned because the bi-flow corridor mostly has head-on interactions, where agents are mostly concerned with the width of the people they are interacting with. This helps explain why 1) the amount of luggage does not significantly affect the speed-density, and 2) why the OCBM approach is able to more precisely follow the observational data.

#### Biflow 2

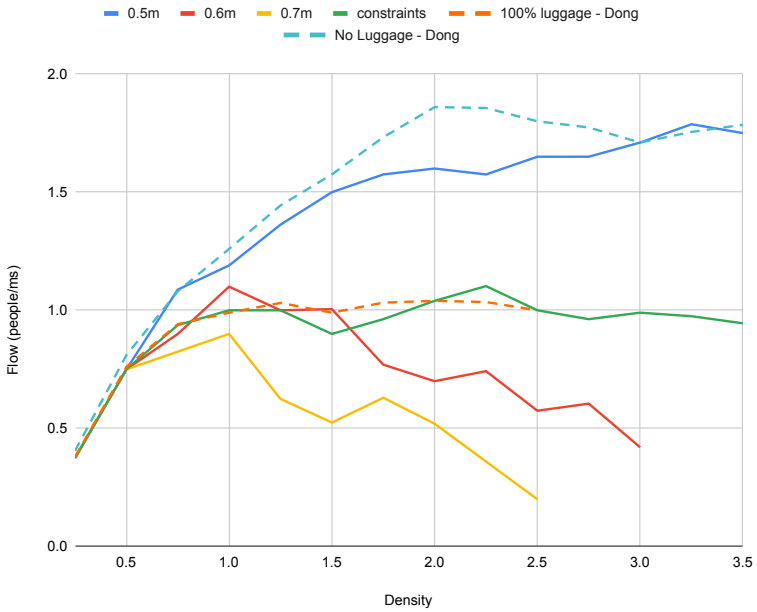
The next experiment tested everyone with luggage in a bi-flow corridor. The results are shown in figure 4.4. The graphs are overlaid with the work of Dong [58] and [218] to more readily see how the model compares to their experimental findings. The values for the simulation results are the same as obtained in the previous chapter, in figure 3.14. This figure extends the previous results with the work of Dong [58]. As with the previous experiment, the presented averages in the graph are calculated from 2000 data points.

Experiment 2 aimed to measure the impact of everyone with luggage in order to understand which modelling approach most closely matched the experimental findings. By having everyone with luggage, the effect of luggage compared to no luggage should be most pronounced. The results show that the constraint model, OCBM, is able to most precisely match the experimental findings.

The experimental results in figure 4.4 showed that flow curves 0.6m and 0.7m radii people appear similar to the observed values for small densities (less than 1 density unit). Above this density the speeds are lower than real-world observations.




(A)




(B)

FIGURE 4.4: Various simulation models and experimental data in bi-directional paths for 100% of the population with luggage: (a) Shows the speed-density relationship, (b) shows the flow-density relationship

This suggests that such singular larger agents can be a suitable simplification for low density, but not for higher density. The results for 0.5m are the same as the previous experiment (Biflow 1), as the proportion of luggage does not change the behaviour for 0.5m crowds. Similarly, 'No Luggage - Dong' is the same as the previous experiment's result of the same name, which is the experimental findings for no one with luggage. The data of Dong [58] is limited to a density up to 2.5 for 100% luggage use. Results are also limited by density for larger sized simulated people. This is due to reaching a maximum density within the simulated area. The constraint-based model is able to handle greater density by being able to more tightly pack people together compared to single larger people. Dong [58] suggest from their findings that people will tend to move at 0.5m/s when at high density. The simulated models do not find this, and rather finds the simulation tends towards a standstill as density increases. With regards to validation, this suggests that at very high density the OCBM is unsuitable for representing real crowds with luggage. This difference in behaviour at high density suggests different underlying mechanics which are not captured by the OCBM. Further work should be carried out to understand how this can be simulated.

#### Double-Exit

The results of the double-exit experiment are shown in figure 4.5. It shows the comparison between the experimental observations of Shi et al. [183] and the simulation results. The results are labelled in the same method used in the paper of Shi et al. [183]. Different cases are labelled with 1X2Y-NN, where 1 is the left door and 2 is the right door and the X/Y following a number explains the rule for that door. The possibilities are B (both with and without luggage), W (without luggage), L (with luggage), e.g. 1B2L means the left hand door can have either type of luggage or without, but the right hand exit can only have people with luggage pass through. The final part of the index (NN) is the proportion of people with luggage in the whole setup. Each experimental configuration was repeatedly run 500 times, and the average of the results presented.

The overall average for each approach is shown in figure 4.6. It shows the average difference of each approach against the observations of Shi et al. [183]. The smaller the error, the closer the approach matches observation. The constraint approach, OCBM, is closest to the experimental observations overall. The constraint approach has an average error of 0.028, which means that on average, OCBM is different to Shi et al. by approximately $0.028 m/s$. Both the approaches of 0.6 and 0.7 are the worst matching, at around $0.1 m/s$.

The double-exit experiment, compared to the previous bi-flow experiment, involves more side-on interactions with neighbours. It is anticipated that if only the size of the side cross-section (i.e. size of the front to back of the person/person+bag) that the constraint model and the 0.7m people should have a similar result. The results of the double-exit experiment show considerable difference between the constraint and 0.7m model. This suggests more interaction angles are present and important within this experiment.

The 0.5m radius people do not change whether they have luggage or not. They are always the same size and have the same behaviour in the model. Within the graph the speed of 0.5m radius people varies between cases because different numbers of people exit the doors, resulting in more or less crowding, which affects the speed. The results of 0.5m reflects how the model would behave if luggage was not accounted for. The speed of 0.6m and 0.7m singular agents is quite affected by

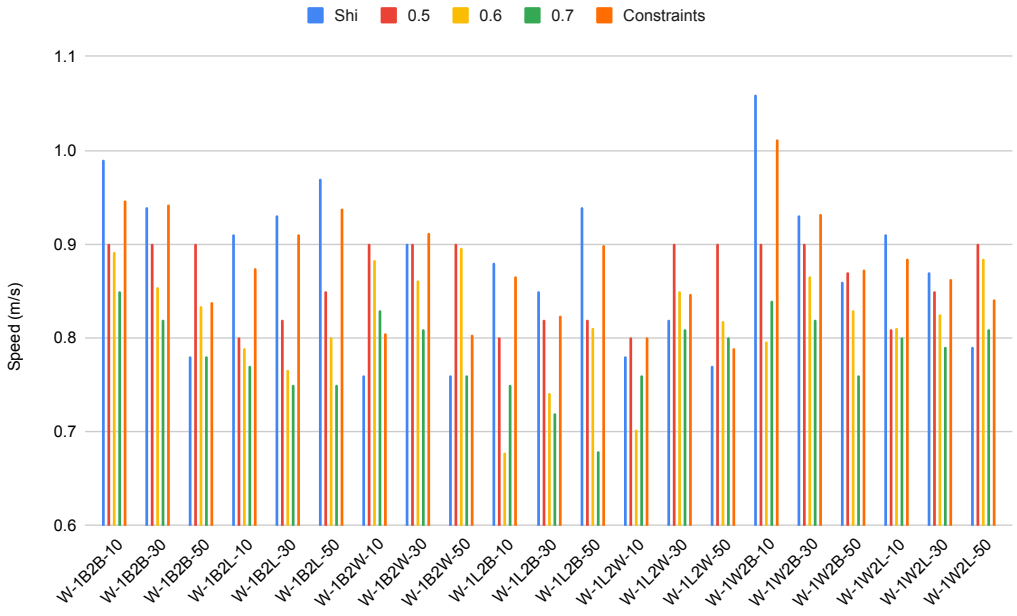



FIGURE 4.5: Double-exit experiment results. It shows a comparison of speed between observations by [183] and simulation results.

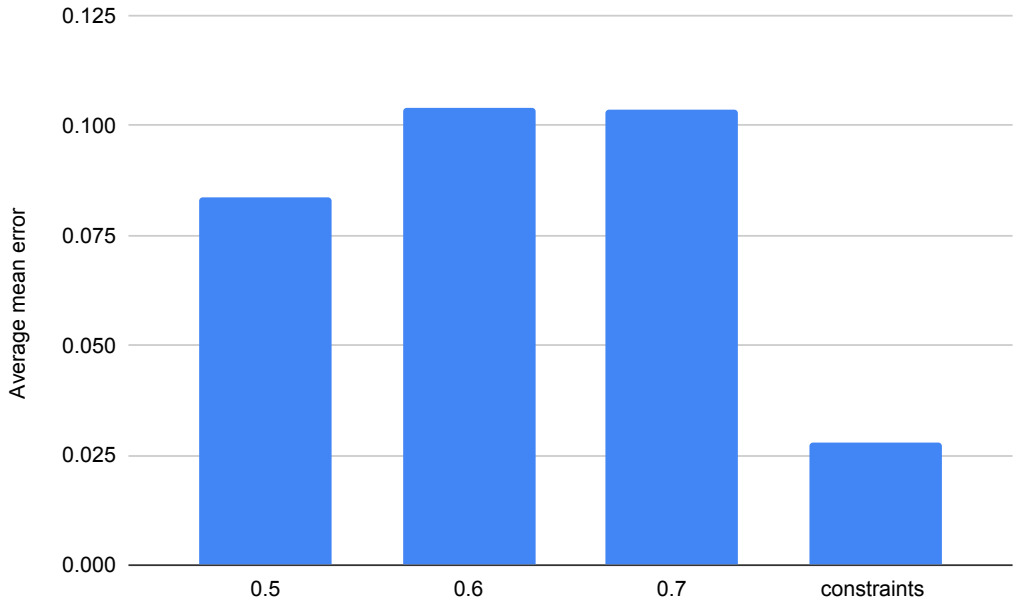



FIGURE 4.6: Mean error for each approach over the whole range of scenarios for the double-exit experiment results. A lower value means the method is, on average, closer to the experimental findings

the number of people per door. This can result in large discrepancies where the observed speed is relatively high even though the number of people exiting per door is imbalanced. An example occurrence of this is W-1L2B-50, where a balance of 42-18 occurs with 50% of the population with luggage. The larger singular agents tend to match more poorly than the regular 0.5m radius people.

OCBM matches best to the observations when the observational speed is large, such as W-1W2B-10. This can be reasoned because of the asymmetric shape of the agent+bag combination which encourages neighbours to move a certain way that is more conducive to moving quicker through the exits. OCBM does not always match best for all the considered cases, but does tend to perform better on average compared to the alternative models, as demonstrated by the smallest average error of the different approaches of $0.03m/s$.

### 4.1.4 Summary

Two experiments were performed to compare simulated models against the observations findings of Dong [58]. The experiments tested scenarios with 50% and 100% of the people with luggage. The results indicate that OCBM was able to replicate the experimental findings of the fundamental diagram under visual inspection of the results. However, the data collected only goes up to a density of 2.5 and cannot simulate large groups because they reach a maximum density within the simulated area. Overall, OCBM seems to behave closer to the experimental findings of the fundamental diagram compared to other methods, especially at high densities.

The double exit experiment compares the experimental data from Shi et al.'s work on pedestrians with luggage to simulations conducted using various simulations. Results indicate that there were significant differences between OCBM and simulating people of size 0.7m, suggesting that factors such as the size of the side cross section may play a role in determining behaviour in this scenario. Furthermore, it was found that the speed of 0.5 metre radius agents (i.e. no bags) varied depending on the case due to varying numbers of people passing through each door, while the speed of 0.6 and 0.7 metre agents tended to be influenced by an uneven distribution of individuals at each door. Finally, OCBM matched well with the observation when the speeds were higher, particularly in scenarios involving equal distributions of people.

The experiments have indicated that, for the scenarios tested, it is better to simulate people and luggage separately using OCBM, as it is able to more accurately capture observed behaviour. OCBM is able to simulate higher densities of people because less space is taken up per person, compared to singular larger agents. However, at high densities, OCBM tends towards a standstill, whereas observation suggests speed tends towards a minimum velocity.

## 4.2 Platform-Train Interface (PTI) Validation

The platform-train interface is an area which demonstrates the importance of specific pedestrian behaviours. People at the PTI follow rules not found in general crowds. Such an example is waiting for people to alight the train before boarding. Without accounting for these behaviours, it is very difficult to make believable or accurate pedestrian motion. People at the PTI also have a higher-than-average amount of luggage, especially on routes linked to airports. These challenges make the PTI scenario useful in testing OCBM. Modelling using alternative approaches of baggage

representation was not considered as this has been explored in the previous section. Validation will aim to assess the flexibility and accuracy of the OCBM modelling approach to meet the chapter contribution of demonstrating the flexibility in modelling multiple pedestrian behaviours simultaneously.

Validation is carried out by comparing the emergent crowd property flow time. This is the period of time when people board and alight the train. Observational data is obtained from industry partners with access to CCTV footage on train platforms[1]. Flow time is extracted from this footage. Simulation flow time is recorded as a model output. By adjusting simulation inputs according to observed data (such as number of people boarding and alighting), the simulation flow time can be compared to observed flow time to see how well the model can match real data.

The following subsections will give more detail. Section 4.2.2 will explain how CCTV data was captured and how the visual images were transformed into observational data. Section 4.2.3 will explain changes to the model and framework in order to account for PTI specific behaviours. Finally, section 4.2.4 will provide results and discussion.

The examination of the PTI aspect was part funded by various industry partners and research grants[2], and has produced a published paper [85], some of which is presented within this section. The CCTV analysis in this section was performed with the assistance of Dr Sam Hayes.

### 4.2.1 About PTI behaviour

The PTI is the region of interest when considering the boarding and alighting of passengers at a train platform. It encompasses the train carriages and the station platform. The PTI transfer period is particularly important because the movement of people can be a safety issue. It is also a key part of understanding how long a train needs to dwell at a platform, which will impact timetabling and potential congestion along the whole line if not correctly accounted for.

The dwell time is the total time the train is stationary at the platform. Within this work dwell time is used interchangeably with the time between doors open to doors closed. This is because the other parts of the dwell time are nearly constant and not affected by passengers. The factors affecting time to doors opening are those related to the dispatch process, which is usually independent of the people involved, nor affected by the time the boarding/alighting process took. Another name used for the boarding/alighting process is the flow time. This is the time taken from the first passenger boarding/alighting until everyone has completed the process.

Figure 4.7 shows what happens after a train stops at a platform and how the motion of people changes during this time. The figure is only a general approximation to elucidate the movement, as specific flow rate is highly variable on numerous factors. The simulation within this section attempts to model the period between doors opening and closing, for people to board and alight the train. This focuses the data collection to a clear period and makes validation more specific.

Two particular pedestrian behaviours are observed to exist at the PTI. At stations in the UK, it is encouraged to allow people to first alight, and then board. Each door

[1]Data collected with ethics approvals from the University of Sheffield under the applications: "Return to Work: Passenger flow modelling at Peckham Rye station", Reference Number 045750; "RateSetter: Improving passenger boarding rate and reducing risk at the Platform-Train Interface", Reference Number 013389; and "RateSetter+ Social Distancing", Reference Number 036472

[2]Merseyrail, RSSB, Network Rail, EPSRC grant https://gtr.ukri.org/projects?ref=ES%2FW000601%2F1, and the Higher Education Innovation Fund (HEIF) program

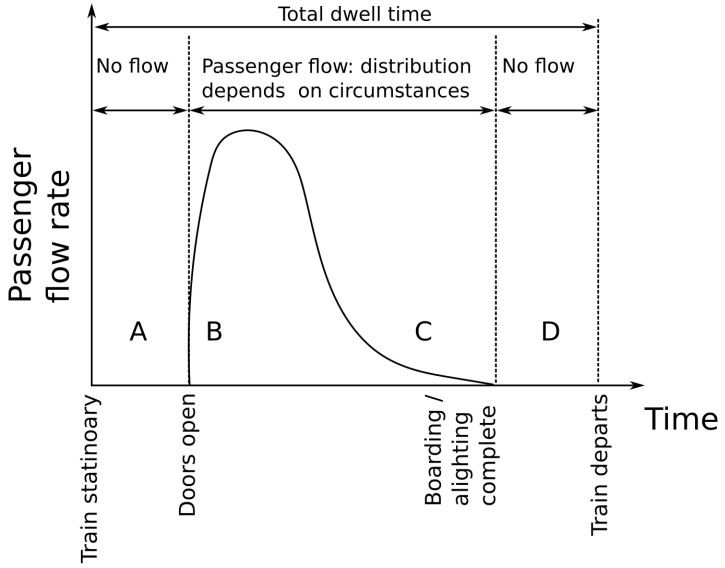



FIGURE 4.7: Passenger flow over time at the PTI for an example service [85]. It covers the process from when the train arrives at a platform to when it departs. People only flow while the doors are open.

alights then boards independently and may take longer than neighbouring doors. Once all doors have completed boarding/alighting the whole train has completed the passenger flow, and the train can begin to depart. A psychological behaviour observed in relation to the PTI is that people will bunch up towards the door they are going to enter. They form a dense crowd centred on each door of the train.

### 4.2.2 PTI Data Collection

Data collection took 2 different forms: CCTV, and survey data. The form of data collection depended on the station examined due to different types of data being available for access.

There are numerous options for recording data from video and images, such as CCTV video. There has been a large advancement in computer vision, which leads to accurate tracking of pedestrian movement [35]. Unfortunately these rely on certain requirements. One issue for these vision based tracking approaches is handling obscuration. Station cameras do not look vertically downwards, so some people towards the background of the camera are obscured for large periods making them very hard to track. This meant that vision based tracking techniques could not be used for decomposing video into useful numerical statistics. Instead, processing of numerical data relied on manual measuring. This limited the amount of footage which could be analysed and was limited by manpower hours.

CCTV data collection was conducted using pre-existing station CCTV systems and received clearance under the remit for their use in ensuring safety and well-being of railway customers and employees. No identifying or personal information relating to passengers or railway staff was taken into the analysis. Three locations were used for this: Birmingham New Street, Leeds, and London Bridge stations. Data from Birmingham New Street was collected on the 12th of October 2020, Leeds footage was obtained on the 26th of September 2020, 12th of March and 26th of May 2021, and footage from London Bridge was obtained on the 4th of May 2021.

CCTV imagery varied between different stations. This variation changed image resolution, overall quality, and vantage position. The cameras are those already in place by the station owners, or British Transport Police. The cameras in place are used for obtaining a wide view of platforms, for the purpose of crime identification. These cameras were not designed for providing a clear view of people's movement during the PTI process. Another issue of the CCTV available is that only the platform's cameras were able to be accessed. It was not possible to record the inside of trains to see the PTI process within the train carriage.

The CCTV footage was analysed by visual inspection, recording the number of passengers boarding and alighting at the PTI and the flow time which started when the first passenger alighted from the train and ended when the last passenger had boarded. These could be recorded per-door, for a limited number of doors which had a sufficiently clear view of the boarding/alighting process. In some cases, only a subset of doors were recorded.

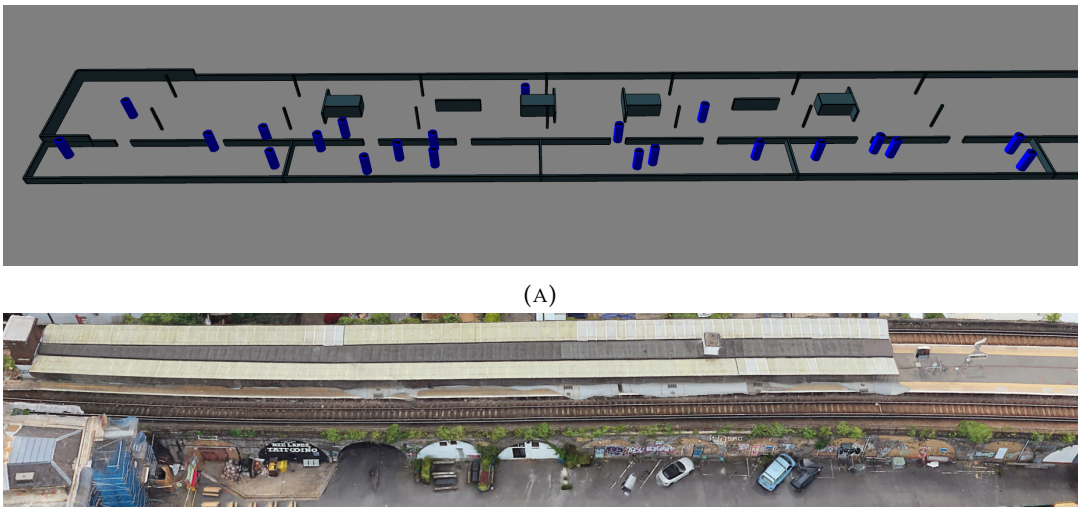
The train stock type was also recorded and grouped into either suburban or intercity stock types based on the location of the train doors. For trains with long dwell times that allow for passenger boarding over a prolonged period only the initial passenger transfer was considered where interaction between boarding and alighting passengers was observed. To avoid artificially inflating flow times individual passengers arriving late after the main flow period were discounted, as arbitrary late arriving passengers were outside the scope of the modelling.

### 4.2.3 PTI model

An example simulation environment is shown in figure 4.8. It shows the environment used for Peckham Rye, showing a train of (N) carriages and the platform. The simulation environment is set up with the walls of each train carriage modelled. Doors are assumed open and their movement is not modelled. This is because the simulation is only testing the time between doors opening and closing and as such they can be considered as an open doorway during this period. The platform edges are modelled as walls to prevent people from stepping off. If the platform has notable features like pillars or benches (such as within figure 4.8b), these are added in with a rectangular approximation. Each station platform experimentally simulated has a unique layout which is Incorporated into the model.

The simulation begins with the train doors fully open. Once all people have reached their destination in-simulation, it is equivalent to the boarding/alighting process finishing. A number of people are seeded into the simulation, they either begin on the platform and aim to board the train at their nearest door, or begin within the train and aim to exit the platform by alighting. This simulation does not model anyone who is waiting and not taking part in the PTI process. The number of people to initialise are determined by the observation case of interest.

Further behaviour is added to the proposed framework to be able to account for PTI specific pedestrian motion. The first of these behaviours is the waiting for alighters to leave before boarding. This is a macroscopic crowd-wide behaviour, rather than an individual one, and for this reason it is not implemented using constraints. The second behaviour is the introduction of line of sight. This is a constraint behaviour in which people are assumed to not have knowledge of people behind them. Every person is assumed to be facing the nearest door. It is the responsibility of people behind to avoid colliding with those in front. This is implemented within constraints by altering agent-agent ORCA constraints using variable reciprocity. The line of sight is assumed to be a total angle of 135 deg directed at the



(A)

(B)

FIGURE 4.8: Virtual PTI environment of Peckham Rye. (A) the simulated environment. Some blue people are in the environment. The bottom area is the train carriages. (B) a Google Earth aerial view of the same location. The obstacles in the environment correspond to the objects under the overhanging area.

nearest door. The change arises if one person can see someone who does not see the initial person. In this case, the person who can see the other takes full responsibility for avoiding them. The justification for this behaviour is from observation that people boarding trains tend to cluster around doors, with everyone facing and focusing on their targeted door, with little attention paid to those behind them. The value of 135 deg is postulated by examining visually how people interact around doors, both from CCTV data and from the simulation, choosing the value which appears most believable.

Previous observation on mainline UK rail has showed that in most cases passengers wishing to board will wait on the platform in an orderly manner until most passengers wishing to alight have done so, even if the doors are wide enough to facilitate bi-directional flow [65]. Realistic representation of this transfer between boarding and alighting passengers is therefore crucial in the prediction of total flow time. To represent the behaviour, an area of the train vestibule was defined that boarding agents can 'see' from outside the train, taking account of door size, shown in green in Figure 4.9. If this area of the vestibule area is clear a passenger waiting to board will begin to board the train, even if there are other people 'hidden' inside the train wishing to alight. However, if the vestibule is not clear of alighting passengers boarding passengers will continue to wait on the platform. This better represents real passenger behaviour than if boarders must wait until all alighting passengers have left the train. Additional influences on the passenger flow include variation of unhindered walking speeds, individual passenger mobility, behaviour of passengers travelling as a group, and the effect on movement of carrying luggage.

The effect of sight is important in modelling PTI crowds. People face the door of interest and pay little attention to anyone behind them. This behaviour translates to simulation by creating a sight-angle for people. If a neighbour is outside this arc, the person will not notice them. In this case the unseen neighbour must take full responsibility to avoid the person, and equivalently the person will make no attempt to avoid the unseen neighbour.

The primary metric of interest from the simulation output is the total flow time for all passengers in the simulation to complete alighting and boarding. The flow time is calculated from the time the train doors open and passengers begin to move and ends when all boarding passengers have moved 2m into the train and alighting passengers have reached a goal location at least two-thirds along the depth of the platform (i.e. 2 or 4 m for the 3 and 6 m platform widths respectively). The locations of each agent during the simulations can be visualised to observe agent trajectories and speeds and to enable time-proximity metrics to be calculated.

Figure 4.9 shows the graphical simulation of the PTI. Blue cylinders along the top of the screen represent boarding people on the platform. Orange cylinders represent alighters within the train. The images are snapshots throughout a particular simulation. It shows a very simple layout where the 3 carriages of the train are at the bottom of the images and contain alighters (orange). The vestibule is overlaid in green within the carriages. The platform is a simple scenario without obstacles and has a depth of 6m.

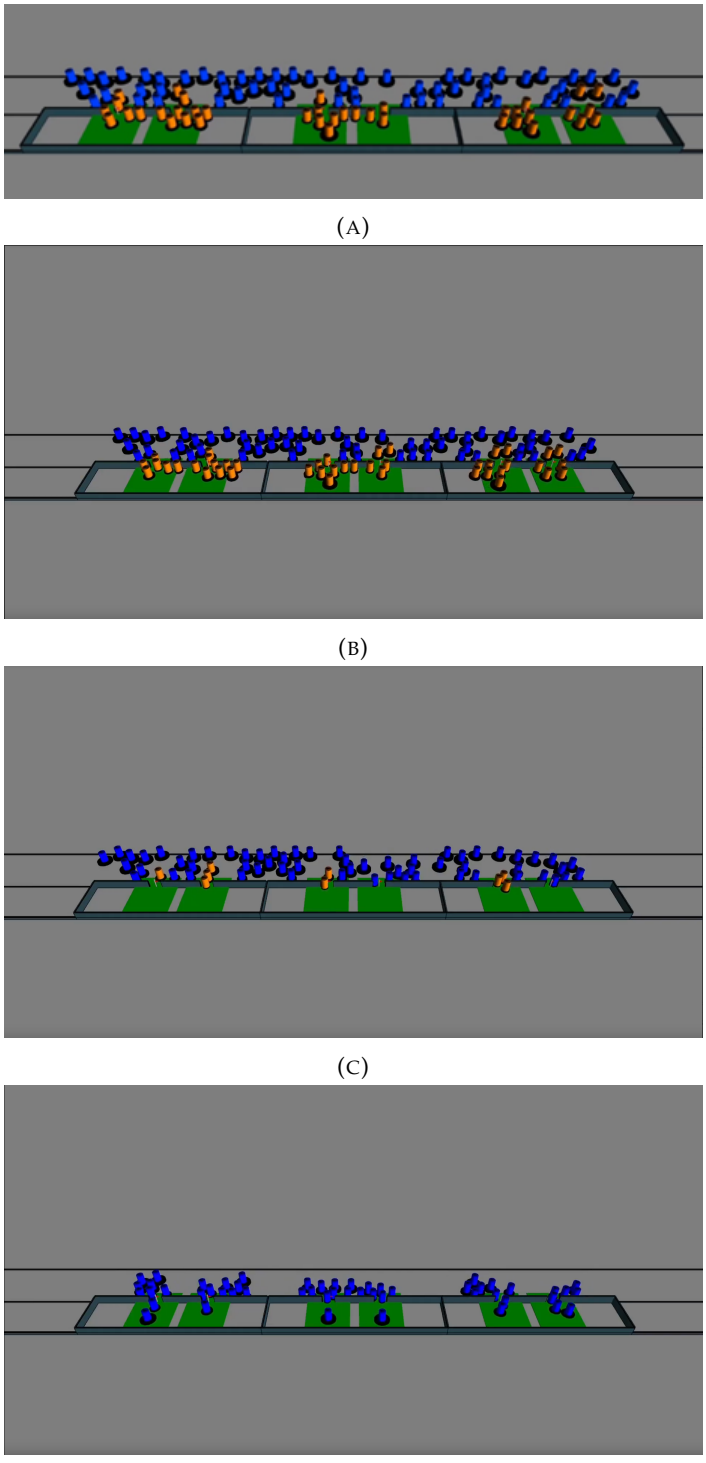
It is usual for train doors to be at a different height to the platform. In addition, there is also a potential gap between the train and the platform. People will take longer to travel along the threshold of the door because of this. The time taken for people to traverse this area is extrapolated from the work of Holloway et al. [94]. Holloway experimentally records the time taken to enter and exit a train depending on step height and gap. The work finds different values for people alighting compared to boarding. This occurs because people can walk down (alighting) quicker than they can ascend (boarding). These values are used within the simulation and linearly extrapolated for cases that do not line up exactly. The result is a "slowdown" region around the train doors which results in people moving more slowly, at a rate dependent on whether they are boarding or alighting.

The observations took place during the COVID-19 pandemic. During this time people were attempting to be socially distant. Observations during this time showed that people would be distancing while waiting for trains. However, when boarding, people would crowd around doors much like there was no social distancing in place. A smaller number of people would remain distanced, and stay further away from the door.

This was modelled by creating the interaction radius of people to be larger than their usual personal space. In cases of complete social distancing each person had a radius of 1m, which allowed for 2m social distancing, which was the country's recommendation at the time. To account for a reduction in social distancing for people crowding around doors, a linear scaling was put in place that adjusted a person's effective radius from maximum distancing while far from doors to no social distancing while within the PTI region. This radius was used for person-person interactions to ensure people distanced from each other. It was not used for person-object interactions so people could still move close to obstacles as normal. The PTI region was set as the first 0.5m of the platform, train-side.

This social distancing is highlighted in figure 4.10. This figure shows a birds-eye view of the PTI, with the trains on the bottom edge. As people move towards the train, their social distancing amount decreases, shown by the smaller yellow circles encircling each person. The effect of social distancing can be seen in figure 4.11, which is a zoomed in section of figure 4.9. It shows a low black skirt surrounding the cylinder people, noting that the skirts are larger for people at the far edge of the platform (top of the image) compared to within the train (bottom).

The model was implemented in the SteerSuite framework by extending the model



(A)

(B)

(C)

(D)

FIGURE 4.9: Graphical simulation of the PTI - Aerial view. a) Snapshot taken around 1 second after doors open. Alighters (orange) are moving onto the platform. Boarders (blue) are waiting until no more people are visibly exiting their door (seen by the green vestibule). b) is taken 5 seconds after the doors opened. Some of the alighters have left the train and exited the simulation by reaching the far side of the platform. c) shows some of the people boarding doors where there are no alighters left in the vestibule, taken 12 seconds after the start. d) shows the PTI process about to finish, 20 seconds after the simulation started. It is waiting for people to board only.

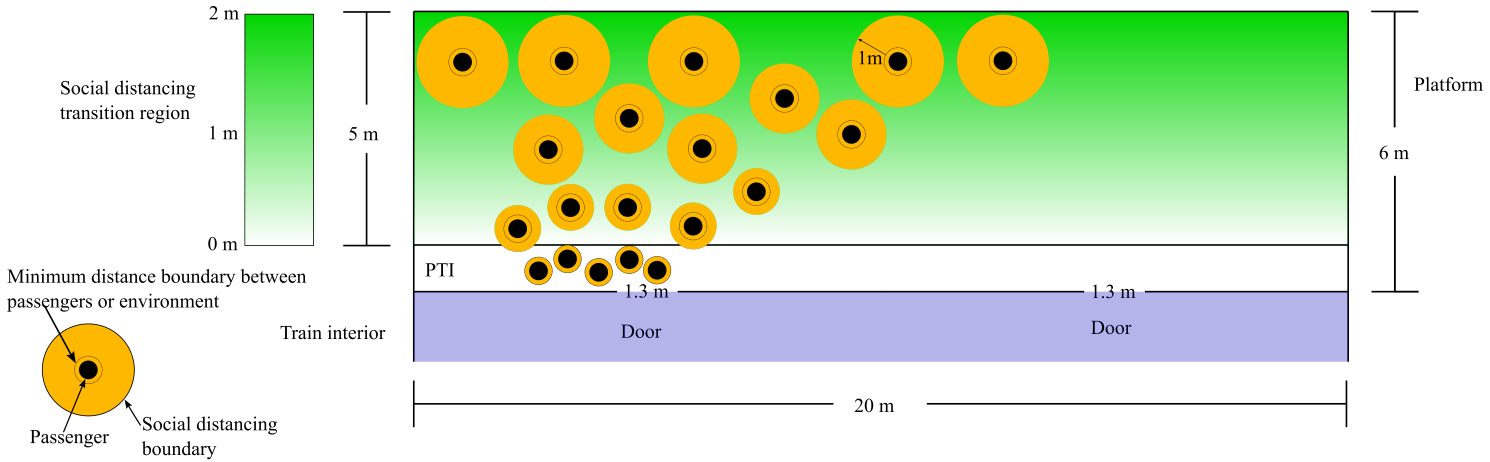



FIGURE 4.10: (Not to scale) Plan view schematic of passenger behaviour on the platform demonstrating the decreasing distance between agents as they move toward a suburban train door. The black circles represent a person's physical extents, while the yellow circle represents the person's social distance, which reduces as they approach the train doors.

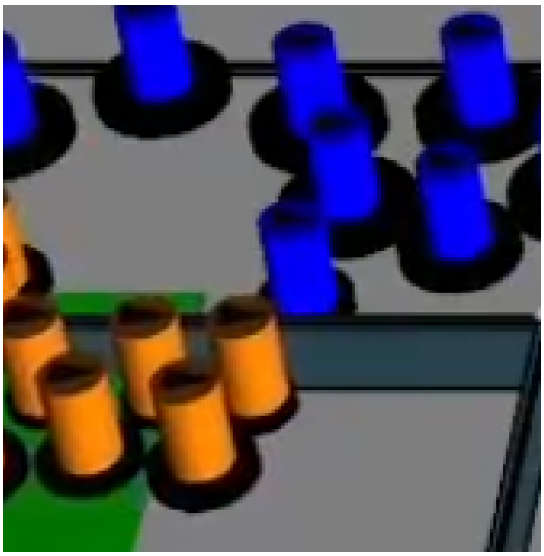


FIGURE 4.11: Graphical view of people socially distancing in the simulation. The black skirt represents their personal social-distancing space. This decreases closer to the train doors. This is a zoomed in look from the simulation in figure 4.9.

presented in the previous chapter. Luggage is included in the model. Whether people had luggage with them was recorded in the experimental observations. Groups are not included because group behaviour is not an important factor in boarding and alighting. Packing closer to doors and the flow through the doorways are more important than groups and group formation during the PTI process.

### 4.2.4 Results

Three platforms were experimentally evaluated through simulation and compared to the observational data: one in Leeds, Birmingham New Street, and London Bridge Station. For each of these the observed flow time was compared to the simulated flow time to compare how well in agreement the simulation is to real observations. For each platform and station with observed data, the data is used to parameterise the simulation as inputs and outputs. Inputs are factors like number of people boarding, alighting, platform layout, number of people with luggage. The output factor is the flow time. For each of the platforms, repeated simulations were executed until the standard deviation was less than 5% or 500 runs, whichever was sooner. This provides the simulation's predicted flow time for the given set of inputs. These were averaged over repeated runs and compared to the observed value.

Table 4.1 shows the data from CCTV observation of passenger volumes and flow times for Leeds Station, Cases 1 to 8 within figure 4.12. It shows model inputs (stock type, boarding and alighting passenger numbers) and outputs for comparison (flow time).

The comparison between the predicted and measured passenger flow times for each of the cases is shown in Figure 4.12. Good agreement was observed between the predicted and measured flow times with the largest difference occurring for Cases 5 and 7. For these cases, the larger flow time observed in the CCTV compared with the modelling output was due to passengers alighting with bicycles, a phenomenon not currently modelled using the OCBM approach. Passengers were also alighting

| Case | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Time of day | 07:40 | 08:00 | 08:25 | 12:30 | 12:50 | 12:57 | 13:04 | 13:05 |
| Stock type (S–Suburban, I–Intercity) | S | I | S | S | S | I | I | I |
| Boarding passengers per door (max., average, min.) | 3, 2, 1 | 2, 1, 1 | 3, 2, 1 | 15, 10, 7 | 7, 5, 2 | 4, 1, 1 | 6, 5, 4 | 13, 8, 4 |
| Alighting passengers per door (max., average, min.) | 12, 7, 2 | 12, 5, 2 | 11, 8, 2 | 17, 10, 5 | 6, 6, 5 | 9, 5, 3 | 6, 5, 4 | 5, 3, 1 |
| Dwell time (s) | 337 | 349 | 280 | 110 | 449 | 643 | 383 | 304 |
| Flow time (s) | 24 | 22 | 21 | 37 | 37 | 20 | 43 | 44 |

TABLE 4.1: Data From CCTV Observation of Passenger Volumes and Flow Times for Leeds Station, Cases 1 to 8. Each of the Train Dwells Observed in the CCTV footage is labelled With a Case Number, 1 to 8, for reference in the main text.

from the adjacent door so boarding passengers could not bypass the blockage by choosing to board through an alternate route, and thus queued at the original door.

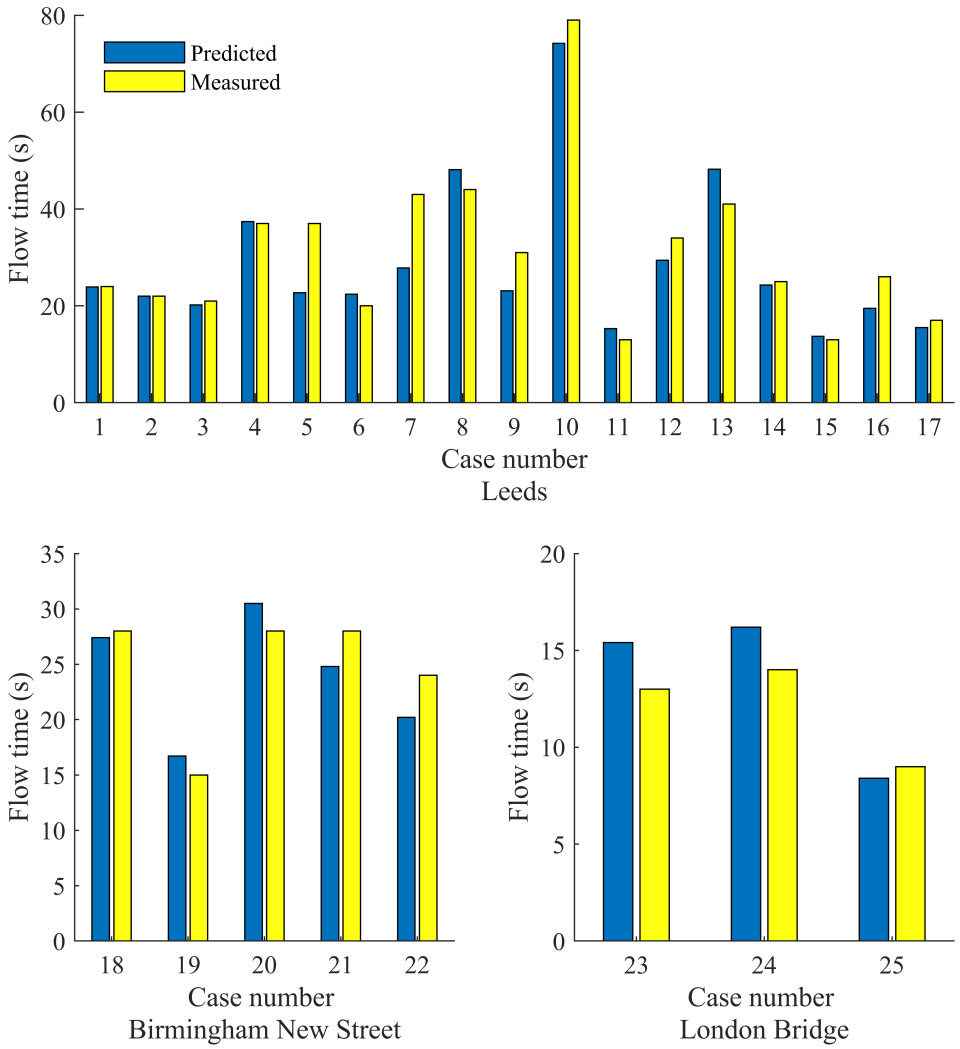



FIGURE 4.12: Measured (yellow) and predicted (blue) flow times for different trains arriving at Leeds, Birmingham New Street, and London Bridge. Each case corresponds to a particular train at a platform

The model predictions for the flow time (figure 4.12) are observed to be close to the measured values, and the root mean square average gives a variation between the measured and predicted flow times of approximately 1.5 s. Considering only the absolute differences in the flow times, this variation is approximately 3.6s, and discounting the outliers discussed above, this variation reduces to 2.4 s. When large numbers of passengers are boarding, such as Case 10, where on average 32 passengers boarded per door, the model predictions maintain good agreement with the real flow times. The model prediction in this case was approximately 5 s less than the real time, representing a 6% variation between the two flow times. Similar variations are present in the other flow times.

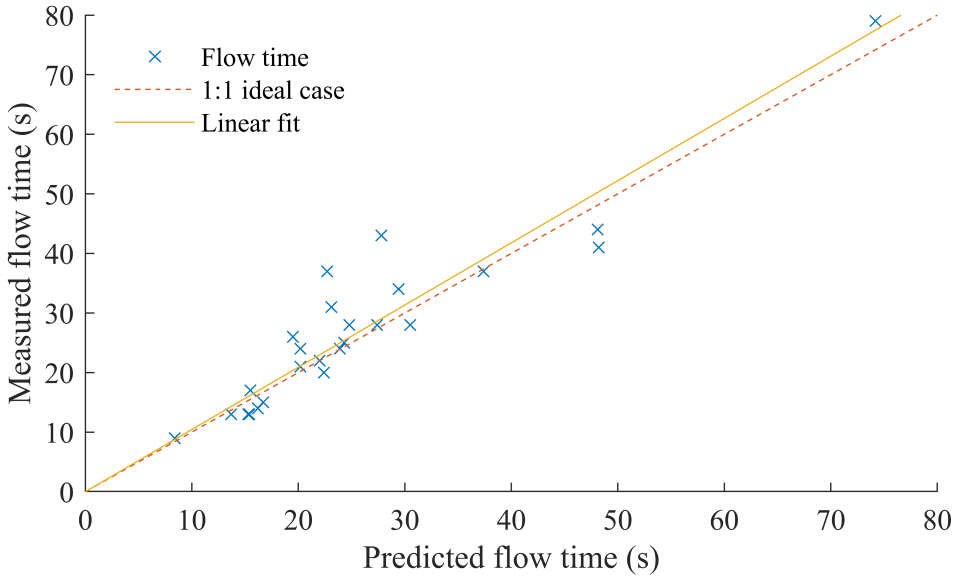



FIGURE 4.13: Predicted vs simulated flow time of the above cases (Figure 4.12). Values that lie on the 1:1 line (dotted) have an exact reproduction of the experimental finding in simulation. The linear fit line (solid) shows the best fit of the data. The closer this line is to the 1:1 line, the more closely simulation matches with observation.

Figure 4.13 shows the correlation of predicted flow times with measured flow times. The dashed line indicates an ideal 1:1 correlation where the model would exactly predict the measured flow times, and the solid line indicates the linear fit to the data. The linear fit was assumed to be of the form $y = ax$ for some constant $a$. The gradient, $a$, of the linear fit is approximately 1.04 compared to the gradient of 1 in the ideal case with a coefficient of determination, $R^2$, value of 0.97 suggesting the linear fitting was in good agreement with the data. Discounting the outliers, the gradient of the line would be approximately 1.03. Since the gradient of the fitted line is greater than the 1:1 case, it suggests that the model in general slightly under predicts the passenger flow time. This is likely due to not having a complete granular representation of the human behaviours at every time step during the boarding and alighting process (e.g., passenger movement speed varies, change in individual directions upon boarding the train). Across all cases considered in the validation of the model, the mean predicted flow time was 26 s and the mean measured flow time was 27 s, representing only a 1 s variation between the model and real flow times.

The standard deviation of the flow times was 13.6 s and 14.4 s for the predicted and measured times, respectively. The absolute standard deviation provides an idea of the different crowds of the train services. A small deviation would suggest that the cases considered had a similar number of people. The similarity between predicted and measured standard deviation suggests that the model is able to handle a broad range of population sizes at the PTI.

These experiments demonstrate that the OCBM constraint-based framework is able to predict observed flow time to an accuracy of > 90% for a PTI scenario varied over a range of stations and platform conditions. For the cases tested, it was able to accurately reflect the flow time for the PTI.

## 4.3 Summary

This chapter has examined the accuracy of OCBM, the constraint-based framework. It has done this through comparing behaviours to literature for luggage-laden people. It has also done so by collecting experimental data of PTI processes at various stations and comparing results with the model. The chapter aimed to provide novel contributions in demonstrating extensibility of OCBM, which is a shortcoming noted of pedestrian models from the literature. This was achieved by simulating PTI behaviour. PTI behaviour involves multiple different behaviours to be accounted for simultaneously, such as line-of-sight and luggage-laden people. The experimental results found good agreement between OCBM simulations by simple observation and observation from data collection. This ability of OCBM to closely match observations when simulating multiple concurrent behaviours of the PTI helps prove the flexibility of OCBM.

The accuracy of OCBM was further tested by comparing the model to literature observations, through luggage behaviour. The results have shown that the OCBM can match the speed-density fundamental diagram of simulating people with bags better than if simulating a singular larger person. It has shown this by comparing to experimental data of a bi-flow corridor and a double-exit scenario to literature. OCBM matches well against observation, but struggles at high density. This is a known issue for numerous models [149], and special attention and behaviour needs to be used for such high density crowds. Future work could explore how to construct high density constraint-based behaviours for use in OCBM to more accurately represent high density crowds. Experimental comparison to a double-exit room found that OCBM was able to accurately match observations for numerous scenarios. OCBM performed best when the scenario involved larger speeds of people. OCBM has the lowest average error of $0.03 m/s$, which is around three times closer to matching the observations on average than the alternative approaches tested.

The process of understanding the accuracy of the model and validating OCBM was undertaken by comparing simulation outputs to experimental observations. The emergent simulated crowd properties arise through interactions between virtual people. Altering the parameters of individuals has a complex relationship to the emergent properties of the crowd. Having the indirectly controlled properties of the emergent crowd match to experimental findings helps reinforce that a model uses fundamental rules which are suitable for the scenario compared. This process was carried out for both luggage and PTI for particular scenarios.

A limiting factor of OCBM is the number of people that can be simulated in real-time. As shown in the previous chapter in figure 3.19 the real-time limit is around 400 people. This limits the model from visualising large-scale venues like populated

stations and stadiums. It also limits the number of repeated runs possible within a reasonable time. Simulation results use many runs of the simulation per data point. Obtaining results for a wide number of parameters can therefore take a long time to obtain. Increased performance of OCBM will allow for more people to be simulated in real time, and also help run more repeated simulation runs for obtaining results. The performance of OCBM is a key concern of the following chapter.

# Chapter 5

# Accelerated Constraint-Based Framework on the GPU

The OCBM constraint-based framework introduced and evaluated in previous chapters has been examined and tested. Its behaviour has been explored and compared to real data to assess how well it can follow observable behaviour. This framework is able to run in real-time for a sufficiently small number of people and constraints (see "performance", section 3.5). The framework's real time performance was measured in Chapter 3.5 and was shown to be limited to around 500 people with 30 constraints per person. The framework cannot simulate large-scale crowds that number in the thousands of people in real-time. This scale of crowd simulation is commonly required for scenarios such as simulating whole buildings, national public events, sports stadiums, and busy city streets. The performance limitations of the approaches previously described therefore limit the scenarios the OCBM framework can be applied to.

Improving the performance of the OCBM framework will permit simulating more people in real-time. As explored in a previous chapter, notably figure 3.21, the 2D linear program solver is where the code spends the longest time (in particular see "performance", section 3.5 and figure 3.21). Improvements to the linear program solver will have the largest impact with respect to performance improvements. This chapter proposes an implementation approach for optimised linear program solver using a data parallel approach targeted at GPU architectures. An important consideration highlighted in the literature review (section 2.6) is ensuring that GPU computation avoids frequent data transfer between host and device. As such this is a key design goal of the linear solver implementation.

To maximise performance gains that can be achieved using a GPU based linear program solver, the whole OCBM framework will be implemented on the GPU as well. This ensures as much data as possible is kept on-device and minimises slow data transfer.

The following novel contributions are proposed for improving the performance of OCBM:

1. A data parallel approach for a novel 2D linear program solver on the GPU
2. Testing and evaluation of the performance of the novel linear program solver against other state-of-the-art solvers

3. Implementation of the data parallel constraint-based framework on GPU hardware

Section 5.1 creates a novel linear program solver on the GPU, specialised in 2D problems. It details the methodology of the approach as well as theory for implementation choices. Section 5.2 tests the performance of the novel linear program solver against other state-of-the-art software. It performs experiments targeted specifically at low dimension problems. Both these section are implemented using CUDA. Section 5.3 implements the whole of OCBM on the GPU, including incorporating the novel linear program solver, within FLAMEGPU. Finally, the chapter's concluding remarks are in section 5.4. The implementation and testing of the GPU linear program solver has been published as [37].

## 5.1 Linear Program Solving on the GPU

This section will introduce a novel linear program solver on the GPU using CUDA. This will be specialised to target the kind of problems generated by OCBM, which are two dimensional. It solves linear programs in large quantities simultaneously to ensure full GPU device utilisation, maximising performance.

Linear programs are a collection of geometric constraints coupled with a desired value. A linear program solver takes the linear program and computes the optimal solution: the value closest to the desired value that satisfies the constraints. In OCBM a linear program is constructed per person each simulation iteration. Each constraint of the linear program is generated by a behaviour or steering. Each behaviour can result in multiple constraints. The linear program used in the CPU implementation of OCBM is Seidel's random incremental linear program solver [180], the same solver used within ORCA [16].

The proposed linear program solver is referred to as Randomized GPU Batch (RGB). It is optimised in two-dimensions on the GPU, and it builds on the idea on Seidel's incremental linear programming algorithm for GPU architecture [180]. See section 2.5.2 for more details on how Seidel's algorithm functions. Alterations to Seidel's work were made to improve performance when implemented on a GPU architecture, increasing compute and memory parallelism. This was achieved by decomposing the calculation into small work units (WU), which are shared between threads in a block to more evenly spread compute and memory load between threads on the device, compared to solving a linear program per thread. The algorithm is designed and tested for solving multiple two dimensional problems concurrently.

RGB solves linear programs in batches. Batches refers to the grouping together of multiple linear programs into one more complex algorithm. Each linear program is computed independently, but computing resources are shared across the GPU device. This ensures more complex problems receive more computational resources of the GPU. RGB computes how to distribute these resources for the given batch of problems, as well as solve each problem.

While RGB, the proposed linear program solver in this chapter, is developed and explained within the context of a pedestrian simulation, the problem of linear program solving is separable from the pedestrian simulation itself. As such the RGB is applicable to solving linear programs more generally, and as such can be evaluated by testing the performance against other linear program solvers.

There are a variety of options for high-performance linear program solving. There is an open-source CPU algorithm, the GNU Linear Programming Kit (GLPK) [138].

It is an open-source software library that provides a powerful set of tools for solving linear programming problems. It offers a user-friendly modelling language and efficient algorithms to optimise decision-making processes. GLPK supports various LP problem formulations and provides features for sensitivity analysis. Another is CLP, a high-performing [143] open-source CPU simplex solver [44] part of COIN-OR. CLP (Coin-or Linear Programming) is a powerful open-source linear programming solver that is part of the COIN-OR project. It provides an efficient and robust solution for solving linear programming problems, which involve optimising linear problems. CLP utilises algorithms such as the simplex method and the primal-dual interior point method to find optimal solutions. CPLEX [102] is a widely used commercial optimisation software developed by IBM that offers powerful capabilities [143] for solving complex mathematical programming problems. Part of this suit are features for linear programming. A GPU linear program solver is the batch-GPU simplex algorithm of Gurung and Ray [77], used as an algorithm aimed at solving batched linear programs on the GPU. It targets larger dimension linear programs and larger sized individual problems, compared to RGB.

These software programs are optimised for solving general linear programs. For any given problem size, and whether sparse or dense, they are suitable for providing good performance and an optimised approach for solving the problem. Gurung and Ray [77] is an outlier in that it is more focused on batch solving multiple problems quickly. All these models suffer from the same apparent weakness, which is that they use the simplex model to solve the linear program. Performance is impacted because the computation required for calculations, such as matrix inverse, is an expensive operation. This creates a motivation for constructing a linear program solver focused on performance for low dimension linear programs, which RGB aims to provide.

### 5.1.1 Implementation of the GPU Linear Program Solver

GPUs use a programming model of a grid of thread blocks to oversubscribe an architectural model consisting of multiple streaming multiprocessors, each containing a set of CUDA cores. Each streaming multiprocessor in a GPU consists of several CUDA cores that are responsible for executing individual threads. These CUDA cores are organised into groups called warps, which consist of a fixed number of threads (typically 32). The GPU architecture also includes a hierarchy of memory subsystems designed to optimise data access and minimise latency. The global memory serves as the main storage space accessible by both the CPU and GPU, although accessing global memory can incur high latency. To mitigate this latency, GPUs employ various levels of on-chip memory, including shared memory and local memory. The literature review details GPU memory in greater detail. Of particular note, shared memory is a fast, low-latency memory that is shared among threads within a thread block. It allows for efficient data sharing and communication between threads, promoting cooperation and synchronisation. Local memory, on the other hand, is private to each thread and resides in global memory. It is used when the data does not fit into the registers or shared memory and is accessed with higher latency compared to shared memory. Modern GPUs include advanced features such as warp schedulers to efficiently manage and execute threads. These features enable the GPU to handle massive parallelism and exploit thread-level parallelism within a warp. GPUs leverage the parallel processing capabilities of numerous CUDA cores organised into streaming multiprocessors. This model allows GPUs to achieve high computational throughput and performance for a wide range of applications

GPU data parallelism refers to a programming paradigm and execution model that allows for the simultaneous execution of the same operation on multiple data elements using a single instruction. Data parallelism takes advantage of the parallel processing capabilities of GPUs, which are specifically designed to handle massive amounts of parallel computations. By applying the same operation to multiple data elements simultaneously, GPUs can achieve significant speedup compared to traditional single-threaded or CPU-based approaches. GPUs have a large number of processing cores that can execute operations concurrently. Data parallelism allows for the distribution of computational tasks across these cores, maximising their utilisation. This efficient utilisation of GPU resources enables higher throughput and allows for the processing of large data in a timely manner.

To implement simple incremental batches of linear program solving on the GPU, each linear program is assigned to a core/thread. Thus the number of active threads is equal to the number of linear programs to be solved. A naive implementation of Seidel's algorithm would result in a large divergence in calculations between threads, as illustrated in Figure 5.1. In this figure some threads require large amounts of computation while others require very little. This imbalance can be attributed to all threads within a warp considering the same index of constraint. In this context, some problems (threads) will be satisfied with the considered constraint while others need to recompute the intermediate optimal solution. This divergence causes an imbalance of workload within a warp and hence results in poor performance.

To address the problem of imbalance, the idea of cooperative thread arrays can be applied [105, 205]. In this approach, threads within a warp or block communicate with each other to share the workload. In the RGB algorithm the most intensive computational aspect is performing the set of 1D linear programs for all previous constraints (see equations 2.3 and 2.4). This was experimentally observed through timing, profiling and analysing of code performance, where around 80% of the time was spent in this computation. Each constraint to perform a 1D linear program can be thought of as the smallest quantity of work, referred to as a work unit (WU). These WUs can be distributed across threads in a block so parallel computation is more balanced. The requirement for this is that the writing of the results $u_{\text{left}}$ and $u_{\text{right}}$ must be done atomically into shared memory. These $u$ values keep track of the left and right edges of the solution in one dimension. Running best values of $u_{\text{left}}$ and $u_{\text{right}}$ are tracked and updated with each constraint by comparing the intersection of the constraint with the current best values of $u$. Atomic writing ensures no race conditions occur for the results. This ensures the correct result but reduces performance compared to a standard write to memory. Other techniques are available to use, such as reduction, but atomic operations work well for unknown set sizes at runtime, and atomic operations have improved in performance with recent hardware [154]. Examination into shared memory atomics shows that on Maxwell hardware (and later) shared memory atomics outperform global atomics [178] and device-wide segmented reduction. Shared memory atomics also has stable performance across a range of workloads, an important aspect for the RGB algorithm where many different amounts of contention are present.

Figure 5.2 highlights the use of cooperative thread arrays. The amount of work required is computed and distributed across all available threads. The sharing of data is done through shared memory. This creates more even workloads which increases the parallelism of the problem and reduces the overall time for computation.

Communicating data between threads in a block is done through shared memory, a region of memory that any threads in the same block can access on chip. Threads will read appropriate constraint data from global memory, stored on device

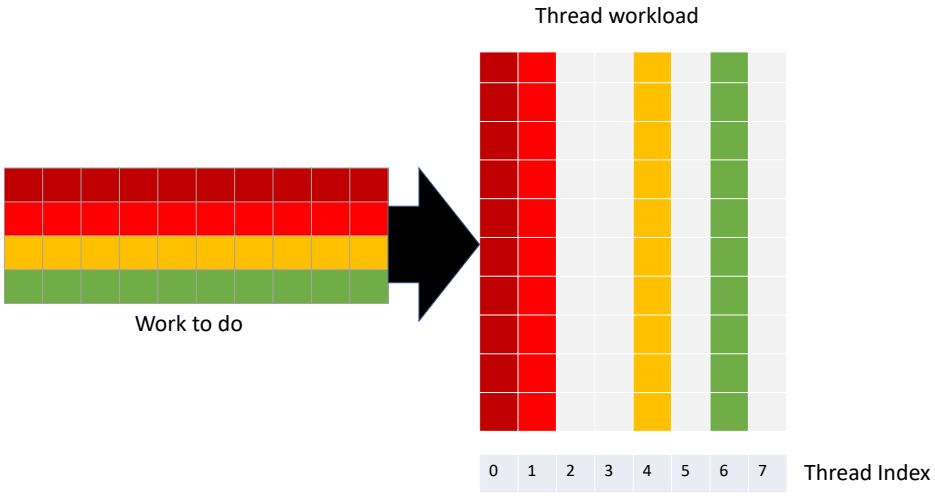



FIGURE 5.1: Visual demonstration of how work is distributed across GPU threads using a naive mapping of work units. This approach is referred to as NaiveRGB. Imbalanced workloads reduce the computation parallelism. The colour shows which linear program the work units original from. Threads 2, 3, 5, and 7 are not performing any computation.

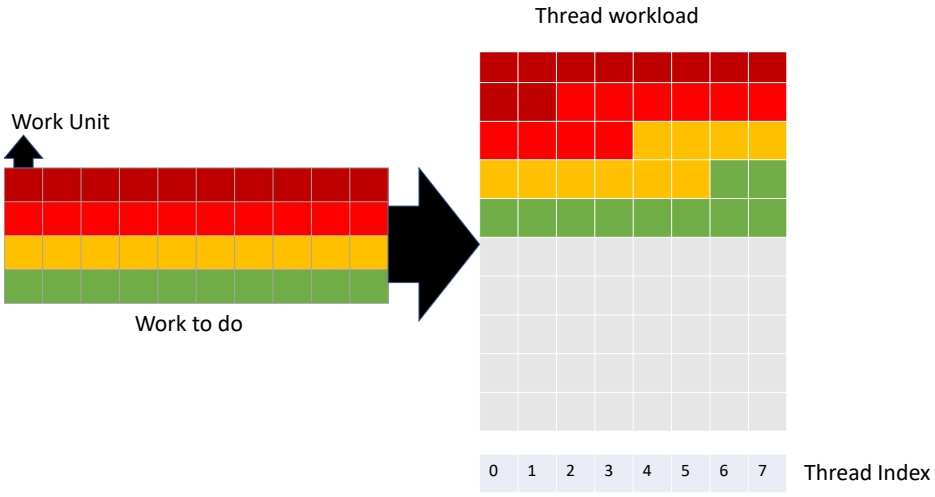



FIGURE 5.2: Distribution of the workload across a warp after use of co-operative thread arrays to balance work units. Work units are distributed evenly across all available threads. Each thread has a similar amount of computation as every other. This optimised approach is referred to as Optimised RGB. The colour of each work unit, corresponding to which linear program it originated from, is distributed across all available threads.

DRAM, and write it to shared memory. It is preferable to store this data in shared memory due to low latency access within a block. Since different threads make numerous reads to the data, shared memory is far more efficient than the alternative global memory, which takes around 100 times longer to access [153]. Shared memory is limited in size so only the most accessed pieces of data can be stored. Shared memory ensures full control over the data stored on device and permits optimisations of the data that could not be achieved by the compiler. The alternative, caching, uses the same physical bit of chip, but must rely on compiler optimisations which cannot be as effective. The remaining data is loaded from global memory and access to such memory is coalesced as much as possible to avoid excessive cache line fetches. Vectorized loads are used to reduce the number of memory requests and increase the utilisation of cache lines where any scattered reads are required. Since information of half-planes is stored in multiple variables (2D position and direction) combining the information into one extended set of data ensures that scattered reads use as much of each cache line as possible. Memory transfer between CPU and GPU is managed through the use of CUDA Managed Memory. This lets the underlying runtime handle the paged transfer of data to and from the device intelligently. It removes the requirement that all data be copied to the device at kernel launch, instead paging in memory as demanded and asynchronously during kernel execution. This reduces the time of copying memory to and from the device. It also allows for allocating memory up to the size of system memory, rather than being restricted to a maximum of the dedicated GPU DRAM. This is important in large problems when the device DRAM is too small, but system DRAM is large enough.

An overview of the algorithm is provided in Listing 5.1. The algorithm runs as many threads as there are problems in a batch to solve, `p`. The program incrementally examines each constraint in a problem and loops over the maximum problem size. The current line to consider is read into shared memory from global device memory. This read uses coalesced memory access for optimal performance. It checks whether the linear program solution, stored in `S`, is satisfied by this line in `unsatisfied` and writes this logical check to a binary value `B`. A `sync_block()` call ensures all data has been written to before it is accessed, avoiding race conditions. The binary value `B` is reduced to calculate the total number of problems in need of recomputation in the block. All threads are mapped to a work unit, a unique tuple of problem and line, and this is repeated over all lines for each problem in need of recomputation. The work unit calculation reads in the assigned line from device memory, `line`, and calculates the intersection point between this `line` and the considered line in shared memory, `SML`. The result is written to shared memory using atomics to avoid race conditions. Another `sync_block()` call ensures all calculations are complete before updating the solution.

LISTING 5.1: RGB GPU Algorithm Overview

```
1  //p is batch problem size
2  //lp_max is maximum LP size
3  //L is a device vector of m × lp_max line constraints containing all
       constraint lines
4  //SML is an empty shared memory vector of length m, containing current
       constraint line
5  //S is an empty device vector of length m, containing the solution
6  //SMS is an empty shared memory scalar of length threads in block,
7  //B is an empty shared memory scalar of length threads in block
8  //block_width is the width of the CUDA Kernel block configuration
9  gpu_parallel_for idx ← 1 to m
10
```

```
11    bidx ← idx % block_width
12
13    for n ← 1 to lp_max
14        SML[bidx] ← L[idx][n]
15        B[bidx] ← call unsatisfied(SML[bidx], S[idx])
16        call sync_block()
17
18        active_threads ← block_reduce_sum(B)
19        wu_count = active_threads * n
20
21        j ← bidx
22        while j < wu_count
23            map_idx, map_n ← map(j)
24            map_bidx ← map_idx % block_width
25            line ← L[map_idx][map_n]
26            i_pos ← call intersect(SML[map_bidx], line)
27            SMS[map_bidx] ← call atomicMin(SMS[map_bidx], i_pos)
28            j ← j + block_width
29
30        call sync_block()
31
32        if not B[bidx]
33            S[idx] ← SMS[bidx]
```

## 5.2 Experimental Performance of the GPU Linear Program Algorithm

In this section, results obtained from running the RGB algorithm using the co-operative threading approach are presented and compared against four other algorithms: (i) an open-source CPU algorithm, the GNU Linear Programming Kit (GLPK) [138]; (ii) CLP, a high-performing [143] open-source CPU simplex solver [44]; (iii) CPLEX, a high-performing [143] multi-core CPU solver [102]; (iv) the batch-GPU simplex algorithm of Gurung and Ray [77], used as an algorithm aimed at solving batched linear programs on the GPU. The naive unbalanced implementation of the RGB algorithm without co-operative threading (as described in section 5.1) is referred to as 'NaiveRGB' and is also tested for comparison.

The GLPK algorithm is parallelised over linear programs, allowing different threads to solve separate problems, and is referred to in the results as 'mGLPK', standing for 'multicore-environment-GLPK'. CPLEX is able to use different methods to solve linear programs – in the tests, the automatic algorithm selector was used, which allows the underlying solver to choose which algorithm it believes is most suitable. CLP is a single-core solver and is set to solve using the dual simplex method. Tests were also run for GLPK in a serial manner. However, the results of this are not shown as performance for the multicore-environment version (mGLPK) was better, with improvement in performance of up to 6 times, i.e. the number of cores on the CPU tested, which shows linear scaling with the number of CPU cores.

All GPU experiments were run on an NVIDIA Titan V GPU card with 12GB dedicated memory and a 6-core Intel i7-6850K with 64 GB RAM. The GPU was connected by PCI-E 2.0. The GPU software was developed with NVIDIA CUDA 8.0 on Ubuntu, and CPU code was compiled with gcc 7.3. The algorithms were timed after the problem had finished initialising on the CPU, and ended when the result had been written to CPU-usable memory. In the case of GPU timing, this included data transfer to and from device as a result of CUDA managed memory paging.

Problem sets are generated using random feasible constraints in two-dimensions: constraint lines are generated randomly and tested to ensure a solution is possible. Only one linear program is generated per run, and copied multiple times into memory to simulate batch numbers. Due to numerical deviations between CPU and GPU floating point accumulation, a tolerance value of 5 significant figures is set on the results to ensure that consistent results are obtained for all algorithms. These problems are repeated multiple times with new random feasible problems, with the error bars representing one standard deviation of uncertainty.

An important computational aspect of the algorithm is the performance of the atomic reduction. Figure 5.6 shows the performance comparison of shared memory atomics, global memory atomics and device-wide segmented reduction (using the CUB library [141]) over a range of contention. Contention is a measure of how many elements must reduce into a final value. A contention of 2 means every 2 elements must reduce their values together. This experiment was designed by implementing shared memory atomics so as to have tight control over the contention value outside of the linear solver experimentation. The maximum measured contention is 512, chosen as the block size of the kernel. Results show shared memory atomics to be consistent in timing and better performance in comparison for all contention measured. Shared memory atomics also has stable performance across a range of contention, an important aspect for the RGB algorithm where many different values of contention occur.

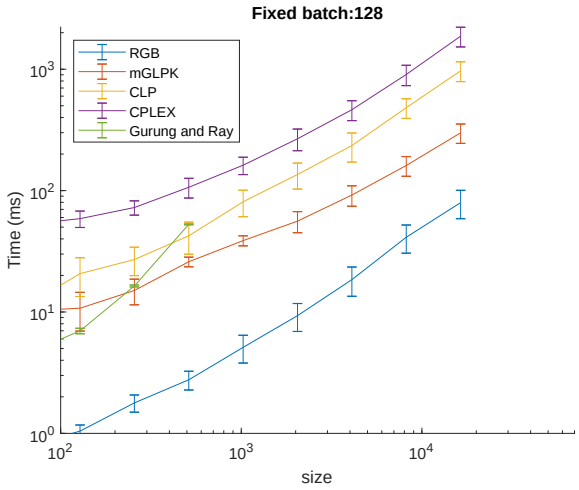
### 5.2.1 Results

The results shown in Figures 5.3a - 5.3c use fixed batch sizes and measure the time taken to solve all LPs when varying the sizes of problems. The RGB algorithm can be seen to outperform the other algorithms above sizes of $2 \times 10^2$. The algorithm of Gurung and Ray was limited to smaller sizes, and can be seen to improve compared to CPU algorithms as the number of batches increases.

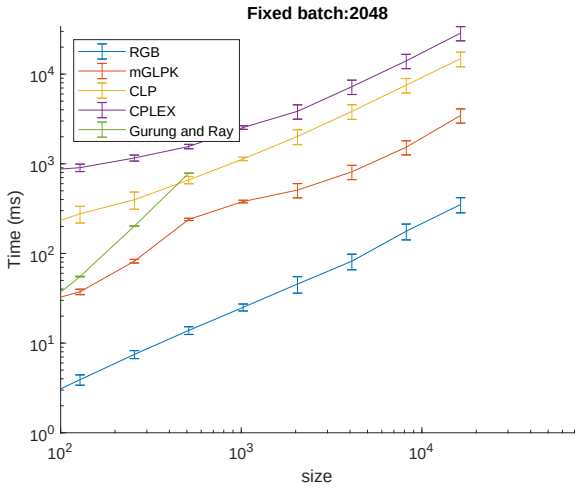
Figures 5.4a - 5.4b show results from the same experiment, but with varying batch numbers and fixed problem sizes. In Figure 5.4a, the RGB algorithm can be seen to outperform the other algorithms for all tested batch sizes with linear program sizes of 128. In Figure 5.4b, the RGB algorithm outperforms the CPU implementations for all tested batch sizes. The Gurung and Ray algorithm was not able to be tested at this large problem size due to GPU memory limitations.

For large batch sizes the majority of the execution time of the RGB algorithm is due to memory initialisation and transfer. For these larger sizes the computation kernel takes less than 30% of the total execution time, with the remainder being used to manage memory. This is highlighted in the surface plot in Figure 5.5. The bright yellow area represents size-batch problems which use more than 40% of the time to initialise and transfer data. The dark blue region is where the majority of time is spent performing computation. It shows that as batch amounts increase, the proportional amount of time spent transferring memory also increases.

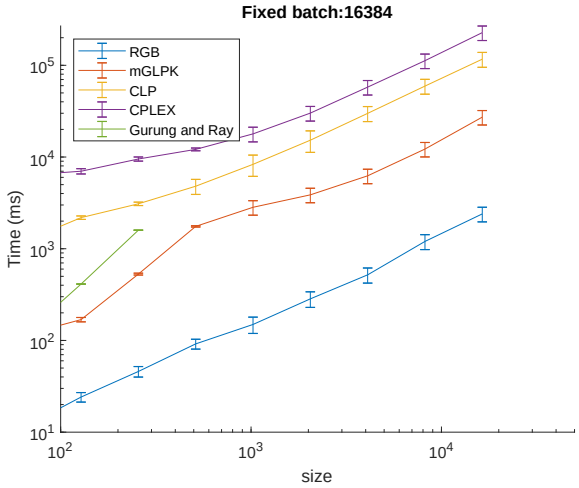
The performance difference between NaiveRGB and optimised RGB is examined in Figure 5.7. To highlight the difference in performance, the relative computation kernel execution time of NaiveRGB and optimised RGB are shown, which ignores the time taken for data transfer. This relative time is the speedup the optimised implementation has over the naive version. The y-axis shows the speedup of the optimised RGB algorithm over the NaiveRGB implementation, with a value of 1 meaning both algorithms execute in the same time.




(A) Timing vs varying linear program sizes for fixed batch amount 128

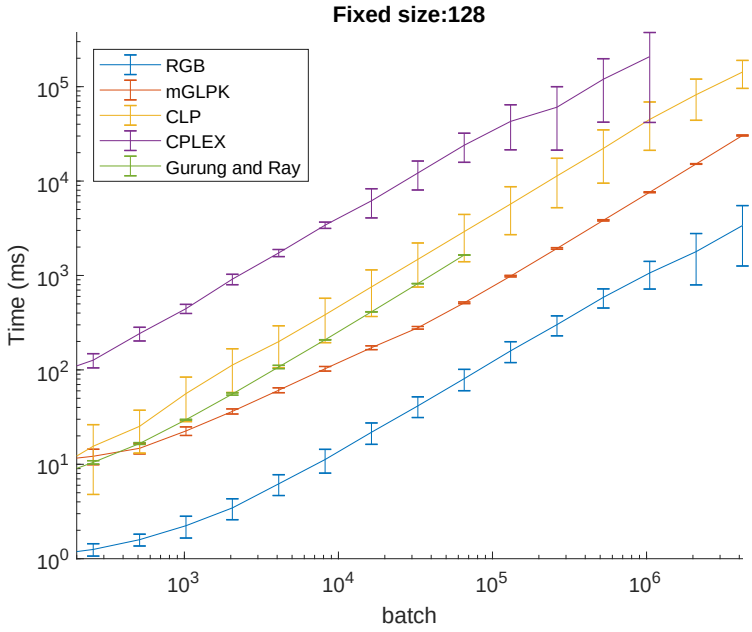



(B) Timing vs varying linear program sizes for fixed batch amount 2048

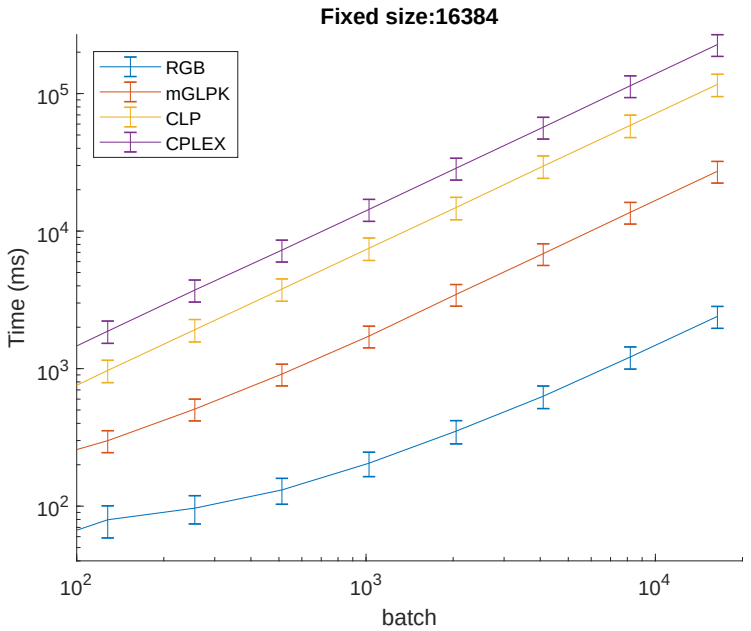



(C) Timing vs varying linear program sizes for fixed batch amount 161384

FIGURE 5.3: Timing comparison of the three algorithms for fixed batch sizes and varied linear program sizes.




(A) Timing vs varying batch amounts for fixed size 64 constraints




(B) Timing vs varying batch amounts for fixed size 8192 constraints

FIGURE 5.4: Timing comparison of the three algorithms for fixed constraint size and varied batch amounts.

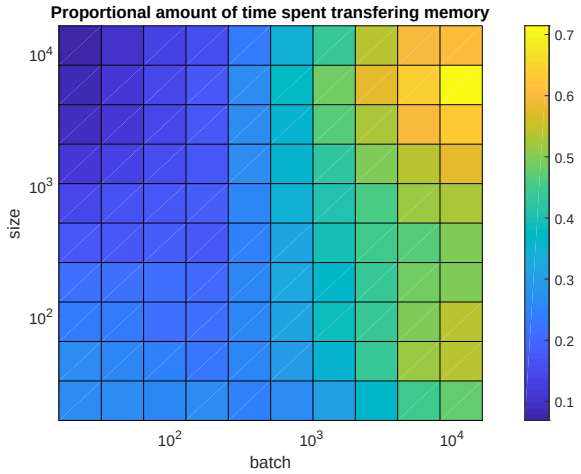



FIGURE 5.5: Proportion of time spent copying memory compared to total execution time for RGB. It compares against both problem batch sizes, and the size of individuals problems which make up a batch.

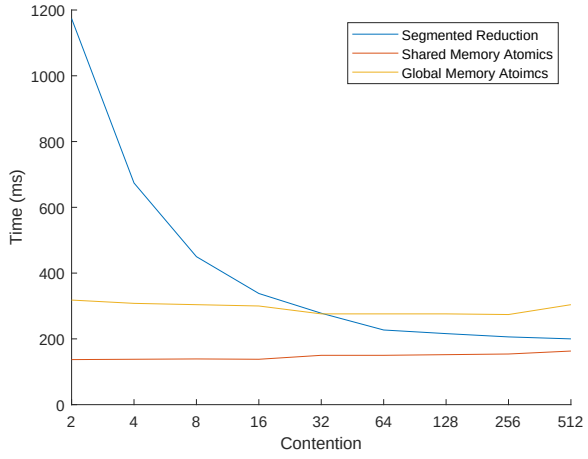
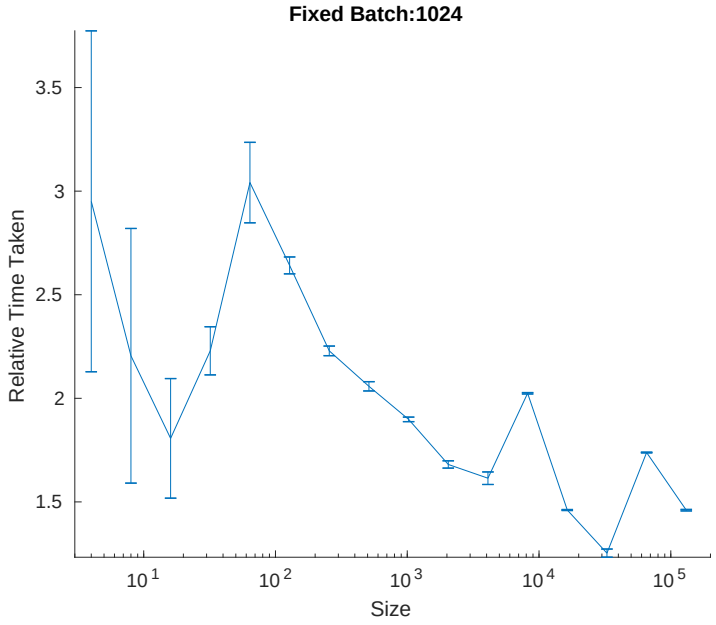
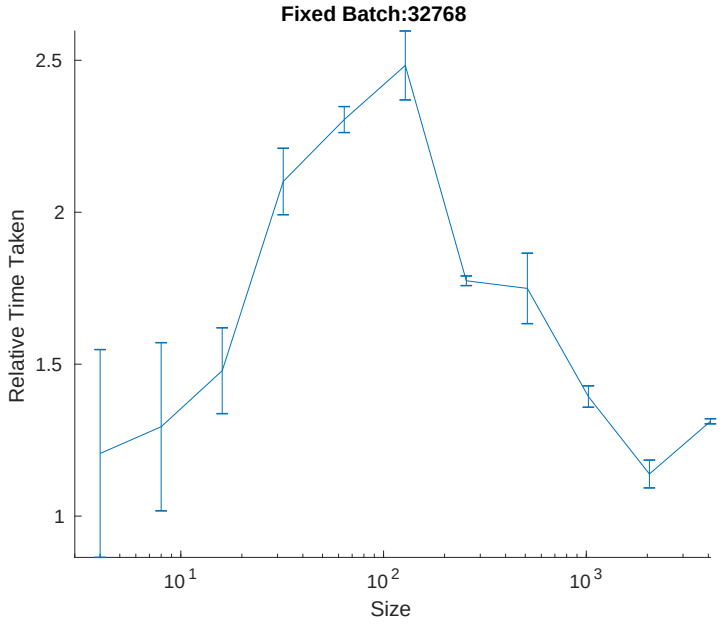



FIGURE 5.6: Performance vs contention for atomics and device-wide segmented reduction. A lower value means the segmented reduction is computed faster.




(A) Relative timing of NaiveRGB and RGB implementations vs varying linear program sizes for 1024 problems




(B) Relative timing of NaiveRGB and RGB implementations vs varying linear program sizes for 32768 problems

FIGURE 5.7: Relative timing comparison of naive NaiveRGB and optimised RGB algorithms for fixed constraint size and varied batch amounts.

### 5.2.2 Discussion of the GPU Linear Program Algorithm

The purpose of the experimentation was to understand if GPU acceleration could be effective in accelerating batch linear solving. In all cases the GPU RGB implementation was shown to outperform the alternative state of the art approaches, regardless of architecture. Examining the scaling of the approaches offers further insight into the performance characteristics of each approach. The results show that there is a trend for a greater speed-up to occur for RGB against CPU algorithms as batch amounts increase and linear program sizes decrease. There is a similar trend for a speed-up for RGB against the algorithm of Gurung and Ray, as batch amounts increase and linear program sizes also increase. This means that the CPU models scale better to larger linear program sizes but do not scale with batch amount. This is expected due to the powerful serial performance of the CPU. This can be seen in figure 5.4 where the time taken to complete execution increases at a greater rate for mGPLK and CLP than RGB when batch sizes are increased. Figure 5.3 highlights the difference in time scaling for varying batch amounts. The batch method of Gurung and Ray can be seen to scale much worse than RGB in these figures. This difference in scaling is due to the RGB model using methods suited to two-dimensional constraints, such as repeatedly running one dimensional LPs, which is computationally light compared to the operations used in the batch simplex algorithm. mGLPK tends to be the best performing out of the CPU models tested due to the multicore environment in which it is run allowing it to be better suited for solving batch amounts.

Maximum speed-ups of the GPU RBG approach are reported as 66x against mGLPK algorithm, and 22x against Gurung and Ray's algorithm. The relative performance of the naive and optimised RGB algorithms peaks in relative difference around $10^2$ for figures 5.3 and 5.4. This is due to launching block sizes in such a way as to be optimised for this size. Various operations used, such as block compression, require a fixed block size at compile time. Further improvements to performance of the optimised RGB algorithm can be made by tailoring block sizes to the expected linear program size. A limit in improving the optimised RGB algorithm performance is the memory bandwidth limit. Using GPUs with greater memory bandwidth should show increased performance over the other tested methods.

The RGB algorithm is the best performing out of the algorithms tested when the GPU device is being fully utilised. For problems that cannot fully utilise the device's cores, CPU algorithms mGLPK and CLP are able to solve the problems faster. Full device utilisation occurs when there are more constraints to consider than cores in the device, i.e. No.-of-batches $\times$ size-of-batches $\gg$ gpu-cores. This is observed to occur at roughly twice the number of cores for the GPU tested. An example region of this case is for small batch sizes, such as one or two problems only. For the tested hardware, a Titan X Maxwell, this is 3072. Above this amount all cores are used in the computation. Below full device utilisation it is hard to hide the large latency that is inherent on the GPU, which can be hidden by exposing greater levels of parallelism.

This chapter has presented a novel implementation of a low-dimension batch linear program solver suitable for GPUs. The implementation makes use of cooperative thread arrays to share workload across threads. The RGB algorithm outperforms Gurung and Ray's work [77] for all problem sizes above 300 constraints in two-dimensions. It also outperforms the CPU algorithms, GLPK [138] and CPLEX [47], for problems that are known to fully utilise the device.

Timings were measured to include data transfer to and from device (i.e. between CPU and GPU). It should be noted that if the batch-GPU linear program algorithm is just one stage of computation that is taking place on the GPU then the time of

copying data from CPU to GPU can be improved. This can vastly increase performance at larger data sizes or for iterative problems such as pedestrian simulations and collision avoidance. When utilising the whole device the problem is memory bound. As such, further improvements to the implementation should optimise data loading. For this implementation the use of CUDA Managed Memory provided sufficient data performance, though this could be improved upon on a per-device basis.

Since the experimental work conducted in this chapter there have been further advances in solver implementations. One prominent linear program solver is the cuSolver library [152] by NVidia. It "provides a collection of dense and sparse direct linear solvers and Eigen solvers". Of particular interest is its ability to solve multiple independent programs in batch methods. The code to the solver is closed source so discussing how the implementation is undertaken is difficult. The user is able to query the API to know how many programs can be solved within a batch, and thus if the API needs to be repeatedly called to solve multiple batches. It does this by calculating the memory of the device, and how much memory one item of the batch will take, to compute this. cuSolver has numerous routines that can handle sparse and linear problems, but does not appear to have routines for the specialised case of low dimensions. This indicates that while cuSolver is powerful and flexible, it lacks the specific performance features for low dimension solvers. It appears to solve problems in batches, but only when all problems in a batch are of the same size. It is able to utilise GPU hardware to solve problems, but the algorithm choices it uses makes it more suitable, like with the approach by Gurung and Ray, to larger problem sizes.

The domain of the presented model is best suited to solving large numbers of low dimensional LPs, each of which has numerous constraints. Spatial problems in two or three dimensions are those that this method could be performant in solving. An advantage of the model is the allowance for different-sized individual LPs within the batches. The distribution of work units ensures that workload is balanced regardless of variance in batch problem sizes. Future directions could examine the applications and performance of the model extended to higher dimensions. It is expected to scale favourably for low dimensional problems, up to around 5 dimensions, due to the efficiency of the core solving algorithm at such low dimensions.

One clear domain that the RGB model is suitable for is the ORCA steering model. Each simulation step will have a linear program per agent/person, and the size of each program will vary depending on the behaviours and the number of neighbours. By having all the computation remain on the GPU it avoids slow host-device memory copies and thus gains even more performance improvement than discussed within this chapter. The problem of pedestrian steering is two-dimensional so it creates two-dimensional linear programs. This method should also be applicable to three-dimension steering simulations such as modelling certain fish-like agents in water, or swarms of drones in the air.

## 5.3 GPU OCBM with RGB

This section describes the implementation of the OCBM constraint-based framework on the GPU using FLAMEGPU. It provides reasons for this, both theoretical, and experimental. It explores why the whole framework should be on the GPU if the linear program routine is. It also explains how the GPU framework is implemented.

### 5.3.1 OCBM on the GPU

When the GPU needs memory not within the device itself, the transfer speed is much slower than any other method. This detriment can be minimised by selecting special memory, or optimising how the memory read/writes occur, but the best choice by far is to not perform any CPU-GPU memory transfer. CPU memory is usually much larger, as it is limited by RAM. This memory is, however, very slow to read on the GPU. A key challenge for GPU implementation is to make use of the most relevant memory location, to access it efficiently (see 2.6 for more information), and to read only what is necessary. Copying data to and from CPU to GPU is one of the slowest memory transfer speeds. Unfortunately it is unavoidable in order to communicate information between the host and device. Care needs to be taken to both minimise the amount of transfer data required, as well as coalesce the memory to make most efficient use of cache line reading and writing.

The performance experiments earlier in this chapter have tested the RGB algorithm. A large performance hit was due to the CPU-GPU memory transfer. This transfer was required to fairly test between the different solvers, ensuring that the problems could be initialised and the results obtained on the CPU.

To fully make use of the performance gains of the RGB algorithm within the constraint-framework, the whole pedestrian simulation would benefit from being on the GPU. This would minimise CPU-GPU memory transfer, and the performance hit that comes with it.

### 5.3.2 Constraint-Based Framework Implementation on the GPU

Microscopic pedestrian simulations, like the framework within this thesis, can be described as agent-based. Each person, or agent, has a state, and follows the same set of rules as everyone else. There are many examples of agent-based models (ABM)s on the GPU [222, 147, 26]. A framework for implementing ABMs is FLAMEGPU [115]. All agents have a small and finite number of different states they can be in, and all agents within the same state obey the same rules. This lends itself well to GPU computation which excels at performing parallel computation on different data. A key requirement for performant GPU ABM is large numbers of agents. This is to ensure the device is fully utilised. For more information on GPU architecture, refer back to section 2.6.

FLAMEGPU uses a number of optimisations to improve general agent based models. One notable optimisation is spatial partitioning of agent communication. Some information must be observed by people in the model. Examples are the radius, speed and position of others nearby. In order to communicate this information between people we use the idea of messages from the FLAME GPU framework [172], which is demonstrated in figure 5.8. Each agent creates a message which contains information on observables about themselves. Each message is assigned a spatial location equal to the position of the agent in the simulation. These messages are organised into spatial bins. Each agent will then read the messages from its associated bin, and those neighbouring. This method is far faster than a brute-force read all approach for large spaces and many people/messages. The associated overhead in organising messages into bins outweighs the cost of reading all messages and discarding those far away. A possible alternate implementation uses a KD-tree spatial partitioning. It is expected, from the work of Li and Mukundan [131], that this grid based spatial partitioning is faster than a KD-tree implementation on the GPU.

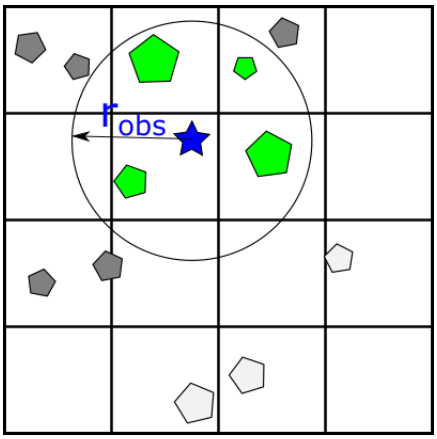



FIGURE 5.8: FLAME message partitioning. The simulation is discretized into spatial bins and people save their message to the corresponding bin. For a given person (blue star), it does not read messages of those in non-neighbouring bins (white pentagons). For those within the same or neighbouring partitioning bins, it calculates whether they are within the observation radius $r_{obs}$. If not, they are ignored (grey pentagons). If they are within the observation radius (green pentagon) the person is aware of them and will attempt to avoid them accordingly, by generating corresponding ORCA half-planes of valid velocities.

A simplified analogue to the functions on the CPU is GPU kernels. A kernel initialises grids of thread blocks to perform procedures that are executed in parallel on the GPU. Functions on the CPU such as lp2, constraints, groups, from figure 3.21 are created as GPU kernels. On the CPU, the lp2 and lp3 routines call the linear program solver as part of their functionality. The lp2 function calls the linear program solver and attempts to solve the initial linear program. If there is no feasible solution, lp3 is run, which repeatedly calls the linear program solver routine, each time shifting constraints according to their hardness until a valid solution can be found by the linear program solver. On the GPU the linear program solver called by lp2 and lp3 is the novel RGB solver. The concept of lp2 and lp3 between CPU and GPU is the same, but changes are made so they run as GPU kernels.

When a linear program does not have a valid solution, the constraints are shifted until one is found. In a given simulation step, some problems may be immediately solvable, while others may need numerous shifts. Each time the constraints are shifted, the linear program routine runs again to check if there is a solution. If there is no solution, the constraints are shifted more. This means that some linear programs can be solved in the first attempt, while other linear programs take much longer to solve. This can be a cause of computational imbalance. To address this, the size of grids of thread blocks adjusts each iteration. The first iteration has enough for every person in the simulation. Then the number of linear programs which failed is collated, and a second iteration of RGB is performed with a smaller batch size, whose linear problems are those which had no feasible solution the first time. This process is repeated. Each iteration, fewer linear problems are performed. These iterations all occur within one time step of the simulation. The process completes once each linear program has a solution.

## 5.4 Summary

This chapter has described GPU OCBM, an optimised GPU implementation of OCBM using the novel linear program solver, RGB. The chapter has detailed how the linear program solver routine, a routine previously noted as a suitable target for optimisation, has been optimised and implemented. By decomposing GPU computation of the linear program into Work Units (WUs), and distributing these WUs across the device, RGB is able to solve batches of linear programs efficiently. This relies on grouping multiple different linear programs together to be solved together. Performance experiments found that RGB is able to outperform other state-of-the-art software for problems which fully utilise the device. Testing included CPU-GPU memory transfer, which was the slowest part of the solver, especially at larger problem sizes which require more memory transfer across devices. The testing was performed on fixed problem sizes within each batch. This was done because of limitations of some of the other software tested against. RGB is able to compute different linear program sizes within each batch.

It shows that as batch amounts increase, the proportional amount of time spent transferring memory also increases.

RGB, which efficiently solves multiple different sizes of linear programs, is useful for use in OCBM for performance improvements. It solves batches of different sized linear problems, which arises in OCBM with each person having different numbers of constraints. OCBM is implemented entirely on the GPU to minimise CPU-GPU memory transfer. This memory transfer becomes more impactful the larger the linear programs are. Linear programs in OCBM are larger when considering more behaviours and when accounting for a larger number of neighbours. By reducing the memory transfer, OCBM is able to scale with better performance to more complex behaviours. The chapter explained how GPU OCBM was implemented, and changes required to incorporate RGB. This includes how to handle cases of infeasible solutions, and the result combines RGB with constraint hardness to find the

With the performance improvements achieved within this chapter, the next chapter demonstrates GPU OCBM in simulating large real time crowds. The next chapter will also explore the performance of GPU OCBM to understand how it improves against the CPU implementation, and potential avenues for further performance optimisations.

# Chapter 6

# Demonstrating the Optimised Framework

The thesis has explored the use of constraint based modelling and proposed OCBM, a novel constraint-based pedestrian model. The previous chapter has improved the performance of more general constraint-based linear solving by proposing a specialised batch linear program solver for the GPU. The chapter described how the OCBM modelling approach has been implemented using the GPU solver. The chapter highlighted how the whole pedestrian simulation process was implemented on the GPU to minimise the amount of low performance CPU-GPU memory transfer.

The performance of the linear solver was examined within the context of an artificial benchmark in the previous chapter. However, the performance of OCBM on the GPU is still to be evaluated. This chapter will demonstrate OCBM on the GPU through a visual simulation of a large-scale crowd and through examination of the computational performance. It will demonstrate the framework visually by displaying single frame renderings of the framework extracted from a simulation. Visual analysis will be used to evaluate the flexibility and scale of the model. The performance of the GPU accelerated OCBM framework will be examined using similar experiments to the previous performance experiments of section 3.5. These simulations are able to make use of the previously explored behaviours of groups and luggage, which are described in section 3.3. Validation and accuracy experiments have been previously performed on these behaviours in chapter 4. Autonomous vehicles are not included due to a lack of available appropriate data. The experiments in this chapter are for a performance study of the GPU implementation, and contribute towards understanding the real time performance of OCBM on the GPU.

This chapter will demonstrate the OCBM on the GPU when applied to a large-scale crowd in real-time. The work will contribute to the thesis research by:

1. Visually displaying large real-time crowd simulations of GPU OCBM.
2. Evaluating the performance of GPU OCBM.

A previous iteration of a performance study of OCBM on the GPU has been published in [38], and parts are used within this chapter.

The chapter begins by displaying OCBM simulating large-scale crowds in real-time in section 6.1. These crowds are limited to only steering behaviour. This is to

avoid too much visual detail on a scene with many moving people. The next section tests the performance of the GPU framework by considering how the implementation scales through model parameters in section 6.2, following a similar set of experiments that were performed on the CPU version in section 3.5. Finally, a conclusion of the findings of this chapter is provided which considers the benchmark observations and explores the suitability of the OCBM model and its GPU implementation with respect to real time applications is section 6.3.

## 6.1 Visual Demonstration of the Simulation

To demonstrate the OCBM implementation on the GPU, two different scenarios are provided. These will visually display the OCBM. The first is a two way crossing in open space. The second is an eight-way crossing. The simulations are visualised using the Unreal Engine, which specialises in rendering 3D models in real time. In both scenarios the experiment is designed to observe the interactions of the crowd as they pass through the environment, causing an area of congestion. The first experiment is expected to demonstrate head-on flows between crowds and the phenomenon of lane formation. The second experiment is expected to demonstrate vortex behaviour.

### Two way

This experiment simulates two crowds totalling a population of 2,500 people at opposite ends of an environment which must navigate past each other to reach their objectives. This experiment demonstrates the behaviour and correctness of the model. The virtual people exist in an open space (without obstacles) attempting to steer past each other. Previous chapters have already examined the accuracy and validity of additional behaviours. The use of behaviours on performance has already been examined. It showed that the addition of behaviour constraints does not affect performance, only the total number of constraints (figure 3.20). For these reasons, no additional behaviours are included in the simulation, only steering.

Two virtual rectangles are created on either side of the environment (see the large red and blue boxes of figure 6.1. The "Red" people spawn somewhere within the red box, and must navigate to a random location within the blue box. Because the starting region of one group is the same area as the goal region of the other it forces the two groups to navigate past each other. Once a person reaches the goal location they are removed from the simulation. Once all people have reached their goal the simulation ends.

People have different sizes and different maximum speeds. Figure 6.1 illustrates this. In this example all people have an equal chance of being of radius 0.5m, 0.75m or 1m (shown by person size in the figure, as well as S, M and L on their tops) and, independently, an equal chance of a desired speed of 1m/s, 1.33m/s or 2m/s. The maximum speed is adjusted to be 125% of the desired speed. In figure 6.1, people moving in the $x$ direction (left to right) have red tops, and people moving in the negative $x$ direction (right to left) have blue tops. Brighter shaded tops indicate the largest desired speed, 2m/s, and the darkest tops indicate the slowest speed, 1m/s.

### Eight way

The second test case was an eight-way crossing, visualised in figure 6.3. Each crowd must navigate 135 deg across the environment, resulting in a vortex-like pattern around the centre. Figure 6.2 shows a diagrammatic setup of the simulation. People

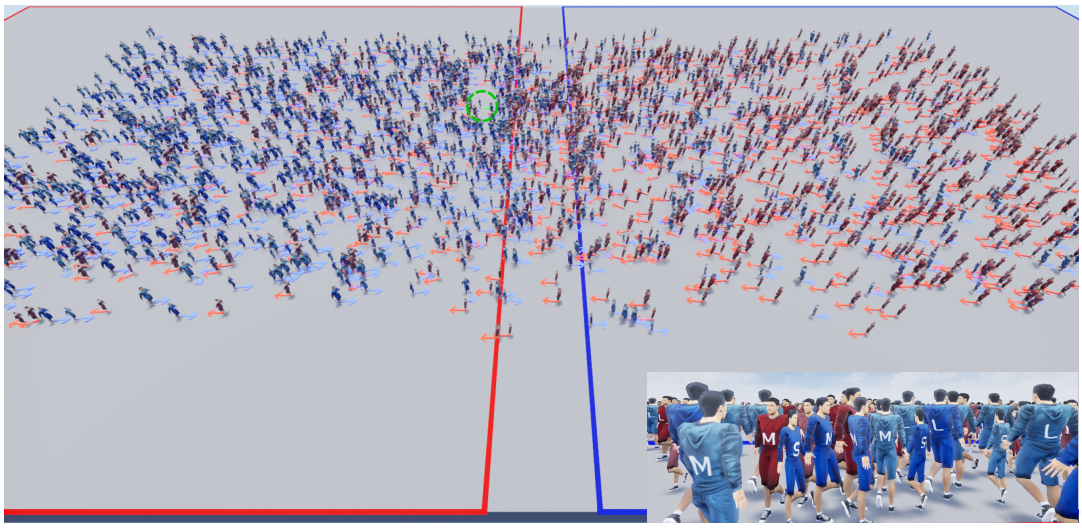


FIGURE 6.1: Visualisation of 2,500 people in Unreal. Two crowds navigate past each other, one heading from left to right (blue clothes) and the other from right to left (red clothes). People are colour coded according to their maximum speeds (using three shades of red or blue, respectively) and have varying radii (indicated by their actual size and also using S, M and L on their tops). Top: scene view from above; One pedestrian is highlighted with a green circle; Inset: view from the perspective of the pedestrian in the green circle

starting in the blue square (top) must navigate towards the red box. Simultaneously, people starting in the red box move towards the green box (right). Like the previous experiment, rectangles are created which serve as the bounds of spawning and goals. This experiment's purpose is to observe emergent properties, such as the expected vortex behaviour. This experiment uses approximately 2 GB of device memory to run.

## 6.2 Performance Experiments

In order to understand the performance of the GPU implementation various computational timing experiments are performed. The experiments are similar to those performed on the CPU version in section 3.5. The experiments test how altering various simulation parameters alters the performance of the model. These experiments follow the same process as the performance experiments of the unoptimised CPU implementation of section 3.5. The implementation of the GPU accelerated OCBM approach is described in the previous chapter 5. The three experiments examine the effect on performance when altering: (i) the population size, (ii) the density, and (iii) the individual routines. Each experiment has been run multiple times to get average values. At least 20 runs per data point are performed and the uncertainty of the average is less than 5%. As with the CPU equivalent of the experiments, each experiment has its own configuration of the model. The details of each are provided in the relevant experiment details. The experiments are performed on an NVidia GTX 970 using CUDA 9.1, with an I7-4790K Intel CPU with 24 GB of RAM.

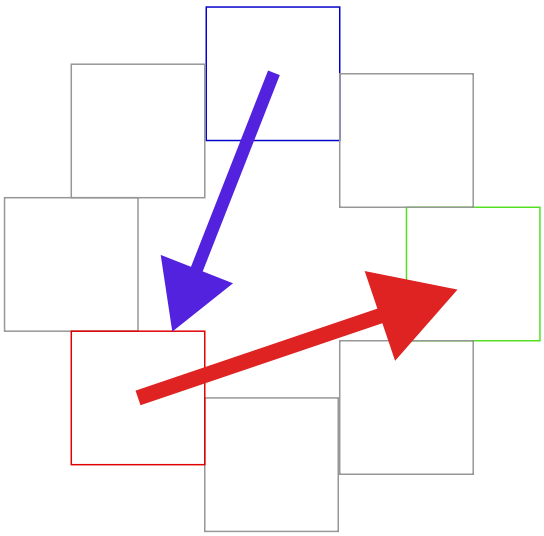


FIGURE 6.2: Set up of the eight-way experiment. The eight boxes represent different crowd starting and ending locations. Three particular boxes are coloured differently to highlight where the crowds of those boxes move.

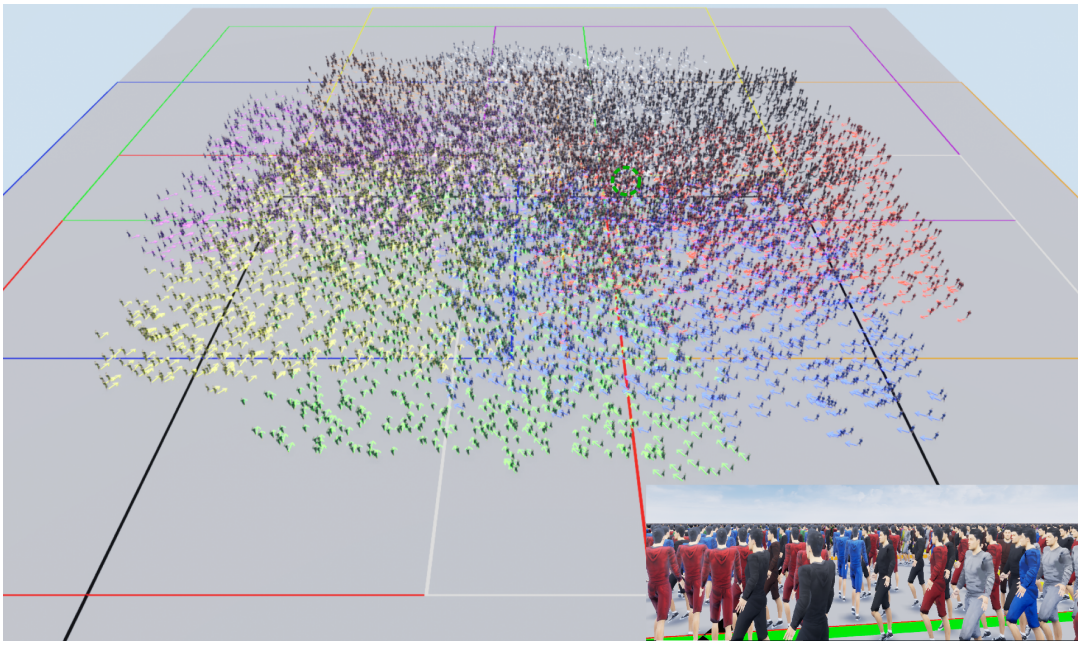


FIGURE 6.3: Visualisation of 10,000 people in Unreal. Eight crowds attempt to navigate through each other to reach their goal at the opposite end of the environment to where they started. Different colours are used for each crowd. Top: scene view from above; One pedestrian is highlighted with a green circle; Inset: view from the perspective of the pedestrian in the green circle

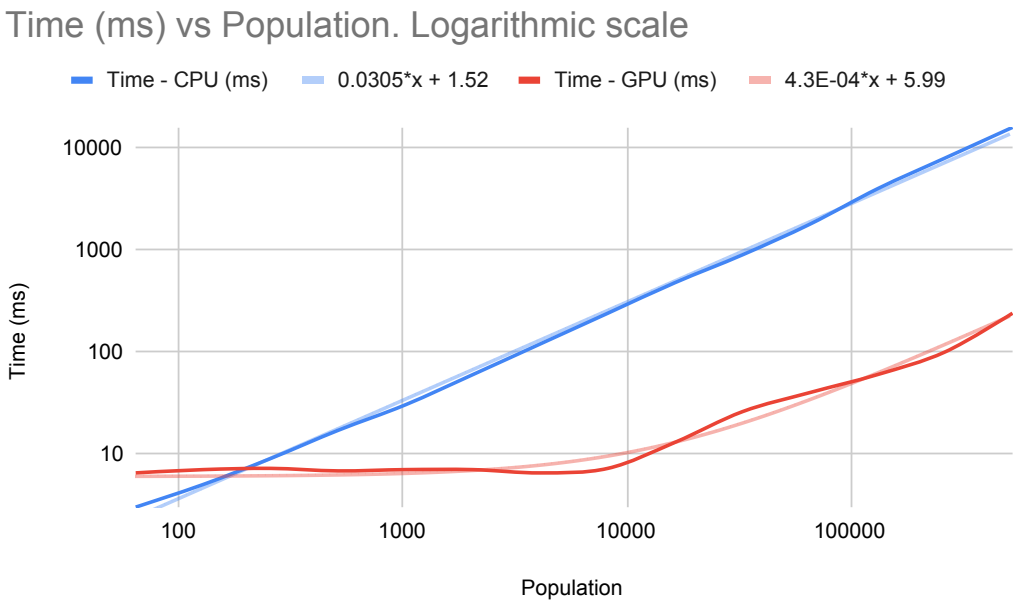



FIGURE 6.4: Time for one simulation iteration for varying population size for CPU and GPU implementation. This excludes rendering.

### Scaling by Population Size

This first experiment tests the effect that population size has on the performance. This experiment is set up exactly like the previous iteration of this experiment from section 3.5.1. This involves increasing the scenario size proportionally to the number of people, and setting up goal locations to keep the density between the scenarios as consistent as possible, around 1 person/$m^2$. No additional behaviours are considered.

The results of the CPU experiment are shown for comparison. The results are shown in figure 6.4. It shows the performance over population for both the GPU version and the previous CPU implementation. Populations of $2^n$ are chosen to fully utilise GPU blocks. A linear fit of the CPU and GPU results are overlaid on the figure. The linear fit of the CPU performance follows a linear trend of $30\mu s$/person. The GPU shows a trend of $0.01\mu s$/person. Each data point is the mean of 1000 seeded runs and the standard deviations are all within 4% of the corresponding mean time.

The GPU simulations runs at close to 30 frames a second (33ms per frame) for up to $5 \times 10^5$ simulated people. The CPU version performs better for smaller number of people, with a crossover occurring at approximately 300 people in the simulation. This is due to the GPU device not being fully utilised for smaller simulations and the reduced throughput being outperformed by the CPU. On the GPU used for experimentation, there was a limit on the amount of usable memory of 4 GB, which corresponds to approximately $5 \times 10^5$ people. Beyond this point additional waves of computation are serialised on the device. It is expected that relative performance increases will continue to be obtained for larger numbers of people for the GPU implementation for GPUs with larger memory capacity. For the values tested, the greatest performance difference is for the largest population size of $5^5$, where the time to simulate one frame on the CPU is approximately 16 seconds compared to $240ms$ on the GPU - a performance increase of 66 times. The CPU implementation

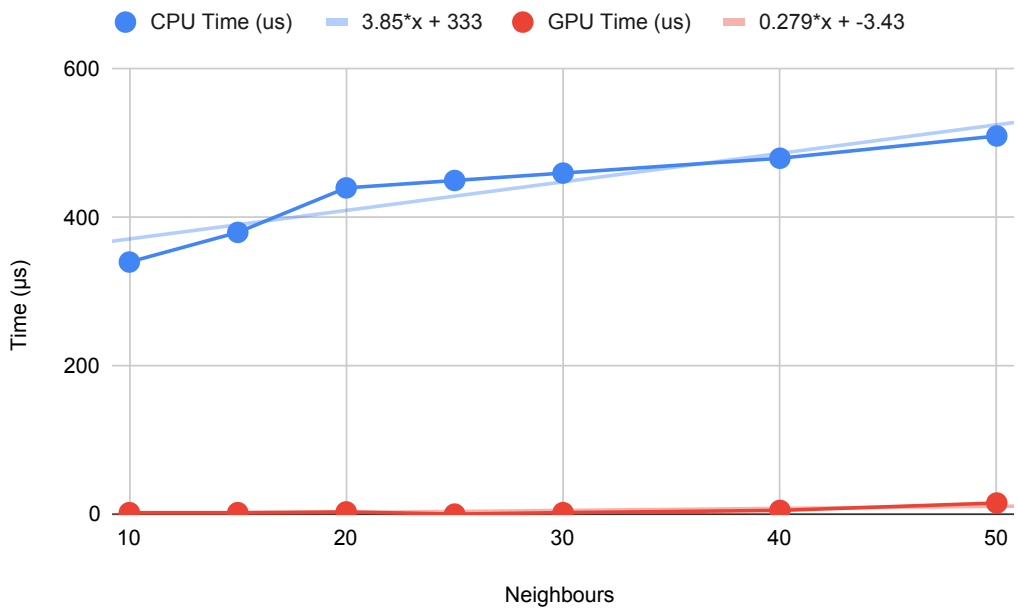



FIGURE 6.5: Performance vs The interaction range/number of constraints per person. Also shows the previous CPU implementation. The linear best fit for both implementations is shown in the lighter corresponding colour.

scales at around 30 microseconds per person, while the GPU scales at 0.01 microseconds per person. This creates a theoretical performance benefit of 3000 times for the GPU at large populations. However, these sizes are not achievable with currently available hardware.

### Examining Interaction Density

For the CPU OCBM implementation, the number of constraints per person was previously shown to have an effect on the performance. The findings showed that each constraint added around 0.04ms per person to the computational time (section 3.5.1 and figure 3.20). This experiment examines how the performance of GPU OCBM is affected by changing the number of constraints. This was done similarly to the CPU equivalent. A population size of around 10,000 people is used to ensure the device is fully utilised. For simplicity this experiment uses the results from the earlier experiment, which showed that the calculation of constraints is equivalent regardless of what caused them, i.e. the time taken to compute a constraint due to steering is the same as the constraint due to luggage. This allows the experiment to be simplified to consider only the computational comparison of constraint numbers in the CPU implementation compared to the GPU implementation. Figure 6.5 shows the results of this experiment, overlaid with the results of the previous finding for the CPU implementation. Each data point is the mean of 1000 seeded runs, and the standard deviation is less than 5% of the corresponding mean.

The figure shows the difference in scaling between the CPU and GPU implementations of OCBM. The GPU version is able to better scale with more constraints, at approximately $0.28\mu s$ compared to $3.9\mu s$ on the CPU. It achieves this because of the efficient linear program solver, which is able to handle larger amounts of work by distributing it across the device. It has a nonlinear relationship between the time

taken and the number of constraints. This is due to overhead in the linear program solver. This overhead is approximately constant for any problem size, and is more noticeable at smaller numbers of constraints, where the time taken to solve the constraints is on the same order of magnitude as the overhead.

#### Examining Model Complexity

This experiment concerns the time taken for the various subroutines of the simulation. It is the equivalent experiment to section 3.5.1 for the GPU equivalent. The experiment compares three different densities of low (0.2) medium (1) and high (2 people/$m^2$). A GPU performance profiler, NVProf, is used to obtain the performance of routines of the simulation. NVProf provides a performance overview of the GPU routines and memory copies in the application. A population size of 10,000 people is used to ensure full device utilisation.

The results in figure 6.6 show different routine performances compared to the CPU version, which is repeated in figure 6.7, and first presented in figure 3.21. The most notable change to the performance between the two implementations is the linear program routines, lp2 and lp3. While lp3 is still costly, the combined timings of the routines involved in constraint construction are larger when accounted together. These routines are "constraint", "groups", "bag", and "autonomous" Which at high density account for 10.6 microseconds compared to the lp3 routine which takes 4 microseconds. The reason the constraint construction routines are the most costly is due to the communication between people on the GPU. Communication between people in FLAMEGPU functions by recording information into a partition boundary matrix. This matrix holds important information which determines which agents are located within the spatially partitioned areas making up the simulation environment. However, even with the optimisations of this approach, it is the most expensive in terms of relative simulation time. Further improvements to the GPU implementation of OCBM would need to consider how to reduce the communication required between people.

## 6.3 Discussion

This chapter has explored the performance of the constraint-based framework on the GPU. It has demonstrated this visually through two different scenarios, as well as through examining the performance profile of the GPU implementation. The visualisation experiments have shown the framework simulating large crowds of people with complex behaviours. Within the cross walk examples used for visual analysis the OCBM GPU implementation was able to simulate around 40,000 people in real-time.

Using a benchmark model designed to parametrically explore both scale and density, the OCBM GPU implementation is shown to be more performant than the previous CPU implementation, with results indicating that OCBM GPU is able to simulate 66 times faster than the CPU implementation for large population sizes. The GPU implementation is less performant than the CPU equivalent for small populations of less than 300 people. This is due to not making full use of the GPU hardware at these smaller scales. However, these small populations run much faster than real time regardless of the implementation used. The performance impact when increasing the number of constraints scales more favourably in the GPU OCBM. This is due to using the optimised linear program solver, RGB, of the previous chapter.

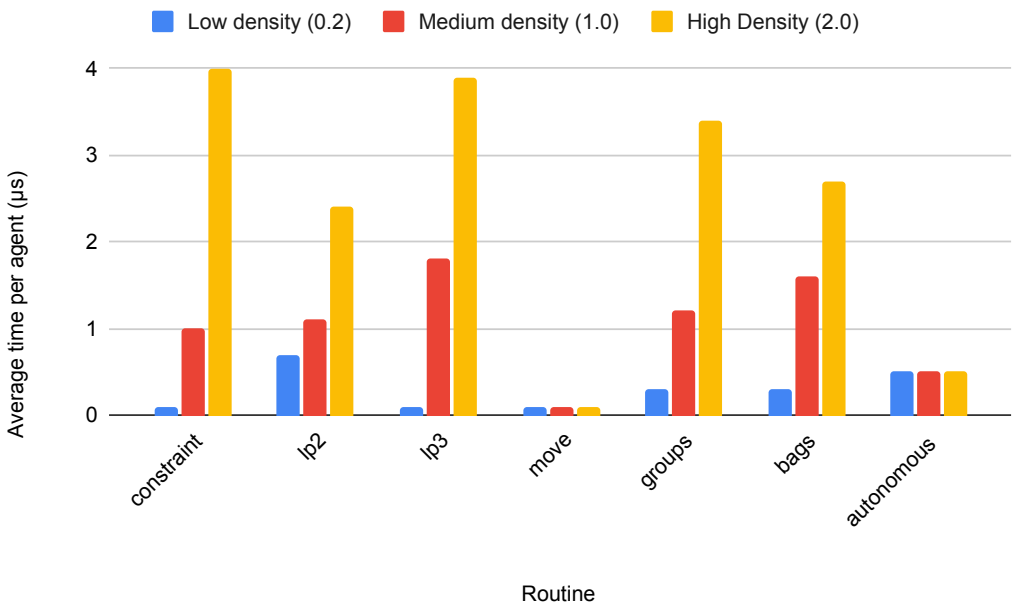



FIGURE 6.6: Timing of various sub-routines of OCBM (routine vs time taken) for the GPU. Taken from a single simulation run. Multiple densities are compared

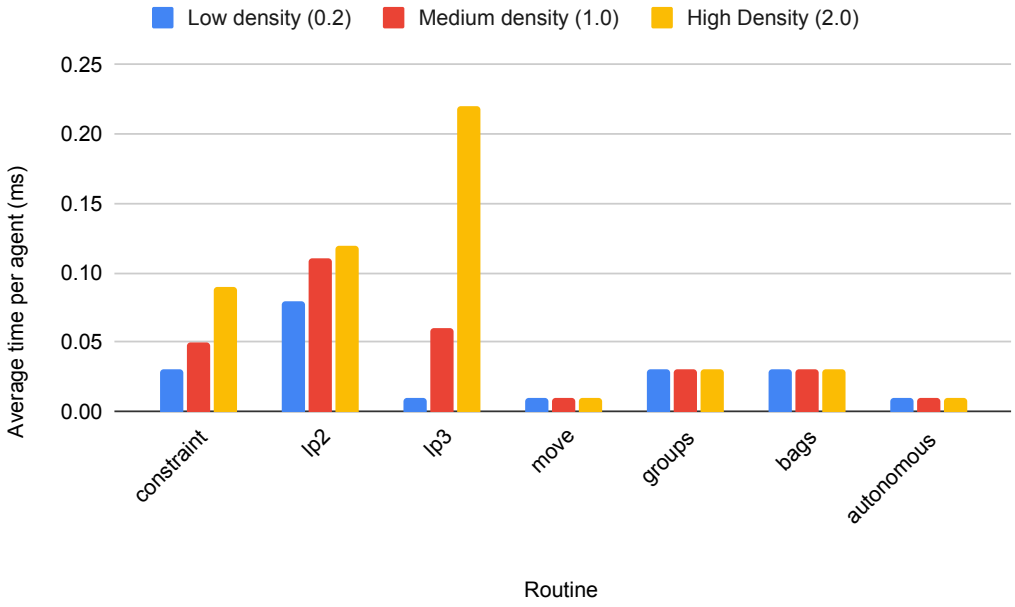



FIGURE 6.7: Timing of various sub-routines of OCBM (routine vs time taken) for the original CPU implementation. Multiple densities are compared

GPU hardware is a rapidly evolving field. Each generation of graphics cards brings about increased performances and larger device memory. The increasing capability of GPUs means that the performance of OCBM GPU increases with newer hardware. Newer GPUs can simulate larger population sizes and calculate more OCBM constraints while running in real time. Further advances of GPU hardware and memory should increase the performance differences between CPU and GPU implementation.

Due to the optimised linear solving, the largest performance bottleneck is observed to be the communication between people. This communication is required so people can understand their surroundings. It is a foundational part of the model. Work is being carried out to create better communication algorithms on GPU architecture [41]. Reducing the communication would require careful overhauling of the basis of the model, and may drastically affect the numerical results. By reducing the rate at which people communicate, this equates to reducing the rate at which people "observe" the environment and note people and their neighbours. A reduction in communication could potentially be achieved by using predictive logic. Each person could track uncertainty of neighbours. This uncertainty increases as communication is skipped, and represents the uncertainty in the possible position of their neighbours. This sort of approach is under development within the field of autonomous cars [132].

# Chapter 7

# Conclusion

Pedestrian simulations are important tools in studying and predicting pedestrian crowds and motion. Their usage has led to more efficient transport hubs, larger events, and safer environments. A key limitation of existing real time approaches is the difficulty in maintaining accurate functionality of models when incorporating additional behaviours.

This thesis has presented a microscopic constraint-based pedestrian model, Optimal Constraint Behaviour Model (OCBM), as an approach to simulate a wide variety of behaviours. These behaviours were found to better match experimental data compared to other tested approaches under visual inspection. This has been achieved by proposing the use of constraint hardness, which emphasises the importance of behaviours on a per-individual basis. OCBM, with constraint hardness, was applied to simulate various behaviours: luggage, groups, autonomous vehicles, and a platform-train interface. How closely the model is able to simulate emergent behaviours was validated against observations and found to perform accurately against simple controlled environments, such as bi-flow corridors. The performance of OCBM was improved by implementing it on the GPU. A performant novel linear program solver on the GPU, RGB, was created. This solver helped improve the performance of GPU OCBM to run up to 30 times faster than the CPU implementation.

OCBM is targeted to address the gap in the research literature of pedestrian simulation models between offline high-accuracy models and real-time inaccurate models. By accelerating performance using GPU hardware, OCBM is able to simulate over 40,000 people in real-time on modest consumer grade hardware. Within the model, pedestrians possess the capability to perform intricate motions, accounting for various behaviours simultaneously. This includes tasks like carrying luggage while travelling in a social group.

Many types of pedestrian models have difficulty in being extended with additional behaviours, especially steering-related behaviours [59]. This limits the range of applications for a model without developing special rules for each steering behaviour. Current models that mitigate this limitation are either specialised and designed for multiple particular behaviour, or use a rules-based approach like cellular automata. The first research question of the thesis (RQ 1) explored and compared pedestrian models in the context of modelling large dense crowds to find the most suitable approach. It explored different models, including microscopic and macroscopic and found limitations in available models for large dense crowds of people

with short-term behaviours.

The next research question (RQ 2) addresses a potential solution to the limitations of current pedestrian models. A constraint-based microscopic simulation model was deemed to be the best candidate for providing a flexible modelling approach suitable to capturing a wide range of steering-level behaviours accurately. This is justified as the use of constraints permits pedestrian velocity to be calculated while simultaneously considering behaviours and steering. The RVO family of steering models [14, 10], especially ORCA [16], were noted as being a suitable balance between complexity and performance to create a complex real time steering model. This resulted in the development of OCBM, a modelling approach described in Chapter 3. which allows specification of pedestrian behaviours.

OCBM functions by calculating behaviours and steering as a linear program. Behaviours and steering limit the range of possible velocities a person can use. The resulting velocity is one which satisfies all the requirements of steering and behaviours. For the case where a feasible solution cannot be found to the linear program, the novel concept of hardness of constraints was constructed. When a linear program is infeasible, this means that there is no velocity a person can take that does not violate some behaviour or functionality. The solution is to adjust the linear program constraints until a velocity can be found. The amount the constraints shift is determined by this new property, their hardness. A hard constraint means that the steering/behaviour that created it should be adhered to. On the contrary, a soft constraint is less important for the person to adhere as stringently to. The softer a constraint, the more likely a person's velocity is to violate the constraint. This constraint hardness method has the capacity for producing visually incorrect results. This is already a weakness of approaches like ORCA, but incorrect motion can be minimised by balancing the hardness of constraints correctly. The use of hardness of constraints can be reasoned from psychology, as discussed in section 3.2.2 which introduced this idea. Some people will be much more willing to separate from their social group if that means they can avoid an autonomous vehicle. The hardness of constraints can be set on a per-person scale, providing complex variations between people in the simulation.

The breadth of behaviour complexity of OCBM, and hence the question of if the approach is flexible, was demonstrated by implementing behaviours in the model: luggage, social groups, autonomous vehicles, and the platform-train interface. Chapter 3 explored how the model behaves when simulating the first three of these behaviours. It was shown that OCBM is able to demonstrate a transition of social groups from a 'v' shape at low density to a column formation at higher density. Ren et al. [170] is a collision avoidance model for groups based on ORCA. Compared to OCBM, it is focused on group behaviours. OCBM performs less complex computation, but by considering groups in terms of constraints, OCBM is able to demonstrate a wider variety of behaviours. It also showed a higher average flow rate for people with luggage compared to simulating people and luggage as singular people. These results show OCBM demonstrating a wide variety of different emergent behaviours. The model is able to simulate a greater variety of constraints than those implemented in the thesis. The prerequisite for implementation in OCBM is for the behaviour to be describable in terms of constraints to a person's velocity. Such example behaviours could be modelling the special effects that occur at incredibly high densities [158], or the way people move when crossing a road with cars [108]. The process of modelling using constraints has been shown to be flexible and logical through the demonstration of the range of behaviours implemented in chapter 3. The OCBM approach makes a novel contribution to the state of the art

by providing a modelling approach that goes beyond simple collision response [38]. Likewise the framework's extensibility means that it is not constrained to represent any one specific behaviour [39].

The accuracy of a model determines how well it can reproduce real, observable behaviours. In the context of microscopic pedestrian simulations, the comparison of emergent properties of crowds between observation and computer simulation is one way to validate the accuracy. An important descriptor of an emergent crowd property is the fundamental diagram - a relationship between average speed of a person against their local density. Fundamental diagrams were used within chapter 4 to evaluate model accuracy.

To understand if the accuracy of interactions between individuals could be maintained when adding complex behaviours (RQ 3), OCBM was tested by applying the model to specific behaviours, namely luggage and the platform-train interface (PTI). The results of this experimentation showed that the OCBM approach of simulating people and luggage as separate entities was better able to match observational data. Compared to an alternative of simulating singular larger agents, OCBM demonstrated the importance of accounting for the difference in cross section between the front and side of people. This is because the front cross-section size of a person is their shoulder width, whereas from the side the cross-section size is their body plus the trailing luggage. This asymmetry is accounted for in OCBM, and permits more accurate behaviour.

Pedestrian simulation models tend to have difficulty in simulating crowds at high density [196]. Special types of models are developed to handle these cases [195]. OCBM was shown to be typical of existing models in this regard. This can be expected as this is also a shortcoming of the underlying ORCA approach [149]. Observation showed that people tend to a movement speed of 0.5m/s. OCBM, in contrast, found that people tended towards a stationary movement of 0m/s. A future research direction could be to explore a method of describing high density motion as constraints which can be incorporated within OCBM to improve its accuracy at high density. Overall, OCBM is able to more accurately predict a wide variety of scenarios which need to account for a variety of different behaviours, compared to other test models, under visual inspection of the fundamental diagrams. In contrast to other state of the art models, OCBM demonstrates flexibility and a wide range of scenarios it is suitable for.

To improve the number of people simulated in real time, the performance of OCBM was examined. It was found through quantitative experimentation that the linear program solver was the most significant factor on performance. To improve the performance of the linear program solver, RGB, a novel linear program solver for the GPU, was developed. This approach aimed to answer research question RQ 4, whether this constraint based approach is suitable for GPU acceleration. Chapter 5 explored the suitability of GPUs in solving multiple linear programs. In particular, it specialised in solving linear programs in two dimensions in batch amounts, where the sizes of the programs within the batch can be of different sizes. RGB functions by calculating the smallest quantity of work needed during the linear program solving stage, referred to as a work unit (WU). WUs are distributed across GPU threads to maximise compute parallelisation. This approach ensures that idle threads in a warp are minimised.

The performance of RGB was tested against state-of-the-art linear program solvers in section 5.2. It was found that RGB can perform 22 times faster than other solvers. This included CPU-GPU memory transfer, a significant cost in offloading compute to GPU devices. Within the context of constraint based pedestrian simulation, the

RGB approach can maximise performance by avoiding host to device data transfer by generating constraints on the device. For this reason, the whole OCBM approach was implemented on the GPU using a GPU agent based simulation framework, leveraging RGB for the constraint solving aspect of the simulation.

By using RGB as well as implementing the whole OCBM framework on the GPU, the code performance was improved. Chapter 6 was then able to explore if OCBM could be accelerated to meet the target of real-time performance (RQ 5). Compared to the CPU implementation, the GPU OCBM had performance increases of up to 66 times. The most noticeable performance gains are found for larger population sizes. For very small populations the GPU device is unable to be fully utilised, and it underperforms against the CPU implementation. Despite this, even underutilised the GPU can outperform the CPU counterpart. The crossover point for the modest computational configuration used within this thesis is at around 1000 people. Above this amount, the GPU version is faster.

GPU OCBM was able to simulate 40,000 people in real-time. This included accounting for complex behaviours, such as for luggage and groups. The performance experiments were limited in population to 524288, or $2^{19}$, because of memory constraints. Future hardware with larger memory will be able to simulate larger crowds.

The flexibility of the OCBM has been explored and tested through behaviours. Beyond crowd individual behaviours, crowd phenomena, when described by constraints, could be a part of OCBM. There are numerous phenomena that occur in specific scenarios which could be further explored. An example is in queues of people. This includes stop-start motion, and the propagation of speed waves. Future work could be performed to find a way of describing the functionality of queues in terms of constraints, and combining them with the OCBM behaviours already implemented.

While validation has been performed on OCBM in certain scenarios, there is a wide range of further scenarios to explore and validate. Further validation of the model extends its use, allowing OCBM to predict and model accurate crowd behaviour in a wider variety of environments and behaviours. Validating OCBM requires further observational experiments of crowds. In addition, it requires observational experiments which incorporate multiple behaviours together. There are difficulties in performing experiments that single out particular, fundamental behaviour, because of how complex each individual's motion is. Another target for validation of OCBM is in the hardness of constraints. Further work to parameterise the constraint hardness could be done to see if OCBM is capable of reflecting behaviours accurately, and how constraint hardness changes for different scenarios.

The field of GPU research is rapidly developing. New more powerful devices are being released, and GPU research is very popular thanks to interest in AI and deep learning. It can be difficult to create cutting-edge research that stays relevant when the field is evolving so rapidly. However, this also means there are new and optimised algorithms being developed. With improved algorithms that pertain to agent based simulations, OCBM will be able to improve its performance by utilising them. In addition, the performance of OCBM should only improve over time, when compared to CPUs, as GPU devices scale to greater computational power.

# Bibliography


[1] Y. Abe and M. Yoshiki. "Collision avoidance method for multiple autonomous mobile agents by implicit cooperation". In: *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*. Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180). Vol. 3. Oct. 2001, 1207–1212 vol.3. DOI: 10.1109/IROS.2001.977147.

[2] Juliane Adrian et al. "A Glossary for Research on Human Crowd Dynamics". In: *Collective Dynamics* 4 (Mar. 13, 2019). DOI: 10.17815/CD.2019.19.

[3] Alexandre Alahi et al. "Social LSTM: Human Trajectory Prediction in Crowded Spaces". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA: IEEE, June 2016, pp. 961–971. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.110. URL: http://ieeexplore.ieee.org/document/7780479/ (visited on 05/17/2023).

[4] J. Alonso-Mora et al. "Reciprocal collision avoidance for multiple car-like robots". In: *2012 IEEE International Conference on Robotics and Automation*. 2012 IEEE International Conference on Robotics and Automation. May 2012, pp. 360–366. DOI: 10.1109/ICRA.2012.6225166.

[5] Javier Alonso-Mora et al. "Optimal Reciprocal Collision Avoidance for Multiple Non-Holonomic Robots". In: *Distributed Autonomous Robotic Systems*. Ed. by Alcherio Martinoli et al. Springer Tracts in Advanced Robotics 83. Springer Berlin Heidelberg, 2013, pp. 203–216. ISBN: 978-3-642-32722-3. DOI: 10.1007/978-3-642-32723-0_15. URL: http://link.springer.com/chapter/10.1007/978-3-642-32723-0_15 (visited on 04/27/2017).

[6] Matthias Althoff and Alexander Mergel. "Comparison of Markov chain abstraction and Monte Carlo simulation for the safety assessment of autonomous cars". In: *IEEE Transactions on Intelligent Transportation Systems* 12.4 (2011), pp. 1237–1247.

[7] Alessandro Antonucci et al. "Generating Reliable and Efficient Predictions of Human Motion: A Promising Encounter between Physics and Neural Networks". In: *IEEE Access* 10 (2022), pp. 144–157. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3138614. arXiv: 2006.08429[cs]. URL: http://arxiv.org/abs/2006.08429 (visited on 09/30/2022).

[8] Stefania Bandini et al. "Towards a Methodology for Situated Cellular Agent Based Crowd Simulations". In: *Engineering Societies in the Agents World VI*. Ed. by Oğuz Dikenelli, Marie-Pierre Gleizes, and Alessandro Ricci. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2006, pp. 203–220. ISBN: 978-3-540-34452-0.

[9] D. Bareiss and J. van den Berg. "Reciprocal collision avoidance for robots with linear dynamics using LQR-Obstacles". In: *2013 IEEE International Conference on Robotics and Automation*. 2013 IEEE International Conference on Robotics and Automation. May 2013, pp. 3847–3853. DOI: 10.1109/ICRA.2013.6631118.

[10] Daman Bareiss and Jur van den Berg. "Generalized reciprocal collision avoidance". In: *The International Journal of Robotics Research* 34.12 (Oct. 2015), pp. 1501–1514. ISSN: 0278-3649, 1741-3176. DOI: 10.1177/0278364915576234. URL: http://journals.sagepub.com/doi/10.1177/0278364915576234 (visited on 04/27/2017).

[11] J Bélanger and P Venne. "The What, Where and Why of Real-Time Simulation". In: (2010).

[12] Nicola Bellomo and Christian Dogbe. "On the Modeling of Traffic and Crowds: A Survey of Models, Speculations, and Perspectives". In: *SIAM Review* 53.3 (Jan. 2011). Publisher: Society for Industrial and Applied Mathematics, pp. 409–463. ISSN: 0036-1445. DOI: 10.1137/090746677. URL: https://epubs.siam.org/doi/abs/10.1137/090746677 (visited on 07/31/2023).

[13] Bently. *LEGION Simulator: Simulation & Modeling Software | Bentley*. URL: https://www.bentley.com/software/legion-simulator/ (visited on 05/22/2023).

[14] J. van den Berg, Ming Lin, and D. Manocha. "Reciprocal Velocity Obstacles for real-time multi-agent navigation". In: *IEEE International Conference on Robotics and Automation, 2008. ICRA 2008*. IEEE International Conference on Robotics and Automation, 2008. ICRA 2008. May 2008, pp. 1928–1935. DOI: 10.1109/ROBOT.2008.4543489.

[15] J. van den Berg et al. "Reciprocal collision avoidance with acceleration-velocity obstacles". In: *2011 IEEE International Conference on Robotics and Automation*. 2011 IEEE International Conference on Robotics and Automation. May 2011, pp. 3475–3482. DOI: 10.1109/ICRA.2011.5980408.

[16] Jur van den Berg et al. "Reciprocal n-Body Collision Avoidance". In: *Robotics Research*. Springer, Berlin, Heidelberg, 2011, pp. 3–19. DOI: 10.1007/978-3-642-19457-3_1. URL: https://link.springer.com/chapter/10.1007/978-3-642-19457-3_1 (visited on 04/27/2017).

[17] Mark de Berg et al. *Computational Geometry*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. ISBN: 978-3-540-77973-5. DOI: 10.1007/978-3-540-77974-2. URL: http://link.springer.com/10.1007/978-3-540-77974-2 (visited on 02/05/2018).

[18] Delphine Bernardin et al. "Gaze anticipation during human locomotion". In: *Experimental Brain Research* 223.1 (Nov. 2012), pp. 65–78. ISSN: 0014-4819, 1432-1106. DOI: 10.1007/s00221-012-3241-2. URL: http://link.springer.com/10.1007/s00221-012-3241-2 (visited on 10/14/2016).

[19] J. L. Berrou et al. “Calibration and validation of the Legion simulation model using empirical data”. In: *Pedestrian and Evacuation Dynamics 2005*. Ed. by Nathalie Waldau et al. Berlin, Heidelberg: Springer, 2007, pp. 167–181. ISBN: 978-3-540-47064-9. DOI: 10.1007/978-3-540-47064-9_15.

[20] Glen Berseth, Mubbasir Kapadia, and Petros Faloutsos. “Robust space-time footsteps for agent-based steering”. In: *Computer Animation and Virtual Worlds* (2015). URL: http://www.cs.rutgers.edu/~mk1353/pdfs/2015-casa-robust-footsteps.pdf (visited on 11/08/2016).

[21] Glen Berseth, Mubbasir Kapadia, and Petros Faloutsos. “SteerPlex: Estimating Scenario Complexity for Simulated Crowds”. In: *Proceedings of Motion on Games*. MIG ’13. New York, NY, USA: ACM, 2013, 45:67–45:76. ISBN: 978-1-4503-2546-2. DOI: 10.1145/2522628.2522650. URL: http://doi.acm.org/10.1145/2522628.2522650 (visited on 06/15/2018).

[22] Glen Berseth et al. “SteerFit: Automated Parameter Fitting for Steering Algorithms”. In: *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’14. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2014, pp. 113–122. URL: http://dl.acm.org/citation.cfm?id=2849517.2849536 (visited on 06/15/2018).

[23] Andrew Best, Sahil Narang, and Dinesh Manocha. “Real-time reciprocal collision avoidance with elliptical agents”. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. 2016 IEEE International Conference on Robotics and Automation (ICRA). May 2016, pp. 298–305. DOI: 10.1109/ICRA.2016.7487148.

[24] J. Bieling, P. Peschlow, and P. Martini. “An efficient GPU implementation of the revised simplex method”. In: *2010 IEEE International Symposium on Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW)*. 2010 IEEE International Symposium on Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW). Apr. 2010, pp. 1–8. DOI: 10.1109/IPDPSW.2010.5470831.

[25] Stephen Bitgood. “An Analysis of Visitor Circulation: Movement Patterns and the General Value Principle”. In: *Curator: The Museum Journal* 49.4 (Oct. 2006), pp. 463–475. ISSN: 00113069, 21516952. DOI: 10.1111/j.2151-6952.2006.tb00237.x. URL: https://onlinelibrary.wiley.com/doi/10.1111/j.2151-6952.2006.tb00237.x (visited on 09/29/2022).

[26] Avi Bleiweiss. “Multi Agent Navigation on the GPU”. In: (), p. 10.

[27] Victor Blue and Jeffrey Adler. “Cellular Automata Microsimulation of Bidirectional Pedestrian Flows”. In: *Transportation Research Record: Journal of the Transportation Research Board* 1678 (Jan. 1, 1999), pp. 135–141. ISSN: 0361-1981. DOI: 10.3141/1678-17. URL: https://trrjournalonline.trb.org/doi/10.3141/1678-17 (visited on 02/05/2019).

[28] Victor Blue and Jeffrey Adler. “Emergent Fundamental Pedestrian Flows From Cellular Automata Microsimulation | Request PDF”. In: *Transportation Research Record: Journal of the Transportation Research Board* 1644 (1998), pp. 29–36. DOI: http://dx.doi.org/10.3141/1644-04. URL: https://www.researchgate.net/publication/245558562_Emergent_Fundamental_Pedestrian_Flows_From_Cellular_Automata_Microsimulation (visited on 02/05/2019).

[29] Victor J. Blue and Jeffrey L. Adler. "Cellular automata microsimulation for modeling bi-directional pedestrian walkways". In: *Transportation Research Part B: Methodological* 35.3 (Mar. 1, 2001), pp. 293–312. ISSN: 0191-2615. DOI: 10.1016/S0191-2615(99)00052-1. URL: https://www.sciencedirect.com/science/article/pii/S0191261599000521 (visited on 11/01/2021).

[30] Patrick M Boesch and Francesco Ciari. "Agent-based simulation of autonomous cars". In: *2015 American Control Conference (ACC)*. IEEE, 2015, pp. 2588–2592.

[31] Jean-François Bonnefon, Azim Shariff, and Iyad Rahwan. "The social dilemma of autonomous vehicles". In: *Science* 352.6293 (2016), pp. 1573–1576.

[32] Ernst Bosina and Ulrich Weidmann. "Estimating pedestrian speed using aggregated literature data". In: *Physica A: Statistical Mechanics and its Applications* 468 (Feb. 2017), pp. 1–29. ISSN: 03784371. DOI: 10.1016/j.physa.2016.09.044. URL: https://linkinghub.elsevier.com/retrieve/pii/S0378437116306604 (visited on 07/15/2019).

[33] Martin Brunnhuber et al. "Bridging the gap between visual exploration and agent-based pedestrian simulation in a virtual environment". In: *Proceedings of the 18th ACM symposium on Virtual reality software and technology*. VRST '12. New York, NY, USA: Association for Computing Machinery, Dec. 10, 2012, pp. 9–16. ISBN: 978-1-4503-1469-5. DOI: 10.1145/2407336.2407339. URL: https://doi.org/10.1145/2407336.2407339 (visited on 05/22/2023).

[34] Transport Systems Catapult. *Driverless Pods*. Catapult. URL: https://ts.catapult.org.uk/innovation-centre/cav/cav-projects-at-the-tsc/self-driving-pods/ (visited on 10/16/2019).

[35] Rohan Chandra et al. "DensePeds: Pedestrian Tracking in Dense Crowds Using Front-RVO and Sparse Features". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). ISSN: 2153-0866. Nov. 2019, pp. 468–475. DOI: 10.1109/IROS40897.2019.8968470.

[36] Panayiotis Charalambous et al. "A Data-Driven Framework for Visual Crowd Analysis". In: *Computer Graphics Forum* 33.7 (2014), pp. 41–50. ISSN: 1467-8659. DOI: 10.1111/cgf.12472. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12472 (visited on 06/15/2018).

[37] John Charlton, Steve Maddock, and Paul Richmond. "Two-dimensional batch linear programming on the GPU". In: *Journal of Parallel and Distributed Computing* 126 (Apr. 1, 2019), pp. 152–160. ISSN: 0743-7315. DOI: 10.1016/j.jpdc.2019.01.001. URL: http://www.sciencedirect.com/science/article/pii/S074373151830399X (visited on 02/01/2019).

[38] John Charlton et al. "Fast Simulation of Crowd Collision Avoidance". In: *Computer Graphics International Conference*. Springer, 2019, pp. 266–277.

[39] John Charlton et al. "Simulating Crowds and Autonomous Vehicles". In: *Transactions on Computational Science XXXVII: Special Issue on Computer Graphics*. Ed. by Marina L. Gavrilova et al. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2020, pp. 129–143. ISBN: 978-3-662-61983-4. DOI: 10.1007/978-3-662-61983-4_8. URL: https://doi.org/10.1007/978-3-662-61983-4_8 (visited on 05/19/2023).

[40] Xu Chen et al. "Social force models for pedestrian traffic – state of the art". In: *Transport Reviews* 38.5 (Sept. 3, 2018). Publisher: Routledge _eprint: https://doi.org/10.1080/014416 pp. 625–653. ISSN: 0144-1647. DOI: 10.1080/01441647.2017.1396265. URL: https://doi.org/10.1080/01441647.2017.1396265 (visited on 10/03/2022).

[41] Robert Chisholm, Steve Maddock, and Paul Richmond. "Improved GPU near neighbours performance for multi-agent simulations". In: *Journal of Parallel and Distributed Computing* 137 (Mar. 1, 2020), pp. 53–64. ISSN: 0743-7315. DOI: 10.1016/j.jpdc.2019.11.002. URL: https://www.sciencedirect.com/science/article/pii/S0743731519301340 (visited on 05/11/2023).

[42] Hui-Qing Chong, Ah-Hwee Tan, and Gee-Wah Ng. "Integrated cognitive architectures: a survey". In: *Artificial Intelligence Review* 28.2 (Aug. 1, 2007), pp. 103–130. ISSN: 1573-7462. DOI: 10.1007/s10462-009-9094-9. URL: https://doi.org/10.1007/s10462-009-9094-9 (visited on 05/22/2023).

[43] Mohcine Chraibi et al. "Modelling of Pedestrian and Evacuation Dynamics". In: *Encyclopedia of Complexity and Systems Science*. Ed. by Robert A. Meyers. Berlin, Heidelberg: Springer Berlin Heidelberg, 2018, pp. 1–22. ISBN: 978-3-642-27737-5. DOI: 10.1007/978-3-642-27737-5_705-1. URL: http://link.springer.com/10.1007/978-3-642-27737-5_705-1 (visited on 10/03/2022).

[44] CLP COIN. *COIN-OR linear programming solver*. 2011.

[45] *Connected and Automated Vehicles: market forecast 2020*. GOV.UK. 2021. URL: https://www.gov.uk/government/publications/connected-and-automated-vehicles-market-forecast-2020 (visited on 05/23/2023).

[46] Marco Costa. "Interpersonal Distances in Group Walking". In: *Journal of Nonverbal Behavior* 34.1 (Mar. 1, 2010), pp. 15–26. ISSN: 1573-3653. DOI: 10.1007/s10919-009-0077-y. URL: https://doi.org/10.1007/s10919-009-0077-y (visited on 12/20/2022).

[47] CPLEX. "12.2 User's Manual". In: *Book 12.2 User's Manual, Series 12.2 User's Manual* (2010).

[48] Luca Crociani et al. "Investigating the effect of social groups in uni-directional pedestrian flow". In: (Oct. 31, 2017).

[49] Sean Curtis, Andrew Best, and Dinesh Manocha. "Menge: A Modular Framework for Simulating Crowd Movement". In: *Collective Dynamics* 1.0 (Mar. 15, 2016), pp. 1–40. ISSN: 2366-8539. URL: https://collective-dynamics.eu/index.php/cod/article/view/A1 (visited on 06/15/2018).

[50] Sean Curtis and Dinesh Manocha. "Pedestrian Simulation Using Geometric Reasoning in Velocity Space". In: *Pedestrian and Evacuation Dynamics 2012*. Ed. by Ulrich Weidmann, Uwe Kirsch, and Michael Schreckenberg. Cham: Springer International Publishing, 2014, pp. 875–890. ISBN: 978-3-319-02446-2. DOI: 10.1007/978-3-319-02447-9_73. URL: http://link.springer.com/10.1007/978-3-319-02447-9_73 (visited on 01/21/2019).

[51] Sean Curtis, Jamie Snape, and Dinesh Manocha. "Way portals: efficient multi-agent navigation with line-segment goals". In: *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. I3D '12: Symposium on Interactive 3D Graphics and Games. Costa Mesa California: ACM, Mar. 9, 2012, pp. 15–22. ISBN: 978-1-4503-1194-6. DOI: 10.1145/2159616.2159619. URL: https://dl.acm.org/doi/10.1145/2159616.2159619 (visited on 05/17/2023).

[52] Sean Curtis et al. "Virtual Tawaf: A case study in simulating the behavior of dense, heterogeneous crowds". In: *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops). Barcelona, Spain: IEEE, Nov. 2011, pp. 128–135. ISBN: 978-1-4673-0063-6. DOI: 10.1109/ICCVW.2011.6130234. URL: http://ieeexplore.ieee.org/document/6130234/ (visited on 05/17/2023).

[53] James E. Cutting, Peter M. Vishton, and Paul A. Braren. "How we avoid collisions with stationary and moving objects". In: *Psychological Review* 102.4 (1995), pp. 627–651. ISSN: 1939-1471 0033-295X. DOI: 10.1037/0033-295X.102.4.627.

[54] Dennis G. Davis and John P. Braaksma. "Adjusting for luggage-laden pedestrians in airport terminals". In: *Transportation Research Part A: General* 22.5 (Sept. 1, 1988), pp. 375–388. ISSN: 0191-2607. DOI: 10.1016/0191-2607(88)90014-3. URL: http://www.sciencedirect.com/science/article/pii/0191260788900143 (visited on 10/11/2017).

[55] Steven Deere et al. "An evacuation model validation data-set for high-rise construction sites". In: *Fire Safety Journal*. Fire Safety Science: Proceedings of the 13th International Symposium 120 (Mar. 1, 2021), p. 103118. ISSN: 0379-7112. DOI: 10.1016/j.firesaf.2020.103118. URL: https://www.sciencedirect.com/science/article/pii/S0379711219307192 (visited on 08/09/2023).

[56] Jur van Den Berg et al. "Centralized path planning for multiple robots: Optimal decoupling into sequential plans." In: *Robotics: Science and systems*. Vol. 2. 2009, pp. 2–3. URL: http://www.academia.edu/download/40318200/Centralized_path_planning_for_multiple_r20151123-8487-1j02trh.pdf (visited on 10/10/2017).

[57] Felix Dietrich and Gerta Köster. "Gradient navigation model for pedestrian dynamics". In: *Physical Review. E, Statistical, Nonlinear, and Soft Matter Physics* 89.6 (June 2014), p. 062801. ISSN: 1550-2376. DOI: 10.1103/PhysRevE.89.062801.

[58] Fangshu Dong et al. "An Experimental Study on the Influence of Carrying Luggage to Bidirectional Pedestrian Flows". In: *2019 9th International Conference on Fire Science and Fire Protection Engineering (ICFSFPE)*. 2019 9th International Conference on Fire Science and Fire Protection Engineering (ICFSFPE). Oct. 2019, pp. 1–6. DOI: 10.1109/ICFSFPE48751.2019.9055776.

[59] Dorine C. Duives, Winnie Daamen, and Serge P. Hoogendoorn. "State-of-the-art crowd motion simulation models". In: *Transportation Research Part C: Emerging Technologies* 37 (Dec. 1, 2013), pp. 193–209. ISSN: 0968-090X. DOI: 10.1016/j.trc.2013.02.005. URL: https://www.sciencedirect.com/science/article/pii/S0968090X13000351 (visited on 04/27/2023).

[60] Z Fang, S. M Lo, and J. A Lu. "On the relationship between crowd density and movement velocity". In: *Fire Safety Journal* 38.3 (Apr. 1, 2003), pp. 271–283. ISSN: 0379-7112. DOI: 10.1016/S0379-7112(02)00058-9. URL: https://www.sciencedirect.com/science/article/pii/S0379711202000589 (visited on 11/01/2022).

[61] Claudio Feliciani, Hisashi Murakami, and Katsuhiro Nishinari. "A universal function for capacity of bidirectional pedestrian streams: Filling the gaps in the literature". In: *PLOS ONE* 13 (Dec. 19, 2018), e0208496. DOI: 10.1371/journal.pone.0208496.

[62] Claudio Feliciani and Katsuhiro Nishinari. "An improved Cellular Automata model to simulate the behavior of high density crowd and validation by experimental data". In: *Physica A: Statistical Mechanics and its Applications* 451 (June 1, 2016), pp. 135–148. ISSN: 0378-4371. DOI: 10.1016/j.physa.2016.01.057. URL: https://www.sciencedirect.com/science/article/pii/S0378437116001047 (visited on 11/01/2021).

[63] Yan Feng et al. "Data collection methods for studying pedestrian behaviour: A systematic review". In: *Building and Environment* 187 (Jan. 1, 2021), p. 107329. ISSN: 0360-1323. DOI: 10.1016/j.buildenv.2020.107329. URL: https://www.sciencedirect.com/science/article/pii/S0360132320306983 (visited on 05/17/2023).

[64] Paolo Fiorini and Zvi Shiller. "Motion Planning in Dynamic Environments Using Velocity Obstacles". In: *The International Journal of Robotics Research* 17.7 (July 1, 1998), pp. 760–772. ISSN: 0278-3649. DOI: 10.1177/027836499801700706. URL: http://journals.sagepub.com/doi/abs/10.1177/027836499801700706 (visited on 04/27/2017).

[65] David Fletcher et al. "RateSetter: roadmap for faster, safer, and better platform train interface design and operation using evolutionary optimisation". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. GECCO '18. New York, NY, USA: Association for Computing Machinery, July 2, 2018, pp. 1230–1237. ISBN: 978-1-4503-5618-3. DOI: 10.1145/3205455.3205605. URL: https://doi.org/10.1145/3205455.3205605 (visited on 10/28/2022).

[66] Chiara Fulgenzi, Anne Spalanzani, and Christian Laugier. "Dynamic Obstacle Avoidance in uncertain environment combining PVOs and Occupancy Grid". In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. 2007 IEEE International Conference on Robotics and Automation. Rome, Italy: IEEE, Apr. 2007, pp. 1610–1616. ISBN: 978-1-4244-0602-9. DOI: 10.1109/ROBOT.2007.363554. URL: http://ieeexplore.ieee.org/document/4209318/ (visited on 02/05/2019).

[67] Martin Funkquist and Staffan Sandberg. *Simulating Group Formations Using RVO*. 2016.

[68] Michael Garland et al. "Parallel computing experiences with CUDA". In: *IEEE micro* 28.4 (2008), pp. 13–27. DOI: 10.1109/mm.2008.57. URL: https://doi.org/10.1109%2Fmm.2008.57.

[69] Jared Gearhart, Kirstin Adair, and Richard Detry. *Comparison of Open-Source Linear Ptorgrramming Solvers*.

[70] A. Giese, D. Latypov, and N. M. Amato. "Reciprocally-Rotating Velocity Obstacles". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 2014 IEEE International Conference on Robotics and Automation (ICRA). May 2014, pp. 3234–3241. DOI: 10.1109/ICRA.2014.6907324.

[71] Jan Gogoll and Julian F Müller. "Autonomous cars: in favor of a mandatory ethics setting". In: *Science and engineering ethics* 23.3 (2017), pp. 681–700.

[72] A. Golas et al. "Hybrid Long-Range Collision Avoidance for Crowd Simulation". In: *IEEE Transactions on Visualization and Computer Graphics* 20.7 (July 2014), pp. 1022–1034. ISSN: 1077-2626. DOI: 10.1109/TVCG.2013.235.

[73] G Greef. "The revised simplex algorithm on a GPU". Honours Year Project. Stellenbosch University South Africa, 2005.

[74] Junyao Guo, Unmesh Kurup, and Mohak Shah. "Is It Safe to Drive? An Overview of Factors, Metrics, and Datasets for Driveability Assessment in Autonomous Driving". In: *IEEE Transactions on Intelligent Transportation Systems* (2019).

[75] Ke Guo et al. "VR-ORCA: Variable Responsibility Optimal Reciprocal Collision Avoidance". In: *IEEE Robotics and Automation Letters* 6.3 (July 2021), pp. 4520–4527. ISSN: 2377-3766. DOI: 10.1109/LRA.2021.3067851.

[76] Amit Gurung and Rajarshi Ray. "Simultaneous Solving of Batched Linear Programs on a GPU". In: *arXiv:1802.08557 [cs]* (Feb. 21, 2018). arXiv: 1802.08557. URL: http://arxiv.org/abs/1802.08557 (visited on 10/09/2018).

[77] Amit Gurung and Rajarshi Ray. "Solving Batched Linear Programs on GPU and Multicore CPU". In: *arXiv:1609.08114 [cs]* (Sept. 26, 2016). arXiv: 1609.08114. URL: http://arxiv.org/abs/1609.08114 (visited on 05/11/2017).

[78] Stephen Guy, Ming Lin, and Dinesh Manocha. "Modeling collision avoidance behavior for virtual humans". In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems. Jan. 1, 2010, pp. 575–582. DOI: 10.1145/1838178.1838182.

[79] Stephen J. Guy et al. "A statistical similarity measure for aggregate crowd dynamics". In: *ACM Transactions on Graphics* 31.6 (Nov. 1, 2012), p. 1. ISSN: 07300301. DOI: 10.1145/2366145.2366209. URL: http://dl.acm.org/citation.cfm?doid=2366145.2366209 (visited on 06/15/2018).

[80] Stephen J. Guy et al. "Least-effort trajectories lead to emergent crowd behaviors". In: *Physical Review E* 85.1 (Jan. 17, 2012), p. 016110. ISSN: 1539-3755, 1550-2376. DOI: 10.1103/PhysRevE.85.016110. URL: https://link.aps.org/doi/10.1103/PhysRevE.85.016110 (visited on 05/17/2023).

[81] Stephen. J. Guy et al. "ClearPath: Highly Parallel Collision Avoidance for Multi-agent Simulation". In: *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '09. New York, NY, USA: ACM, 2009, pp. 177–187. ISBN: 978-1-60558-610-6. DOI: 10.1145/1599470.1599494. URL: http://doi.acm.org/10.1145/1599470.1599494 (visited on 10/07/2016).

[82] Milad Haghani et al. "Panic, Irrationality, and Herding: Three Ambiguous Terms in Crowd Dynamics Research". In: *Journal of Advanced Transportation* 2019 (Aug. 8, 2019). Publisher: Hindawi, e9267643. ISSN: 0197-6729. DOI: 10.1155/2019/9267643. URL: https://www.hindawi.com/journals/jat/2019/9267643/ (visited on 07/31/2023).

[83] J. a. J. Hall. "Towards a practical parallelisation of the simplex method". In: *Computational Management Science* 7.2 (Apr. 1, 2010), pp. 139–170. ISSN: 1619-697X, 1619-6988. DOI: 10.1007/s10287-008-0080-5. URL: https://doi.org/10.1007/s10287-008-0080-5 (visited on 02/05/2018).

[84] Yanbin Han and Hong Liu. "Modified social force model based on information transmission toward crowd evacuation simulation". In: *Physica A: Statistical Mechanics and its Applications* 469 (Mar. 1, 2017), pp. 499–509. ISSN: 0378-4371. DOI: 10.1016/j.physa.2016.11.014. URL: https://www.sciencedirect.com/science/article/pii/S0378437116308196 (visited on 05/22/2023).

[85] Sam Hayes et al. "Validation of Agent-Based Passenger Movement Modeling for Railway Stations Subject to Social Distancing During the COVID-19 Pandemic". In: *Transportation Research Record* (June 1, 2022). Publisher: SAGE Publications Inc, p. 03611981221093634. ISSN: 0361-1981. DOI: 10.1177/03611981221093634. URL: https://doi.org/10.1177/03611981221093634 (visited on 05/23/2023).

[86] Liang He et al. "Dynamic Group Behaviors for Interactive Crowd Simulation". In: *arXiv:1602.03623 [cs]* (Feb. 11, 2016). arXiv: 1602.03623. URL: http://arxiv.org/abs/1602.03623 (visited on 10/17/2016).

[87] Dirk Helbing, Illes Farkas, and Tamas Vicsek. "Simulating Dynamical Features of Escape Panic". In: *Nature* 407.6803 (Sept. 2000), pp. 487–490. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/35035023. arXiv: cond-mat/0009448. URL: http://arxiv.org/abs/cond-mat/0009448 (visited on 08/04/2022).

[88] Dirk Helbing and Péter Molnár. "Social force model for pedestrian dynamics". In: *Physical Review E* 51.5 (May 1, 1995), pp. 4282–4286. DOI: 10.1103/PhysRevE.51.4282. URL: http://link.aps.org/doi/10.1103/PhysRevE.51.4282 (visited on 10/06/2016).

[89] Dirk Helbing et al. "Simulation of pedestrian crowds in normal and evacuation situations". In: vol. 21. Jan. 1, 2002, pp. 21–58.

[90] L. F. Henderson. "On the fluid mechanics of human crowd motion". In: *Transportation Research* 8.6 (Dec. 1, 1974), pp. 509–515. ISSN: 0041-1647. DOI: 10.1016/0041-1647(74)90027-6. URL: https://www.sciencedirect.com/science/article/pii/0041164774900276 (visited on 09/30/2022).

[91] L. F. Henderson. "The Statistics of Crowd Fluids". In: *Nature* 229.5284 (Feb. 1971), pp. 381–383. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/229381a0. URL: https://www.nature.com/articles/229381a0 (visited on 09/30/2022).

[92] Colin M. Henein and Tony White. "Agent-Based Modelling of Forces in Crowds". In: *Multi-Agent and Multi-Agent-Based Simulation*. Ed. by Paul Davidsson, Brian Logan, and Keiki Takadama. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2005, pp. 173–184. ISBN: 978-3-540-32243-6. DOI: 10.1007/978-3-540-32243-6_14.

[93] Claudia Hollmann. "A cognitive human behaviour model for pedestrian behaviour simulation". PhD thesis. University of Greenwich, Jan. 2015. 372 pp. URL: https://gala.gre.ac.uk/id/eprint/13831/ (visited on 05/22/2023).

[94] Catherine Holloway et al. "Effect of vertical step height on boarding and alighting time of train passengers". In: *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit* 230.4 (May 2016), pp. 1234–1241. ISSN: 0954-4097, 2041-3017. DOI: 10.1177/0954409715590480. URL: http://journals.sagepub.com/doi/10.1177/0954409715590480 (visited on 07/11/2022).

[95] Sebastian Hörl. "Agent-based simulation of autonomous taxi services with dynamic demand responses". In: *Procedia Computer Science* 109 (2017), pp. 899–904.

[96] Yanghui Hu and Nikolai W. F. Bode. "A systematic review and meta-analysis on the effect social groups have on the egress times of pedestrian crowds". In: *Transportmetrica A: Transport Science* 0.0 (Nov. 9, 2021). Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/23249935.2021.1998243, pp. 1–20. ISSN: 2324-9935. DOI: 10.1080/23249935.2021.1998243. URL: https://doi.org/10.1080/23249935.2021.1998243 (visited on 11/04/2022).

[97] Yanghui Hu et al. "Social groups barely change the speed-density relationship in unidirectional pedestrian flow, but affect operational behaviours". In: *Safety Science* 139 (July 1, 2021), p. 105259. ISSN: 0925-7535. DOI: 10.1016/j.ssci.2021.105259. URL: https://www.sciencedirect.com/science/article/pii/S0925753521001041 (visited on 06/24/2022).

[98] Shenshi Huang et al. "Experimental study on one-dimensional movement of luggage-laden pedestrian". In: *Physica A: Statistical Mechanics and its Applications* 516 (Feb. 2019), pp. 520–528. ISSN: 03784371. DOI: 10.1016/j.physa.2018.09.038. URL: https://linkinghub.elsevier.com/retrieve/pii/S0378437118311646 (visited on 07/15/2019).

[99] Morgan J. Hurley et al. *SFPE Handbook of Fire Protection Engineering*. Google-Books-ID: xP2zCgAAQBAJ. Springer, Oct. 7, 2015. 3510 pp. ISBN: 978-1-4939-2565-0.

[100] *IBM Effects of multithread execution in CPLEX Optimization Studio models - United States*. Dec. 4, 2013. URL: http://www.ibm.com/support,%20https://www-304.ibm.com/support/docview.wss?uid=swg21653811 (visited on 10/10/2018).

[101] *IBM ILOG CPLEX Optimizer performance benchmarks | IBM Analytics*. URL: https://www.ibm.com/analytics/optimization-software (visited on 10/10/2018).

[102] IBM ILOG. *IBM ILOG CPLEX V12.1: User's manual for CPLEX*. Vol. 46. Jan. 1, 2009. 157 pp.

[103] John James. "A Preliminary Study of the Size Determinant in Small Group Interaction". In: *American Sociological Review* 16.4 (1951). Publisher: [American Sociological Association, Sage Publications, Inc.], pp. 474–477. ISSN: 0003-1224. DOI: 10.2307/2088278. URL: https://www.jstor.org/stable/2088278 (visited on 11/04/2022).

[104] Asja Jelić et al. "Properties of pedestrians walking in line: Fundamental diagrams". In: *Physical Review E* 85.3 (Mar. 28, 2012), p. 036111. DOI: 10.1103/PhysRevE.85.036111. URL: http://link.aps.org/doi/10.1103/PhysRevE.85.036111 (visited on 10/12/2016).

[105] Adwait Jog et al. "OWL: cooperative thread array aware scheduling techniques for improving GPGPU performance". In: (2013), p. 12. DOI: 10.1145/2499368.2451158. URL: https://doi.org/10.1145%2F2499368.2451158.

[106] Jin Hyuk Jung and DIANNE P. O'Leary. "Implementing an interior point method for linear programs on a CPU-GPU system". In: *Electronic Transactions on Numerical Analysis* 28 (2008), pp. 174–189.

[107] Jin Hyuk Jung and Dianne P. O'Leary. "Cholesky decomposition and linear programming on a GPU". In: *Scholarly Paper, University of Maryland* (2006).

[108] B Raghuram Kadali and P Vedagiri. "Modelling pedestrian road crossing behaviour under mixed traffic condition". In: 55 (2013).

[109] Mubbasir Kapadia et al. "SteerBug: An Interactive Framework for Specifying and Detecting Steering Behaviors". In: *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '09. New York, NY, USA: ACM, 2009, pp. 209–216. ISBN: 978-1-60558-610-6. DOI: 10.1145/1599470.1599497. URL: http://doi.acm.org/10.1145/1599470.1599497 (visited on 06/15/2018).

[110] Mubbasir Kapadia et al. "Virtual Crowds: Steps Toward Behavioral Realism". In: *Synthesis Lectures on Visual Computing* 7.4 (Nov. 10, 2015), pp. 1–270. ISSN: 2469-4215. DOI: 10.2200/S00673ED1V01Y201509CGR020. URL: http://www.morganclaypool.com/doi/abs/10.2200/S00673ED1V01Y201509CGR020 (visited on 10/07/2016).

[111] I. Karamouzas and M. Overmars. "Simulating and Evaluating the Local Behavior of Small Pedestrian Groups". In: *IEEE Transactions on Visualization and Computer Graphics* 18.3 (Mar. 2012), pp. 394–406. ISSN: 1077-2626. DOI: 10.1109/TVCG.2011.133.

[112] Twin Karmakharm, Paul Richmond, and Daniela Romano. "Agent-based Large Scale Simulation of Pedestrians With Adaptive Realistic Navigation Vector Fields." In: Theory and Practice of Computer Graphics 2010, TPCG 2010 - Eurographics UK Chapter Proceedings. Jan. 1, 2010, pp. 67–74. DOI: 10.2312/LocalChapterEvents/TPCG/TPCG10/067-074.

[113] Hyung Joo Kim et al. "Dynamic Motion Planning of 3D Human Locomotion Using Gradient-Based Optimization". In: *Journal of Biomechanical Engineering* 130.3 (Apr. 21, 2008), pp. 031002–031002. ISSN: 0148-0731. DOI: 10.1115/1.2898730. URL: http://dx.doi.org/10.1115/1.2898730 (visited on 11/01/2016).

[114] Peter Kipfer. "LCP Algorithms for Collision Detection Using CUDA". In: *GPU Gems*. Vol. 3. Addison Wesley, 2007, pp. 723–740. ISBN: 978-0-321-51526-1. URL: http://http.developer.nvidia.com/GPUGems3/gpugems3_ch33.html (visited on 10/24/2016).

[115] Mariam Kiran et al. "FLAME: simulating large populations of agents on parallel hardware architectures". In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*. 2010, pp. 1633–1636.

[116] Ansgar Kirchner et al. "Discretisation effects and the influence of walking speed in cellular automata models for pedestrian dynamics". In: *Journal of Statistical Mechanics: Theory and Experiment* 2004.10 (Oct. 28, 2004), P10011. ISSN: 1742-5468. DOI: 10.1088/1742-5468/2004/10/P10011. arXiv: cond-mat/0410706. URL: http://arxiv.org/abs/cond-mat/0410706 (visited on 02/05/2019).

[117] Boris Kluge and Erwin Prassler. "Recursive Probabilistic Velocity Obstacles for Reflective Navigation". In: *Field and Service Robotics: Recent Advances in Reserch and Applications*. Ed. by Shin'ichi Yuta et al. Springer Tracts in Advanced Robotics. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 71–79. ISBN: 978-3-540-32854-4. DOI: 10.1007/10991459_8. URL: https://doi.org/10.1007/10991459_8 (visited on 02/05/2019).

[118] Franziska Klügl. "A validation methodology for agent-based simulations". In: *Proceedings of the 2008 ACM symposium on Applied computing*. SAC '08: The 2008 ACM Symposium on Applied Computing. Fortaleza, Ceara Brazil: ACM, Mar. 16, 2008, pp. 39–43. ISBN: 978-1-59593-753-7. DOI: 10.1145/1363686.1363696. URL: https://dl.acm.org/doi/10.1145/1363686.1363696 (visited on 08/09/2023).

[119] Franziska Klügl and Guido Rindsfüser. "Large-Scale Agent-Based Pedestrian Simulation". In: *Multiagent System Technologies*. Ed. by Paolo Petta et al. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2007, pp. 145–156. ISBN: 978-3-540-74949-3. DOI: 10.1007/978-3-540-74949-3_13.

[120] Adrian Kłusek et al. "An implementation of the Social Distances Model using multi-GPU systems". In: *The International Journal of High Performance Computing Applications* 32.4 (July 2018), pp. 482–495. ISSN: 1094-3420, 1741-2846. DOI: 10.1177/1094342016679492. URL: http://journals.sagepub.com/doi/10.1177/1094342016679492 (visited on 05/21/2023).

[121] Raphael Korbmacher and Antoine Tordeux. "Review of Pedestrian Trajectory Prediction Methods: Comparing Deep Learning and Knowledge-Based Approaches". In: *IEEE Transactions on Intelligent Transportation Systems* (2022), pp. 1–19. ISSN: 1524-9050, 1558-0016. DOI: 10.1109/TITS.2022.3205676. URL: https://ieeexplore.ieee.org/document/9899358/ (visited on 09/29/2022).

[122] Gerta Köster et al. "On modelling the influence of group formations in a crowd". In: *Contemporary Social Science* 6.3 (Nov. 1, 2011), pp. 397–414. ISSN: 2158-2041. DOI: 10.1080/21582041.2011.619867. URL: http://dx.doi.org/10.1080/21582041.2011.619867 (visited on 10/17/2016).

[123] Gerta Köster et al. "Validation of Crowd Models Including Social Groups". In: Pedestrian and Evacuation Dynamics 2012 SE - 87. Jan. 1, 2012. ISBN: 978-3-319-02446-2. DOI: 10.1007/978-3-319-02447-9-87.

[124] Barbara Krausz and Christian Bauckhage. "Automatic detection of dangerous motion behavior in human crowds". In: *2011 8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. 2011 8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS). Klagenfurt, Austria: IEEE, Aug. 2011, pp. 224–229. ISBN: 978-1-4577-0844-2 978-1-4577-0845-9. DOI: 10.1109/AVSS.2011.6027326. URL: https://ieeexplore.ieee.org/document/6027326/ (visited on 08/29/2023).

[125] Erica D. Kuligowski, Richard D. Peacock, and Bryan L. Hoskins. "A Review of Building Evacuation Models, 2nd Edition". In: *NIST* (Nov. 1, 2010). Last Modified: 2017-02-19T20:02-05:00 Publisher: Erica D. Kuligowski, Richard D. Peacock, Bryan L. Hoskins. URL: https://www.nist.gov/publications/review-building-evacuation-models-2nd-edition (visited on 07/31/2023).

[126] Arthur D. Kuo. "The six determinants of gait and the inverted pendulum analogy: A dynamic walking perspective". In: *Human Movement Science*. European Workshop on Movement Science 2007European Workshop on Movement Science 2007 26.4 (Aug. 2007), pp. 617–656. ISSN: 0167-9457. DOI: 10.1016/j.humov.2007.04.003. URL: http://www.sciencedirect.com/science/article/pii/S0167945707000309 (visited on 10/21/2016).

[127] Taras I. Lakoba, D. J. Kaup, and Neal M. Finkelstein. "Modifications of the Helbing-Molnár-Farkas-Vicsek Social Force Model for Pedestrian Evolution". In: *Simulation* 81.5 (May 2005), pp. 339–352. ISSN: 0037-5497. DOI: 10.1177/0037549705052772. URL: http://dx.doi.org/10.1177/0037549705052772 (visited on 04/17/2017).

[128] M. E. Lalami, V. Boyer, and D. El-Baz. "Efficient Implementation of the Simplex Method on a CPU-GPU System". In: *2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum*. 2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum. May 2011, pp. 1999–2006. DOI: 10.1109/IPDPS.2011.362.

[129] Alon Lerner et al. "Context-Dependent Crowd Evaluation". In: *Computer Graphics Forum* 29.7 (2010), pp. 2197–2206. ISSN: 1467-8659. DOI: 10.1111/j.1467-8659.2010.01808.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2010.01808.x (visited on 06/15/2018).

[130] Alon Lerner et al. *Data Driven Evaluation of Crowds*. Nov. 21, 2009. 75 pp. DOI: 10.1007/978-3-642-10347-6_7.

[131] Bo Li and Ramakrishnan Mukundan. *A Comparative Analysis of Spatial Partitioning Methods for Large-scale, Real-time Crowd Simulation*. Václav Skala - UNION Agency, 2013. ISBN: 978-80-86943-75-6. URL: http://dspace5.zcu.cz/handle/11025/10652 (visited on 03/28/2019).

[132] Kunming Li et al. "Crowd Prediction and Autonomous Navigation with Partial Observations". In: *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC). Macau, China: IEEE, Oct. 8, 2022, pp. 1829–1836. ISBN: 978-1-66546-880-0. DOI: 10.1109/ITSC55140.2022.9922257. URL: https://ieeexplore.ieee.org/document/9922257/ (visited on 05/17/2023).

[133] Patrick Lin. "Why ethics matters for autonomous cars". In: *Autonomous driving*. Springer, Berlin, Heidelberg, 2016, pp. 69–85.

[134] Baoxi Liu et al. "A social force evacuation model driven by video data". In: *Simulation Modelling Practice and Theory* 84 (May 1, 2018), pp. 190–203. ISSN: 1569-190X. DOI: 10.1016/j.simpat.2018.02.007. URL: https://www.sciencedirect.com/science/article/pii/S1569190X18300236 (visited on 05/22/2023).

[135] R Lohner et al. "Real-time micro-modelling of a million pedestrians". In: *Engineering Computations* 33.1 (Mar. 7, 2016), pp. 217–237. ISSN: 0264-4401. DOI: 10.1108/EC-02-2015-0036. URL: https://www.emerald.com/insight/content/doi/10.1108/EC-02-2015-0036/full/html (visited on 05/21/2023).

[136] R. Lovreglio et al. "A pre-evacuation database for use in egress simulations". In: *Fire Safety Journal* 105 (Apr. 1, 2019), pp. 107–128. ISSN: 0379-7112. DOI: 10.1016/j.firesaf.2018.12.009. URL: https://www.sciencedirect.com/science/article/pii/S0379711218302212 (visited on 08/09/2023).

[137] Charles M. Macal and Michael J. North. "Agent-based modeling and simulation". In: *Proceedings of the 2009 Winter Simulation Conference (WSC)*. Proceedings of the 2009 Winter Simulation Conference (WSC). ISSN: 1558-4305. Dec. 2009, pp. 86–98. DOI: 10.1109/WSC.2009.5429318.

[138] Andrew Makhorin. *GLPK (GNU linear programming kit)*. 2008. URL: https://www.gnu.org/software/glpk/glpk.html (visited on 05/25/2018).

[139] Bertrand Maury and Juliette Venel. "A mathematical framework for a crowd motion model". In: *Comptes Rendus Mathematique* 346.23 (Dec. 1, 2008), pp. 1245–1250. ISSN: 1631-073X. DOI: 10.1016/j.crma.2008.10.014. URL: https://www.sciencedirect.com/science/article/pii/S1631073X08003051 (visited on 10/04/2022).

[140] Neil McBride. "The Ethics of Driverless Cars". In: *SIGCAS Comput. Soc.* 45.3 (Jan. 2016), pp. 179–184. ISSN: 0095-2737. DOI: 10.1145/2874239.2874265. URL: http://doi.acm.org/10.1145/2874239.2874265 (visited on 10/16/2019).

[141] Duane Merrill. *CUB library*. 2013.

[142] Rodolfo Migon Favaretto et al. "Investigating cultural aspects in the fundamental diagram using convolutional neural networks and virtual agent simulation". In: *Computer Animation and Virtual Worlds* 30.3 (May 2019). ISSN: 1546-4261, 1546-427X. DOI: 10.1002/cav.1899. URL: https://onlinelibrary.wiley.com/doi/10.1002/cav.1899 (visited on 08/01/2023).

[143] Hans Mittelmann. *Benchmark of Simplex LP solvers*. Feb. 10, 2018. URL: http://plato.asu.edu/ftp/lpsimp.html (visited on 10/08/2018).

[144] M MONONEN. *Recast & Detour*. original-date: 2013-09-15T18:12:45Z. 2014. URL: https://github.com/recastnavigation/recastnavigation (visited on 05/22/2023).

[145] Mehdi Moussaïd et al. "The Walking Behaviour of Pedestrian Social Groups and Its Impact on Crowd Dynamics". In: *PLOS ONE* 5.4 (Apr. 7, 2010), e10047. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0010047. URL: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0010047 (visited on 10/07/2016).

[146] Mehdi Moussaïd et al. "Traffic Instabilities in Self-Organized Pedestrian Crowds". In: *PLOS Comput Biol* 8.3 (Mar. 22, 2012), e1002442. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1002442. URL: http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1002442 (visited on 10/07/2016).

[147] HUBERT MRÓZ and JAROSŁAW WĄS. "Discrete vs. Continuous Approach in Crowd Dynamics Modeling Using GPU Computing". In: *Cybernetics and Systems* 45.1 (Jan. 2, 2014). Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/019697 pp. 25–38. ISSN: 0196-9722. DOI: 10.1080/01969722.2014.862104. URL: https://doi.org/10.1080/01969722.2014.862104 (visited on 05/21/2023).

[148] Rahul Narain et al. "Aggregate Dynamics for Dense Crowd Simulation". In: *ACM SIGGRAPH Asia 2009 Papers*. SIGGRAPH Asia '09. New York, NY, USA: ACM, 2009, 122:1–122:8. ISBN: 978-1-60558-858-2. DOI: 10.1145/1661412.1618468. URL: http://doi.acm.org/10.1145/1661412.1618468 (visited on 10/10/2016).

[149] Sahil Narang et al. "Generating Pedestrian Trajectories Consistent with the Fundamental Diagram Based on Physiological and Psychological Factors". In: *PLOS ONE* 10.4 (Apr. 13, 2015), e0117856. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0117856. URL: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0117856 (visited on 10/24/2017).

[150] Ashwin Nayak and Zhendong Su. "Randomized Algorithms for Linear Programming". In: *Citeseer* (1996).

[151] Alexandre Nicolas and Fadratul Hafinaz. "Social groups in pedestrian crowds: Review of their influence on the dynamics and their modelling". In: *arXiv:2107.13293 [physics]* (July 28, 2021). arXiv: 2107.13293. URL: http://arxiv.org/abs/2107.13293 (visited on 06/24/2022).

[152] NVidia. *cuSOLVER*. URL: https://docs.nvidia.com/cuda/cusolver/index.html (visited on 05/22/2023).

[153] Nvidia. *Programming guide*. 2010.

[154] Nvidia. *Tuning CUDA Applications for Maxwell*. 2018. URL: http://docs.nvidia.com/cuda/maxwell-tuning-guide/index.html (visited on 04/25/2018).

[155] C. O'Sullivan and C. Ennis. "Metropolis: Multisensory Simulation of a Populated City". In: *2011 Third International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*. 2011 Third International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES). May 2011, pp. 1–7. DOI: 10.1109/VS-GAMES.2011.9.

[156] Jan Ondřej et al. "A Synthetic-vision Based Steering Approach for Crowd Simulation". In: *ACM SIGGRAPH 2010 Papers*. SIGGRAPH '10. New York, NY, USA: ACM, 2010, 123:1–123:9. ISBN: 978-1-4503-0210-4. DOI: 10.1145/1833349.1778860. URL: http://doi.acm.org/10.1145/1833349.1778860 (visited on 10/12/2016).

[157] Matthew Owen, Edwin R. Galea, and Peter J. Lawrence. "The Exodus Evacuation Model Applied To Building Evacuation Scenarios". In: *Journal of Fire Protection Engineering* 8.2 (May 1, 1996). Publisher: SAGE Publications Ltd STM, pp. 65–84. ISSN: 1042-3915. DOI: 10.1177/104239159600800202. URL: https://journals.sagepub.com/doi/abs/10.1177/104239159600800202 (visited on 05/23/2023).

[158] Daniel R. Parisi et al. "Pedestrian dynamics at the running of the bulls evidence an inaccessible region in the fundamental diagram". In: *Proceedings of the National Academy of Sciences* 118.50 (Dec. 14, 2021). Publisher: Proceedings of the National Academy of Sciences, e2107827118. DOI: 10.1073/pnas.2107827118. URL: https://www.pnas.org/doi/full/10.1073/pnas.2107827118 (visited on 11/04/2022).

[159] Daniel R. Parisi et al. *Social Distance Characterization by means of Pedestrian Simulation*. Sept. 8, 2020. DOI: 10.48550/arXiv.2009.04019. arXiv: 2009.04019[physics]. URL: http://arxiv.org/abs/2009.04019 (visited on 05/22/2023).

[160] Chonhyon Park et al. "Simulating high-DOF human-like agents using hierarchical feedback planner". In: *Proceedings of the 21st ACM Symposium on Virtual Reality Software and Technology*. ACM, 2015, pp. 153–162. URL: http://dl.acm.org/citation.cfm?id=2821604 (visited on 10/24/2016).

[161] Dino Pedreschi et al. "Meaningful Explanations of Black Box AI Decision Systems". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.1 (July 17, 2019). Number: 01, pp. 9780–9784. ISSN: 2374-3468. DOI: 10.1609/aaai.v33i01.33019780. URL: https://ojs.aaai.org/index.php/AAAI/article/view/5050 (visited on 05/21/2023).

[162] Ingrid Pettersson and I. C. MariAnne Karlsson. "Setting the stage for autonomous cars: a pilot study of future autonomous driving experiences". In: *IET Intelligent Transport Systems* 9.7 (May 12, 2015), pp. 694–701. ISSN: 1751-9578. DOI: 10.1049/iet-its.2014.0168. URL: https://digital-library.theiet.org/content/journals/10.1049/iet-its.2014.0168 (visited on 10/14/2019).

[163] Nikolaos Ploskas and Nikolaos Samaras. "Efficient GPU-based implementations of simplex type algorithms". In: *Applied Mathematics and Computation* 250 (Jan. 1, 2015), pp. 552–570. ISSN: 0096-3003. DOI: 10.1016/j.amc.2014.10.096. URL: https://doi.org/10.1016/j.amc.2014.10.096.

[164] Nikolaos Ploskas and Nikolaos Samaras. "GPU accelerated pivoting rules for the simplex algorithm". In: *Journal of Systems and Software* 96 (Oct. 1, 2014), pp. 1–9. ISSN: 0164-1212. DOI: 10.1016/j.jss.2014.04.047. URL: https://doi.org/10.1016/j.jss.2014.04.047.

[165] Atanas Poibrenski et al. "M2P3: multimodal multi-pedestrian path prediction by self-driving cars with egocentric vision". In: *Proceedings of the 35th Annual ACM Symposium on Applied Computing*. SAC '20. New York, NY, USA: Association for Computing Machinery, Mar. 30, 2020, pp. 190–197. ISBN: 978-1-4503-6866-7. DOI: 10.1145/3341105.3373877. URL: https://doi.org/10.1145/3341105.3373877 (visited on 09/29/2022).

[166] E. Prassler, J. Scholz, and P. Fiorini. "Navigating a Robotic Wheelchair in a Railway Station during Rush Hour". In: *The International Journal of Robotics Research* 18.7 (July 1, 1999), pp. 711–727. ISSN: 0278-3649. DOI: 10.1177/02783649922066529. URL: http://journals.sagepub.com/doi/abs/10.1177/02783649922066529 (visited on 04/18/2017).

[167] Manon Prédhumeau, Anne Spalanzani, and Julie Dugdale. "Pedestrian Behavior in Shared Spaces With Autonomous Vehicles: An Integrated Framework and Review". In: *IEEE Transactions on Intelligent Vehicles* 8.1 (Jan. 2023). Conference Name: IEEE Transactions on Intelligent Vehicles, pp. 438–457. ISSN: 2379-8904. DOI: 10.1109/TIV.2021.3116436.

[168] Amir Rasouli and John K Tsotsos. "Autonomous vehicles that interact with pedestrians: A survey of theory and practice". In: *IEEE transactions on intelligent transportation systems* (2019).

[169] Amir Rasouli et al. *Multi-Modal Hybrid Architecture for Pedestrian Action Prediction*. Nov. 16, 2020. arXiv: 2012.00514[cs]. URL: http://arxiv.org/abs/2012.00514 (visited on 05/12/2023).

[170] Z. Ren et al. "Group Modeling: A Unified Velocity-Based Approach". In: *Computer Graphics Forum* (Sept. 1, 2016), n/a–n/a. ISSN: 1467-8659. DOI: 10.1111/cgf.12993. URL: http://onlinelibrary.wiley.com/doi/10.1111/cgf.12993/abstract (visited on 10/17/2016).

[171] Craig W. Reynolds. "Flocks, Herds and Schools: A Distributed Behavioral Model". In: *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '87. New York, NY, USA: ACM, 1987, pp. 25–34. ISBN: 978-0-89791-227-3. DOI: 10.1145/37401.37406. URL: http://doi.acm.org/10.1145/37401.37406 (visited on 04/17/2017).

[172] Paul Richmond. *Flame gpu technical report and user guide*. Department of Computer Science Technical Report CS-11-03. University of Sheffield, 2011.

[173] Paul Richmond and Daniela M. Romano. "A high performance framework for agent based pedestrian dynamics on gpu hardware". In: *Proceedings of EUROSIS ESM* 2008 (2008). URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.144.273&rep=rep1&type=pdf (visited on 04/18/2017).

[174] Ivan Rodin et al. *Predicting the Future from First Person (Egocentric) Vision: A Survey*. July 28, 2021. DOI: 10.48550/arXiv.2107.13411. arXiv: 2107.13411[cs]. URL: http://arxiv.org/abs/2107.13411 (visited on 09/29/2022).

[175] Laura von Rueden et al. "Combining Machine Learning and Simulation to a Hybrid Modelling Approach: Current and Future Directions". In: *Advances in Intelligent Data Analysis XVIII*. Ed. by Michael R. Berthold, Ad Feelders, and Georg Krempl. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 548–560. ISBN: 978-3-030-44584-3. DOI: 10.1007/978-3-030-44584-3_43.

[176] M. Rufli, J. Alonso-Mora, and R. Siegwart. "Reciprocal Collision Avoidance With Motion Continuity Constraints". In: *IEEE Transactions on Robotics* 29.4 (Aug. 2013), pp. 899–912. ISSN: 1552-3098. DOI: 10.1109/TRO.2013.2258733.

[177] Stefan Rybacki, Jan Himmelspach, and Adelinde M. Uhrmacher. "Experiments with Single Core, Multi-core, and GPU Based Computation of Cellular Automata". In: *2009 First International Conference on Advances in System Simulation*. 2009 First International Conference on Advances in System Simulation. Sept. 2009, pp. 62–67. DOI: 10.1109/SIMUL.2009.36.

[178] Nikolay Sakharnykh. *GPU Pro Tip: Fast Histograms Using Shared Atomics on Maxwell*. NVIDIA Developer Blog. Mar. 17, 2015. URL: https://devblogs.nvidia.com/gpu-pro-tip-fast-histograms-using-shared-atomics-maxwell/ (visited on 10/10/2018).

[179] Wilko Schwarting et al. "Social behavior for autonomous vehicles". In: *Proceedings of the National Academy of Sciences* 116.50 (2019), pp. 24972–24978.

[180] Raimund Seidel. "Small-dimensional linear programming and convex hulls made easy". In: *Discrete & Computational Geometry* 6.3 (Sept. 1, 1991), pp. 423–434. ISSN: 0179-5376, 1432-0444. DOI: 10.1007/BF02574699. URL: https://doi.org/10.1007/BF02574699 (visited on 05/11/2017).

[181] Armin Seyfried et al. "Enhanced Empirical Data for the Fundamental Diagram and the Flow Through Bottlenecks". In: *Pedestrian and Evacuation Dynamics 2008*. Ed. by Wolfram W. F. Klingsch et al. Berlin, Heidelberg: Springer, 2010, pp. 145–156. ISBN: 978-3-642-04504-2. DOI: 10.1007/978-3-642-04504-2_11.

[182] Jiten Shah, G.J. Joshi, and Purnima Parida. "Behavioral Characteristics of Pedestrian Flow on Stairway at Railway Station". In: *Procedia - Social and Behavioral Sciences* 104 (Dec. 2013), pp. 688–697. ISSN: 18770428. DOI: 10.1016/j.sbspro.2013.11.163. URL: https://linkinghub.elsevier.com/retrieve/pii/S1877042813045540 (visited on 04/23/2019).

[183] Zhigang Shi et al. "The effect of symmetrical exit layout on luggage-laden pedestrian movement in the double-exit room". In: *Safety Science* 155 (Nov. 1, 2022), p. 105874. ISSN: 0925-7535. DOI: 10.1016/j.ssci.2022.105874. URL: https://www.sciencedirect.com/science/article/pii/S0925753522002132 (visited on 11/01/2022).

[184] Z. Shiller, F. Large, and S. Sekhavat. "Motion planning in dynamic environments: obstacles moving along arbitrary trajectories". In: *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*. Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164). Vol. 4. 2001, 3716–3721 vol.4. DOI: 10.1109/ROBOT.2001.933196.

[185] Zvi Shiller, Rajan Prasanna, and Jeremy Salinger. "A Unified Approach to Forward and Lane-Change Collision Warning for Driver Assistance and Situational Awareness". In: *SAE Mobilus* (2008). DOI: 10.4271/2008-01-0204. URL: http://papers.sae.org/2008-01-0204/ (visited on 04/18/2017).

[186] Nirajan Shiwakoti et al. "Examining influence of merging architectural features on pedestrian crowd movement". In: *Safety Science* 75 (June 2015), pp. 15–22. ISSN: 09257535. DOI: 10.1016/j.ssci.2015.01.009. URL: https://linkinghub.elsevier.com/retrieve/pii/S0925753515000107 (visited on 05/12/2023).

[187] Harmeet Singh et al. "Modelling subgroup behaviour in crowd dynamics DEM simulation". In: *Applied Mathematical Modelling* 33.12 (Dec. 1, 2009), pp. 4408–4423. ISSN: 0307-904X. DOI: 10.1016/j.apm.2009.03.020. URL: http://www.sciencedirect.com/science/article/pii/S0307904X09000808.

[188] Shawn Singh et al. "An Open Framework for Developing, Evaluating, and Sharing Steering Algorithms". In: *Motion in Games*. Ed. by Arjan Egges, Roland Geraerts, and Mark Overmars. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2009, pp. 158–169. ISBN: 978-3-642-10347-6. DOI: 10.1007/978-3-642-10347-6_15.

[189] Shawn Singh et al. "Footstep Navigation for Dynamic Crowds". In: *Symposium on Interactive 3D Graphics and Games*. I3D '11. New York, NY, USA: ACM, 2011, pp. 203–203. ISBN: 978-1-4503-0565-5. DOI: 10.1145/1944745.1944783. URL: http://doi.acm.org/10.1145/1944745.1944783 (visited on 10/19/2016).

[190] Shawn Singh et al. "SteerBench: a benchmark suite for evaluating steering behaviors". In: *Computer Animation and Virtual Worlds* 20.5 (Sept. 1, 2009), pp. 533–548. ISSN: 1546-427X. DOI: 10.1002/cav.277. URL: http://onlinelibrary.wiley.com/doi/10.1002/cav.277/abstract (visited on 11/04/2016).

[191] Shawn Singh et al. "Watch out! a framework for evaluating steering behaviors". In: *International Workshop on Motion in Games*. Springer, 2008, pp. 200–209.

[192] Edmund Smith, Jacek Gondzio, and Julian Hall. "GPU Acceleration of the Matrix-Free Interior Point Method". In: *Parallel Processing and Applied Mathematics*. Springer Berlin Heidelberg, 2012, pp. 681–689. DOI: 10.1007/978-3-642-31464-3_69. URL: https://doi.org/10.1007%2F978-3-642-31464-3_69.

[193] J Snape. *RVO2: Optimal Reciprocal Collision Avoidance (C++)*. URL: https://github.com/snape/RVO2 (visited on 04/14/2017).

[194] Daniele G. Spampinato and Anne C. Elstery. "Linear optimization on modern GPUs". In: *2009 IEEE International Symposium on Parallel & Distributed Processing*. IEEE, May 2009, pp. 1–8. DOI: 10.1109/ipdps.2009.5161106. URL: https://doi.org/10.1109%2Fipdps.2009.5161106.

[195] S. Stuvel et al. "Torso Crowds". In: *IEEE Transactions on Visualization and Computer Graphics* PP.99 (2016), pp. 1–1. ISSN: 1077-2626. DOI: 10.1109/TVCG.2016.2545670.

[196] S. A. Stüvel. *Dense Crowds of Virtual Humans*. Apr. 20, 2016. URL: http://dspace.library.uu.nl/handle/1874/330667 (visited on 11/08/2016).

[197] Kirsten Moana Thompson. "Scale, Spectacle and Movement: Massive Software and Digital Special Effects in The Lord of The Rings". In: *From Hobbits to Hollywood*. Section: From Hobbits to Hollywood. Brill, Jan. 1, 2006, pp. 283–299. ISBN: 978-94-012-0151-3. DOI: 10.1163/9789401201513_021. URL: https://brill.com/display/book/9789401201513/B9789401201513_s021.xml (visited on 05/22/2023).

[198] C. Thornton et al. "Pathfinder: An Agent-Based Egress Simulator". In: *Pedestrian and Evacuation Dynamics*. Ed. by Richard D. Peacock, Erica D. Kuligowski, and Jason D. Averill. Boston, MA: Springer US, 2011, pp. 889–892. ISBN: 978-1-4419-9725-8. DOI: 10.1007/978-1-4419-9725-8_94.

[199] Wouter van Toll et al. "A comparative study of navigation meshes". In: *Proceedings of the 9th International Conference on Motion in Games*. MIG '16. New York, NY, USA: Association for Computing Machinery, Oct. 10, 2016, pp. 91–100. ISBN: 978-1-4503-4592-7. DOI: 10.1145/2994258.2994262. URL: https://doi.org/10.1145/2994258.2994262 (visited on 05/22/2023).

[200] Antoine Tordeux, Mohcine Chraibi, and Armin Seyfried. "Collision-Free Speed Model for Pedestrian Dynamics". In: *Traffic and Granular Flow '15*. Ed. by Victor L. Knoop and Winnie Daamen. Cham: Springer International Publishing, 2016, pp. 225–232. ISBN: 978-3-319-33482-0. DOI: 10.1007/978-3-319-33482-0_29.

[201] Antoine Tordeux and Andreas Schadschneider. "White and relaxed noises in optimal velocity models for pedestrian flow with stop-and-go waves". In: *Journal of Physics A: Mathematical and Theoretical* 49 (May 6, 2016), p. 185101. DOI: 10.1088/1751-8113/49/18/185101.

[202] Paul M. Torrens and Simin Gu. "Inverse augmentation: Transposing real people into pedestrian models". In: *Computers, Environment and Urban Systems* 100 (Mar. 1, 2023), p. 101923. ISSN: 0198-9715. DOI: 10.1016/j.compenvurbsys.2022.101923. URL: https://www.sciencedirect.com/science/article/pii/S0198971522001673 (visited on 01/13/2023).

[203] Monika Twarogowska, Paola Goatin, and Regis Duvigneau. "Comparative Study of Macroscopic Pedestrian Models". In: *Transportation Research Procedia*. The Conference on Pedestrian and Evacuation Dynamics 2014 (PED 2014), 22-24 October 2014, Delft, The Netherlands 2 (Jan. 1, 2014), pp. 477–485. ISSN: 2352-1465. DOI: 10.1016/j.trpro.2014.09.063. URL: https://www.sciencedirect.com/science/article/pii/S2352146514000994 (visited on 07/31/2023).

[204] Ezel Üsten, Helena Lügering, and Anna Sieben. "Pushing and Non-pushing Forward Motion in Crowds: A Systematic Psychological Observation Method for Rating Individual Behavior in Pedestrian Dynamics". In: *Collective Dynamics* 7 (Aug. 5, 2022), pp. 1–16. ISSN: 2366-8539. DOI: 10.17815/CD.2022.138. URL: https://collective-dynamics.eu/index.php/cod/article/view/A138 (visited on 08/01/2023).

[205] Yangzihao Wang et al. “Gunrock: A High-Performance Graph Processing Library on the GPU”. In: *arXiv:1501.05387 [cs]* (2016), pp. 1–12. DOI: 10.1145/2851141.2851145. arXiv: 1501.05387. URL: http://arxiv.org/abs/1501.05387 (visited on 04/17/2018).

[206] We Build Websites - 3bit.co.uk. *About | London Stadium*. URL: http://www.london-stadium.com/stadium/about.html (visited on 07/31/2023).

[207] Ulrich Weidmann. “Transporttechnik der fußgänger: transporttechnische eigenschaften des fußgängerverkehrs, literaturauswertung”. In: *IVT Schriftenreihe* 90 (1993). Publisher: ETH Zurich.

[208] Tomer Weiss et al. “Position-based real-time simulation of large crowds”. In: *Computers & Graphics* 78 (Feb. 1, 2019), pp. 12–22. ISSN: 0097-8493. DOI: 10.1016/j.cag.2018.10.008. URL: http://www.sciencedirect.com/science/article/pii/S0097849318301699 (visited on 05/20/2019).

[209] Edward Wigley and Gillian Rose. “Who’s behind the wheel? Visioning the future users and urban contexts of connected and autonomous vehicle technologies”. In: *Geografiska Annaler: Series B, Human Geography* 102.2 (Apr. 2, 2020), pp. 155–171. ISSN: 0435-3684, 1468-0467. DOI: 10.1080/04353684.2020.1747943. URL: https://www.tandfonline.com/doi/full/10.1080/04353684.2020.1747943 (visited on 01/26/2023).

[210] D. Wilkie, J. van den Berg, and D. Manocha. “Generalized velocity obstacles”. In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems. Oct. 2009, pp. 5573–5578. DOI: 10.1109/IROS.2009.5354175.

[211] Haoling Wu et al. “Research on the Effects of Heterogeneity on Pedestrian Dynamics in Walkway of Subway Station”. In: *Discrete Dynamics in Nature and Society* 2016 (2016), pp. 1–10. ISSN: 1026-0226, 1607-887X. DOI: 10.1155/2016/4961681. URL: http://www.hindawi.com/journals/ddns/2016/4961681/ (visited on 07/15/2019).

[212] Qiancheng Xu et al. *Generalized collision-free velocity model for pedestrian dynamics*. Aug. 27, 2019. DOI: 10.1016/j.physa.2019.122521. arXiv: 1908.10304[physics]. URL: http://arxiv.org/abs/1908.10304 (visited on 10/04/2022).

[213] Kazuhiro Yamamoto, Satoshi Kokubo, and Katsuhiro Nishinari. “Simulation for pedestrian dynamics by real-coded cellular automata (RCA)”. In: *Physica A: Statistical Mechanics and its Applications* 379.2 (June 2007), pp. 654–660. ISSN: 03784371. DOI: 10.1016/j.physa.2007.02.040. URL: https://linkinghub.elsevier.com/retrieve/pii/S0378437107001835 (visited on 02/05/2019).

[214] Zhe Yang et al. “Proxemic group behaviors using reciprocal multi-agent navigation”. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)* (2016), pp. 292–297. DOI: 10.1109/ICRA.2016.7487147.

[215] Jianhong Ye, Xiaohong Chen, and Nanjing Jian. “Impact analysis of human factors on pedestrian traffic characteristics”. In: *Fire Safety Journal* 52 (Aug. 1, 2012), pp. 46–54. ISSN: 0379-7112. DOI: 10.1016/j.firesaf.2012.05.003. URL: http://www.sciencedirect.com/science/article/pii/S0379711212000720 (visited on 12/03/2019).

[216] Zhipeng Yin, Jinghao Liu, and Lei Wang. "Less-Effort Collision Avoidance in Virtual Pedestrian Simulation". In: *Proceedings of the 2019 International Conference on Artificial Intelligence and Computer Science*. AICS 2019. New York, NY, USA: Association for Computing Machinery, July 12, 2019, pp. 488–493. ISBN: 978-1-4503-7150-6. DOI: 10.1145/3349341.3349456. URL: https://doi.org/10.1145/3349341.3349456 (visited on 01/13/2023).

[217] Hao Zhang et al. "Modified two-layer social force model for emergency earthquake evacuation". In: *Physica A: Statistical Mechanics and its Applications* 492 (Feb. 2018), pp. 1107–1119. ISSN: 03784371. DOI: 10.1016/j.physa.2017.11.041. URL: https://linkinghub.elsevier.com/retrieve/pii/S0378437117311172 (visited on 05/21/2023).

[218] Jun Zhang and Armin Seyfried. "Empirical Characteristics of Different Types of Pedestrian Streams". In: *Procedia Engineering* 62 (2013), pp. 655–662. ISSN: 18777058. DOI: 10.1016/j.proeng.2013.08.111. URL: https://linkinghub.elsevier.com/retrieve/pii/S1877705813012939 (visited on 07/25/2019).

[219] Wenwen Zhang et al. "Exploring the impact of shared autonomous vehicles on urban parking demand: An agent-based simulation approach". In: *Sustainable Cities and Society* 19 (2015), pp. 34–45.

[220] Xiaoping Zheng, Tingkuan Zhong, and Mengting Liu. "Modeling crowd evacuation of a building based on seven methodological approaches". In: *Building and Environment* 44.3 (Mar. 2009), pp. 437–445. ISSN: 03601323. DOI: 10.1016/j.buildenv.2008.04.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S0360132308000577 (visited on 09/29/2022).

[221] Z. Zheng and H. Zhang. "Highly parallel crowd simulation using speed field". In: *2014 International Conference on Audio, Language and Image Processing (ICALIP)*. 2014 International Conference on Audio, Language and Image Processing (ICALIP). July 2014, pp. 84–89. DOI: 10.1109/ICALIP.2014.7009762.

[222] Benedikt Zönnchen and Gerta Köster. "GPGPU Computing for Microscopic Pedestrian Simulation". In: *Parallel Computing: Technology Trends*. IOS Press, 2020, pp. 93–104. DOI: 10.3233/APC200029. URL: https://ebooks.iospress.nl/doi/10.3233/APC200029 (visited on 05/21/2023).