The increasing number of people moving from the country to cities has caused that humanity becomes more vulnerable to incidents such as natural disasters, man-made incidents, and bombing attacks. Additionally, events like the World Trade Centre attack in 2012 have led to the developing of the disaster management research field. This is focussed on generating tools and control strategies that provide valuable information to managers who have to take complex decisions to decrease the losses. However, some of the emergency response plans includes evacuating people from the disaster zone. Consequently, traffic evacuation systems are required to generate dynamic plans that take into account the current state of the network.

This research aims to develop a control strategy that takes into account the available resources, location of shelters, accessible data, and the network clearance time. As a result, this approach utilizes officers for guiding traffic because they are an available in any city, they can provide information using mobile devices, and their presence might calm drivers who are more likely to follow orders from policeman than any traffic signal.

The proposed technique is based on an algorithm that utilizes a predictive application for estimating the state of traffic. Thus, it also employs a heuristic to evaluate if it is worthy or not to reroute an evacuation exit. To evaluate the new control strategy, a traffic simulator was created. Its implementation is based on the parallel computing to overcome the computational cost of the different modules of the software application. Thus it provides metrics such as total evacuation time, number of evacuees at shelter for every time step, and the state of intersections during the evacuation. The model was tested in two environments with different distribution of vehicles. The outcomes were compared with those obtained from an uncontrolled environment and those from the random approach. According the metrics, the dynamic strategy has the best performance because its traffic distribution is better and its average evacuation time is lower than that of the uncontrolled scenario. However, the computational time of the predictive application is its main drawback.
# Contents

Acknowledgements

1 Introduction
   1.1 Traffic Evacuation Management ........................................ 2
      1.1.1 Statement of the Problem .................................. 2
   1.2 Proposed Solutions .................................................. 2
   1.3 Objectives and Contributions ........................................ 4
      1.3.1 Objectives .................................................. 4
      1.3.2 Contributions .............................................. 4
   1.4 Report Structure ..................................................... 5

2 Background
   2.1 Input Data .......................................................... 7
      2.1.1 Static Data .................................................. 7
      2.1.2 Dynamic Data ............................................... 8
   2.2 Traffic Network Representation .................................... 8
   2.3 Optimization Methods ................................................ 8
      2.3.1 Graph Based Methods ....................................... 9
      2.3.2 Shelter Destination Methods ................................. 10
      2.3.3 Demand Managing Methods .................................. 10
   2.4 Control Strategies .................................................. 11
      2.4.1 Actuated signals ............................................ 11
      2.4.2 Police officers ............................................. 11
      2.4.3 Multi-Agent Systems ...................................... 12
   2.5 Simulation Tools .................................................... 12
      2.5.1 Existing Simulators ........................................ 12
      2.5.2 Simulator Components ...................................... 13
         2.5.2.1 Coordination ......................................... 13
         2.5.2.2 Navigation ............................................ 13
         2.5.2.3 Traffic Behaviour .................................... 14
   2.6 Latent Force Model .................................................. 14
   2.7 Graphic Processing Units .......................................... 15

3 Methodology ............................................................... 17
   3.1 Functionality of the system ....................................... 17
   3.2 Data Acquisition .................................................... 18
   3.3 Manager ............................................................... 19
CONTENTS

3.4 User Interface ........................................... 21
3.5 Traffic Control ........................................... 21

4 Implementation ........................................... 23
4.1 Software .................................................. 23
4.2 Programming Model ..................................... 24
4.3 Data Source and Acquisition Module ..................... 25
4.4 Navigation System ....................................... 25
  4.4.1 Navigation System of Vehicles ....................... 25
  4.4.2 Police Navigation System ........................... 27
4.5 Traffic Behaviours ....................................... 27
4.6 Coordination ............................................. 29
4.7 Control .................................................... 29
  4.7.1 Description ........................................... 29
  4.7.2 Dynamic Control Approach .......................... 30
  4.7.3 Random Approach .................................... 32

5 Empirical Evaluation ....................................... 33
5.1 Environment With A Concentrated Distribution Of Buildings .... 33
  5.1.1 Traffic Distribution .................................. 34
  5.1.2 Evacuation Rate ..................................... 34
  5.1.3 Total Evacuation Time ............................... 35
5.2 Environment With A Spread Distribution of Buildings .............. 35
  5.2.1 Traffic Distribution .................................. 36
  5.2.2 Evacuation Rate ..................................... 36
  5.2.3 Total Evacuation Time ............................... 36
5.3 Execution Time of the Prediction .......................... 37

6 Discussion .................................................. 39
6.1 Discussion ................................................ 39
6.2 Future Work .............................................. 40
6.3 Conclusions .............................................. 40

7 Appendix ................................................... 41
Acknowledgements

I want to express my gratitude to God who has given me the opportunity to make real some of my dreams.

I would like to express my gratitude to my supervisor Dr. Sarvapali Ramchurn for his support, knowledge, and motivation.

I also would like to thank to Siddarta for his support.

My sincere thanks is also for the Ecuadorian people who has given me their motivation and financial support throuhg the SENESCYT. This opportunity will let me help to the development of my country.

Last but not least, my thanks are for my family in special my mother who has been with me every day.
Humanity has a deep relationship with the environment. Since the beginning, the human race has been dealing with disasters such as hurricanes, wildfires, and earthquakes. However, due to the increasing number of people moving from rural areas to cities, it is difficult to ignore that humankind is becoming more vulnerable to natural and man-made incidents. For instance, 2010 Chile earthquake affected 15,747 people per 100,000 inhabitants and the economical losses were 30 billion dollars [1].

Events like the Three Mile Island incident in Pennsylvania, World Trade Centre attacks in 2011, and Hurricane Katrina in 2005 may have led to the creation of the Disaster Management Research Field. Nevertheless, due to the some differences among these events, they have been classified as short-notice disasters and no-notice-disasters. According Wolshon et al [2], short-notice disasters such as hurricanes can be predicted at least 24 hours before they happen by utilizing tools like weather forecasting. On the other hand, the awareness time window for no-notice disasters might be null [3]. Therefore, taking decisions such as defining the evacuation routes, guiding traffic, and spreading awareness for mitigating losses in this scenario may be challenging due to the inter-dependent factors of the problem [4].

In recent years, disaster management is an increasingly important research area. As a result, projects such as ORCHID [5] and RocoCup Rescue [6] have been created for developing emergence response systems that integrates the emergency teams and evacuees. However, due to the period of time available and the complexity of the field, this master project is focussed on developing a control strategy for evacuating cars from an incident area.

After providing an overview of the evacuation domain, the statement of the problem is explained in the next section.
1.1 Traffic Evacuation Management

The emergency response for incidents might change from one to another. These may include the intervention of many teams such as policemen, firemen, and other rescue teams. Additionally, if the incident is severe, personnel from other councils could be needed. Consequently, planning and emergency management are utilized to address the emergency. Although planning tries to provide a solution for every type of scenario, the uncertainty of the domain makes planning inefficient. On the other hand, emergency management not only generates plans but also allocates personnel and equipment to solve the problem dynamically. This flexibility makes emergency management robust enough to address any change in the disaster environment [7]. As a result, traffic evacuation management involves generating evacuation paths, allocating officers for spreading information about the incident, and guiding vehicles.

1.1.1 Statement of the Problem

The growing size of cities makes the available resources insufficient to deal with disaster in areas of high density of traffic. Besides, limitations such as guidance mechanisms and lack of data makes difficult to generate a plan to guide evacuees toward safe zones. Consequently, a dynamic control strategy that takes in account available resources, requirements, and destination points has to be developed. Additionally, to test the effectiveness of the approach a simulation tool for evaluating large scale evacuations responses has to be implemented.

1.2 Proposed Solutions

A considerable amount of articles has been published to overcome this problem. Therefore according the optimization approach they have been classified into three approaches: Graphs based, shelter destination, and departure scheduling methods. Additionally, the control strategies depending on the mechanism to guide traffic are defined as actuated signals and police officers based solutions. Consequently, these models and control mechanisms are studied in this section.

Hamza et al. [8] suggest two graph based methods: All links and fastest links approach. Although the first solution utilizes all the roads for evacuating, its drawback is the lack of uniform distribution of traffic. On the other hand, Fastest links distributes cars among the fastest paths but this concentrates traffic in few roads which makes them more susceptible to another incidents such as a bombing attach. Consequently, [9] suggest the hybrid model which is a combination of the described approaches. Despite it takes the advantages of the models, it also has their main disadvantage which is the lack of a meeting place. These three solutions move cars toward any location outside of the
1. Introduction

evacuation zone. This might cause additional losses because seriously injured people may be guided further from hospitals.

To solve graph based approach limitations, shelter based models design plans that gathers people in specific points. For instance, [10] present the Reference Adaptive Control Model. The execution logic begins with the collection of data using the Monitoring System. Then, based on the Objectives, the Reference Model predicts the desired state of the network. After, obtaining the difference between the real and desire states, the Controller generates schemes (optimal splitting rate for traffic signals at intersections) to minimize their discrepancies. Finally, the Adjusting Mechanism applies the corrective strategies to the system. Additionally, to generate plans according current requirements, the cycle is repeated until the evacuation has finished. However, some of these models limitations are the requirement of real time information and the complexity of the information required. Consequently, in the worst disaster scenario information form sensors might not be available. Although officers could be sent to provide information while guiding the traffic, the complexity of estimating parameters such as turning flow rates at every intersection complicates the usability of these solutions.

Another perspective for minimizing traffic clearance time is to manage the injection rate of cars to roads. For example, [11] minimizes both network’s clearance time and injured people travelling time in order to decrease the number of deaths by scheduling the departure of evacuees. Nevertheless, the optimization of both parameters has a high computational cost which limits its application. Plus, Murray and Wolshon [12] claims that another drawback of this approach is that people may become selfish due to the survival instinct which makes difficult to convince them to stay in the evacuation zone until wounded victims have been moved to hospitals.

Tough the evacuation strategies defers in the optimized parameters, all of them require a system to guide traffic at junctions. As stated at the beginning of this section, these mechanisms could be police officers or actuated signals. Whereas, signals might handle adequately traffic, the main disadvantages of this approach are their prohibitive cost and power electricity dependency. In contrast, while police officers are cheaper, the limitations are the number of officers and their travelling time to arrive to junctions. Nevertheless, even though all the junctions in a city were controlled by actuated signals, they might be useless depending on the optimization strategy. Consequently, both have to be taken into account for designing evacuation responses.

To conclude, most of the studies have been focussed on finding the fastest evacuation paths, however little attention has been paid to combine available guidance systems and path generation models, efficiently.
1.3 Objectives and Contributions

Based on the gap stated in the previous section, the project aims to design an evacuation strategy that makes a powerful combination between the guidance system and the path generation optimization model in emergency scenarios. As a result, the research has two main objectives.

1.3.1 Objectives

The primary aim is to propose an algorithm that redirects the flow of cars for decreasing the network clearance time depending on the following criteria:

- Predicted traffic at junctions.
- Travelling time to reach destinations.
- Available police officers to guide traffic.

The second main objective is to create a software application that has the following features:

- Be designed as a Multi-agent system.
- Handle real world scenarios.
- Evaluate the proposed algorithm.

1.3.2 Contributions

The proposed strategy is a starting point for developing models that use predictions of queues at intersections to re-route evacuation paths. The implementation of this approach provides a predictive control at junctions instead of a corrective scheme. Consequently, the number of jams caused by the lack of police officers and the total evacuation time can be decreased because of the opportune re-route of traffic. Additionally, the software application based on multi-agent systems is a flexible tool to model real world scenarios and get metrics such as total evacuation time, percentage of evacuated cars during the simulation, number of cars at intersections, and the state of intersections in every time step of the process. These allow evaluating the performance of traffic management techniques.
1.4 Report Structure

This paper has been divided into six chapters. Chapter 2 begins by laying out the state of the art of the traffic evacuation management research field, and provides a critique of the advantages and disadvantages of the proposed strategies in terms of real world requirements. Consequently, it covers the areas of traffic network representation, optimization methods, and control strategies. Additionally, the simulation tools utilized to evaluate the strategies are examined taking into account their similarities with real world evacuations. Besides background information is presented at the end of this Chapter.

Chapter 3 states the design of both the simulator and the control strategy based on the objectives of the project. Consequently, it is organized according the general structure of the simulation system.

Chapter 4 examines the components of the control strategy and simulator. This chapter follows a similar organization as the previous chapter.

Chapter 5 present the results and evaluates the control strategy performance against the random re-routing and without any control technique. The parameters employed are those explained in the subsection 1.3.2.

The last chapter discusses the outcomes of the project according the performance of the model, presents the conclusions, and suggests the future work for exploiting this research.
Chapter 2

Background

After defining the aim of this research and its justification, the state of the art of the Traffic Evacuation Management (TEM) field and background information are described in this section. It begins analyzing the factors that affect the performance of the control strategies such as input data, optimization methods, and traffic control mechanisms. Then, the simulation tools and the approaches to implement their components are explained. At the end of this chapter, background information of both the proposed dynamic control strategy and the implemented simulator is presented.

2.1 Input Data

In order to assess and apply control strategies, the current state of the environment has to be feed to the simulator. The parameters required for the system varies depending on the goal of the research. However, they can be classified as static and dynamic information.

2.1.1 Static Data

The stationary information indicates the physical distribution of the traffic network such as location and dimensions of roads, buildings, and parks. As [13] states, the environment depends on the macroscopic or microscopic scope of the simulation. So, the scenario could be an intersection, road, or a town. For instance, Namewaka et. al [14][pp. 5] presents a microscopic simulator that employs information from GoogleMaps[15] to extract the configuration of the roads. However, the digital map is customized to differentiate important details such as roads line separation, and road traffic laws. On the other hand, many researchers such as [16] utilizes data from OpenStreetsMap[17] because it is accessible and ease to integrate to the system.
2.1.2 Dynamic Data

Although dynamic data changes with the passing of time, it can be gathered using sensors or surveillance systems. However, real-time communication may not be available during an emergency. Consequently, [8] present a software simulator that can also use average values for parameters such as speed of cars, density of traffic, and number of cars. On the other hand, Aved et. al [9] propose another method for worst case scenarios. This consists of police officers sending information through mobile devices to the operator who updates the parameters. Nevertheless, this approaches efficiency can be affected by the complexity of the information, which may not be easy of infer such as splitting rate at intersections, and the typing incorrect values.

2.2 Traffic Network Representation

The fundamental purpose of the TEM is minimizing the time required to move evacuees toward shelters or out of the evacuation zone. As a result, optimization methods are employed to find the fastest or shortest evacuation path. Nevertheless, before explaining the different approaches, an accurate representation of the traffic network is needed because the optimization phase performance depends on the accuracy and simplicity of this model[8].

Yue-ming Chen and Hui Geng [18] utilize a link-node representation, where a junction is illustrated as a node and a road as a link. [8] identifies each intersection by an ID number and its location \((x, y)\) in the network, and each link is described through its tail node ID and head node ID. Although simplicity is one of the advantages of this approach, it is not accurate due to the assumption that roads are always straight. Alternatively, [19] employs a vector representation, where crossings and irregularities of roads are represented by nodes. Though the accuracy is better than the first approach, it might be computationally expensive handling roads as vectors. Therefore, a hybrid version of both approaches is implemented in this project. The vector model is employed for drawing the traffic network and the node-link strategy is utilized for the optimization process.

2.3 Optimization Methods

Once explained the traffic network representation, the next step is describing the optimization methods. However, due to the variety of approaches they have been classified depending on their input data, destination points, and variables into Graph Based, Shelter Destination, and Demand Management models.
2.3.1 Graph Based Methods

Most of the data required by these methods to find the shortest path is static information such as length of streets, distribution of buildings, and location of incidents. As a result, a search tree is often employed to generate the optimal route. To provide a better insight of them this section describes three Graph based methods: All Links, Fastest Links, and hybrid approach.

Hamza et al.[8] propose and evaluates the first and second method. The Fastest Links employs a breadth first search tree to guarantee that all links are visited only once, where the root node is the incident, the leaf nodes are those located out of the circular evacuation zone that has the location of the disaster as centre and its radius depends on the incident type. Additionally, roads flow is constrained to only one direction because it decreases the conflictive points such as the number of crossings at intersections. The second graph based approach is the Fastest Links that is characterized by assigning an average speed to each segment. Plus, it takes into account roads’ capacity to calculate the path with the lowest travelling time. The search is performed using a multicast tree where the incident point is the source and the destinations are the receivers. Both approaches have advantages and disadvantages depending on the application. For instance, All Links approach could be simple because its model requires few input data for the breath first search. However, it may not be efficient for cities with high concentration of evacuees in small areas such as skyscrapers, stadiums, or coliseums because roads’ capacity is not taking into account. This can lead to bottlenecks in some of the routes and increase the losses and the evacuation time. Whereas, All Links method requires few data, Fastest Link needs an average speed per link and its capacity to calculate the fastest route. An advantage of this approach is that the evacuation time is lower than using the first model. Nevertheless, cars are distributed among few segments that makes the evacuation routes vulnerable to other incidents.

Based on the advantages and disadvantages of the previous solutions, [20] combined them to create the Hybrid approach that employs more segments than the Fastest Link and also takes into account the capacity of the roads. As a consequence, the traffic distribution is better. This makes the evacuation paths less susceptible to other incidents. Although the hybrid model takes the advantages of both approaches, their weaknesses are also part of the combined approach. For example evacuees are not move towards shelters which can increase the number of losses due to the lack of appropriate medical care. Besides, its computational time might be the highest of the three methods.

Summarizing, graph based models could be efficient for scenarios with uniform distribution of demand, few real time information, and with low probability of having seriously injured evacuees.
2.3.2 Shelter Destination Methods

Whereas, Graph Based Models might not need real time information, Shelter Destination approaches require data such as flow in the roads, number of cars at intersections, and speed of vehicles for their mathematical model of the optimization. For instance, Liu et al.\cite{10} proposes an adaptive control that is integrated by a Monitoring System, Adjustment Mechanism, Controller, Objectives, Reference Model, and Micro simulation. The purpose of the Reference Model is to predict the desire state of traffic based on data from the Monitoring System. These predictions are compared with the current conditions by the Controller to generate traffic control schemes to minimize the evacuation time subject to the mass-balance, flow conservation, and propagation constraints. At the end, the Adjustment Mechanism applies the corrective strategies. This is a closed loop process that aims to reduce the evacuation time. However, this approach is computationally expensive which limits its application for large scale scenarios. To overcome this disadvantage, \cite{21} has proposed a theoretical model that employs scheduling algorithms and routing heuristics. Nevertheless, the evaluation is missing which leaves the validation of its efficiency inconclusive.

On the other hand, \cite{18} intends to achieve the same goal using a dynamic control of the traffic flow. This optimization is based on the Pontryagin minimum principle that is focused in finding an optimal splitting rate at intersections that guarantees a low travelling time. Though the optimization is achieved, its main limitation is the high computational time. Consequently, \cite{22} and \cite{23} propose another approach to outperform the other methods. It decreases the delays at intersections through minimizing the number of conflictive points generated by the turning movements. In the same way, Parr and Kaisar \cite{24} intends to improve the traffic flow at junctions by applying dynamic programming to optimize the phase splits. However, some of them assume an ideal control mechanism at intersections.

In essence, despite the variety of advantages of these methods, their drawbacks are their strong dependence of real-time information, the costly equipment for controlling intersections, and their poor scalability.

2.3.3 Demand Managing Methods

Although previous approaches intend to guarantee a continuous flow on the roads, Demand Managing Methods proposed to regulate the injection rate of cars by employing a flow control strategy and using all the capacity of roads \cite{11}. The optimization strategy is scheduling evacuees’ departure time through their categorization according their injuries, where seriously injured people have the highest priority due to their health condition. However, departures of cars from different categories are allowed in order to exploit the capacity of the network. As a consequence, their algorithm looks for evacuate all the
refugees of the highest category as soon as possible while moving out as many victims as possible of the second highest class. Additionally, Chen and Zhang [25] based on the performance of staged and simultaneous evacuation strategies for traffic networks with different structures conclude that the best domains to scheduling departures are those that have high density of population and grid block structure. Nevertheless, Murray and Wolshon [12] state that this approach might be implausible because of the survival instinct of people. Commonly, Humans in danger situations are selfish and it might be complicated to convince them to stay in the evacuation zone until the highest category has been evacuated.

2.4 Control Strategies

Having explained the representations of the traffic network and the optimization methods, the next step is describe the control approaches to guide traffic out of the evacuation zone. As stated in subsection 2.3.2, most of the delays during evacuation are caused at intersections; therefore junctions are often managed by actuated signals, police officers and multi-agent systems. Both approaches are examined in order to understand their advantages and disadvantages.

2.4.1 Actuated signals

For daily control of intersections pre-timed signals are employed and sometimes they might be employed for evacuation using a peak hour set up. However, as [24] suggests they may be inefficient in a disaster framework because they enable the traffic to links that are empty which increases the evacuation time. This weakness is the result of the lack of feedback of the current state of roads. To overcome this limitation, recent evidence suggests that actuated signals have a better performance because they receive real-time information through sensors or surveillance systems [8][18]. As a consequence, the optimization parameter is the splitting time. Nevertheless, the key problem with this approach is that actuated signals need electricity to receive data, process information, and guide traffic; and electric power might not be available in the worst case scenario. Another disadvantage is the prohibitive cost of acquiring and installing them in the traffic network. In summary, although this approach provides better outcomes than pre-timed signals, its cost and electrical power dependence limits its application domain.

2.4.2 Police officers

Whereas actuated signals are expensive, police officers are cheaper. Nevertheless, they are a limited resource whose allocation has to be optimized in order to achieve the best
outcomes. Liu et. al [10] highlights that intersections should be controlled by officers because drivers are more likely to follow their guidance. Therefore, Aved et. al [9] proposes a voting algorithm to prioritize junctions that depends on the number of legs and its Euclidean distance to the incident. However, another drawback is the lack of control due to the mobilization of policeman to their destination points which can lead to bottlenecks and jams in the evacuation routes.

2.4.3 Multi-Agent Systems

Although VanMiddlesworth et al. suggest a daily usage agent-based model control for intersections, it can also be applied during evacuation. The control system is based on the assumption that every junction has a control mechanism that can access the information of the vehicle before arriving to the crossing. Parameters such as speed, traffic flow, and distance are employed to accept or deny the arrival of the car because the goal of the control mechanism is to ensure a maximal flow through all the links of the intersection, simultaneously [26]. Despite the advantages, under the current circumstances it is difficult to implement this model due to the lack of infrastructure and newest technology in both the traffic network and cars.

2.5 Simulation Tools

After constructing the optimization models, they have to be evaluated. As a result, this section describes some of the simulators utilized to test models and their main components. As a result, coordination, navigation, and traffic behaviour approaches for modelling traffic scenarios are presented in the second subsection. The source of data is omitted because it was described in Section 2.1.

2.5.1 Existing Simulators

A considerable amount of researchers prefer to employ simulation tools to test the proposed approaches. [3] argue that the main reason for this preference is that simulators allow a detail study of the outcomes and performance of the implemented evacuation procedures in a safety manner. As a consequence, these tools have been developed depending on the variables that have been optimized. To illustrate, [27] suggests a behaviour model that is assessed using TransWorld that is a multi-agent traffic simulation system. In contrast, Hamza et. al [8] and Aved et al. [9] prefer the TSIS (Transportation System Integrated Software) that is based on the CORSIM simulator which emulates traffic conditions in urban areas. However, some of the models are not scalable because its computational time grows with the size of the simulation area or number of agents.
On the other hand, [19] offers a solution for crowd evacuation management that is based on a graphics processing unit which employs parallelism to simulate around 7000 agents. This approach could be employed for the traffic evacuation field in order to build more powerful simulation tools.

### 2.5.2 Simulator Components

#### 2.5.2.1 Coordination

In emergency scenarios, the spread of awareness could be done by mobile devices, media, and police officers. However, sometimes the only available communication systems are officers and other evacuees because the other communication systems may be affected by the incident. According to Long [19], due to the limited number of policemen and the number of possible destinations it is a constraint optimization problem. This can be optimized using genetic algorithms such as hill climber where the length of the genome is equal to the number of available officers and the genes are any possible destination point. However it can get stuck in a local optima solution. Another solution proposed by [28] is the Distributed Stochastic Algorithm. The main search loop consists in generating solutions through random methods and randomly corrected them if they violate the constraints. Although it is simple, it is not optimal because it may fall in infinite loops.

#### 2.5.2.2 Navigation

The function of the navigation module is to move cars through roads until they reach their destinations. Consequently, the path finding algorithms utilized to define the routes are explained in this subsection.

As mentioned in the section 2.2, the general representation of the traffic network is the link-node approach. As a result most the navigation systems are based in a tree search. For instance [8] employs a breadth first tree, where the root nodes are the incidents, to find the evacuation path and guarantee that the intersections are visited only once. However, the distribution of the traffic is poor, which may cause congestion in some of the roads. On the other hand, in the same article a multicast tree is presented to find the fastest routes. Consequently, it takes into account the average speed allowed in the link and its capacity. Nevertheless, traffic is concentrated in few roads which make them more vulnerable. On the contrary, [27] find the optimal evacuation routes applying the Dijkstra algorithm. For this search the root is the current position of the vehicle and the goal destination is any node in the safe zone. In every iteration the closest intersection is expanded. The expansion consist in verifying if it is the destination and add its adjacent nodes to the search if it is not. Although this solution is optimal, its disadvantage is that memory scales with the number of vehicles.
2.5.2.3 Traffic Behaviour

According [29] traffic behaviour includes free driving, passing another cars, and car following. These depend of many factors such as type of vehicle, state of the road, and driver behaviour. Consequently, the large number of parameters and the data available make difficult the simulation. As a result, to decrease the complexity, some researchers prefer to evaluate the models using microsimulations that constraint some of the parameters of the domain. Therefore, some of the presented algorithms are for freeway traffic and they are based on the NaSch model [30]. This approach splits the roads into cells of length equal to 7.5m which is the safe distance for calculating the speed. Therefore, the basic behaviour is achieved applying the following conditions: if the speed is lower than the desired velocity, the agent has to accelerate; if there is a probability of crashing, the breaks are pressed; otherwise do nothing. On the other hand, [29] proposes a model that utilizes a neural network to model additional behaviours such as turning to the left or right. As a consequence, a binary combination of layer weights is employed to control the car.

To conclude, the field of evacuation management research needs a software simulator that overcomes the difficulties of traffic evacuation modelling in urban areas. Its features should be: its performance has to be independent of the dimensions of the simulated area; the format to input the data has to be friendly with the users, the traffic network should be dynamic; and, the output parameters has to be significant to evaluate the control strategy proposed.

In the following sections, backgorund information for the control strategy and simulator is provided.

2.6 Latent Force Model

Under emergency scenarios caused by hurricanes, tornados, or manmade disasters, the uncertainty of the reliability and availability of real time information and communications systems makes taking decisions difficult. To date various research projects such as ORCHID and RoboCup Rescue have been focussed on developing reliable and robust systems for emergence response. However, in the Traffic Evacuation field remains the challenge of defining a reliable model that requires little or simple real time information. This problem can be resolved by utilizing a queue predictor developed by [31]. Although it was originally created for a call centre application, it has been modified to estimate length of queues at intersections. Consequently, it is explained in this section.

As Reece et. al claim the requirement to use this model is that the system could be reduced to the form of the equation 2.1 [31][eq. 6]. Consequently, this application consists of two phases: the definition of the Latent Force Model of the system and the length queue prediction using the extended Kalman filter.
\[
\frac{dx(t)}{dt} = Dx(t) + Bw(t)
\] (2.1)

Where, \(x(t)\) represents the vector of variables, \(B\) and \(D\) are non-random coefficients matrices, and \(w(t)\) symbolizes the latent forces vector.

Because of the applications’ domain, its representation can be based on queue’s Pointwise Stationary Fluid Flow Approximation model and the customers arrival dynamics. This is justified because the injection of clients to the system can be defined as M/M/1 due to the fact that they are served in the same order as they arrive and only one customer can be attended at a time. The representation of the process is illustrated by the equation 2.2 [31][eq. 68].

\[
\frac{dL}{dt} = -K(t) \left( \frac{L(t)}{1+L(t)} \right) + W(t)
\] (2.2)

Where \(L\) represents the mean queue length, \(W(t)\) is the customers injection rate which is assumed as a Gaussian process, and \(K(t)\) is the mean service rate. However, due to the non-linearity of the model, the prediction is based on the extended Kalman filter. Additionally, Reece et. al assume that the mean value of the length is a good approximation of its instantaneous values before calculating the estimated values of the variable and the covariance.

Finally, based on the estimates generated for the application, police officers can be deployed to redirect traffic before the jams are originated at intersections. Consequently, there is an improvement in the distribution of cars on the network which may decrease the evacuation time.

### 2.7 Graphic Processing Units

To date various technologies have been applied to develop simulators. However, Graphics Processing Units (GPU) are becoming more popular because the processing time and their affordable price. Although, programs run faster, GPU’s storage capacity is still a restriction for its utilization. Nevertheless, a CPU + GPU combination facilitates the software development due to pieces of the program can run in the parallel units while others are running in CPU [32]. This property improves the scaling of the application which is one of the goals of this project. Consequently, the architecture of this device is presented.

Nvidia GPUs’ hardware and software architectures are bases in similar principles. For example, the FERMI architecture of GEFORCE cards contains 512 streaming multiprocessors, this number depends on the card, grouped in sets of 32 cores called warps that consist of 16 SM each (SM), and these cores can execute independently a thread [33][pp. 16].

Similarly, the programming architecture CUDA follows an analogous methodology. The
process begins when a program invokes parallel kernels. Then, the GPU instantiates every kernel on a grid which is a set of blocks. At the end, every block runs an instance of the kernel in each thread that is differentiated by its ID. In other words this process resembles the parallelism of the GPU. Additionally, to improve the running speed, cards have a hierarchical structured memory shown in Figure 2.1. This means that each tread has its own local memory, but blocks in the same grid have shared memory, while the global memory is distributed among grids [34][pp.6]. An efficient allocation of the shared memory guarantees that a process running time can be decreased due to this architecture allows that thousands of threads can be running at the same time.

Figure 2.1: CPU - GPU device data transfer[35][pp. 22].
Chapter 3

Methodology

This chapter describes the architecture of the simulation engine and the control strategy to solve the gaps introduced in chapters 1 and 2. Consequently, their functionality, requirements, and structure are explained following the execution logic of the simulator.

3.1 Functionality of the system

The purpose of the system is to emulate the behaviour of drivers in urban areas during an evacuation and estimate the performance of different control strategies in a safety manner. The structure of the simulator illustrated in Figure 3.1 is divided into three modules: Data Acquisition, Manager, and User Interface.

![Figure 3.1: Structure of the software simulator](image)

The process begins with the collection of information from the environment, such as distribution of roads, number of buildings, and number of evacuees. This constitutes the bases for representing the navigation map which is utilized for the other components. While, the data acquisition is executed only once, the Manager has to operate constantly because it includes the behaviour, coordination, navigation subsystems, and the traffic control. Moreover, this second module is linked to the Interface which has three functions: interact with the user, displaying the simulation, and showing the results. Its purpose is to obtain information, monitor the current state of the environment, and present the evolution of the main parameters such as cars arriving rate to shelters, total evacuation time, and state of intersections through the passing of time.

Having explained in general terms the functionality of each module, their detailed description is given in the following sections.
3.3 Data Acquisition

Two types of data are required for the system. The first constitutes the settings introduced by an operator through the user interface and the other is the description of the environment. This section is focussed on the second kind because its acquisition not only involves the integration process but also includes the selection of source of information. This might facilitate or make difficult the extraction and assimilation of the data by the system. Consequently, the required features and origins of data have to be analyzed.

Datum is one of the fundamental components of the simulator’s architecture. It emphasizes the spatial relation among buildings, intersections, roads and their surroundings. Additionally, the usability of the system depends on having a wide an accessible data source. As presented in the Chapter 1 the aim of this project is to develop a simulator for real world environments. As a consequence, physical traffic networks needs a workable representation for computers.

The input information format that is becoming widely accepted in the simulation field is digital maps. One of the reasons might be the growing number of organizations such as EDINA, Open Street Maps, and Google Maps that gather and make data available for researching purposes. Additionally, the geographical information can be handled and accessed easily by a computer which simplifies the engineering of data acquisition modules.

According to [36] digital maps are simple to manage due to its vector data model which utilizes point, lines, and polygons to represents objects of the environment such as location of petrol stations, roads, and buildings. Its minimum unit of data are points which contain the spatial information such as latitude and longitude. Consequently, roads and buildings are expressed as vectors of points. The differences between these two types are the first and last point which are different for a line and the same for an area. Besides, to facilitate the analysis, data is layered according the type of object and attributes are assigned to each vector. These features are shown in the Figure 3.2.

After defining the source of information and gathering the data, this has to be integrated to the software simulator. The modelling of the data has to be simple and reusable in order to minimize the memory space required for the environment representation. These requirements are some of the features of the object model representation. Therefore, this approach could be applied to the simulation application.

To conclude, data acquisition module includes integrating the information to the system and defining the data type required. For the current project digital maps are the environmental information source because they are accessible, simple, and portable for a computing system.
3. Methodology

19

(a) Point

```xml
<node id="2679899908" lat="51.3169994" lon="-0.5669299">
  <tag k="amenity" v="fuel"/>
  <tag k="name" v="Horrison"/>
</node>
```

(b) Line

```xml
<way id="150560087" >
  <nd ref="1435888417"/>
  <nd ref="1632598376"/>
  <nd ref="1632932707"/>
  <tag k="highway" v="unclassified"/>
  <tag k="lon_ref" v="1"/>
  <tag k="maxspeed" v="50 mph"/>
  <tag k="name" v="Chesney Road"/>
</way>
```

(c) Area

```xml
<way id="1699982236" >
  <nd ref="401514548"/>
  <nd ref="401514502"/>
  <nd ref="401514517"/>
  <nd ref="401514517"/>
  <nd ref="401514518"/>
  <nd ref="401514519"/>
  <tag k="addr:city" v="Koking"/>
  <tag k="addr:country" v="GSP"/>
  <tag k="addr:street" v="York Road"/>
  <tag k="building" v="house"/>
</way>
```

Figure 3.2: Vector representation of the basic forms utilized for digital maps for representing points, streets, and buildings [36].

3.3 Manager

As Figure 3.1 shows, the next component in the architecture is the Manager. This constitutes the engine of the process because it encloses the logic to update the simulation in every time step. Additionally, due to the agent based modelling approach of this project, this module has to manage the navigation, coordination, traffic control, and behaviour subtasks. To support the importance of the functionality of the Manager, Figure 3.3 illustrates its methods. Each sub process has a fundamental role in the performance of this component. The operation begins with the coordination. Its functions are spread the awareness, determine the best destinations for police officers based on a utility function, and allocate them. Once, the evacuees’ alert state has been updated, the Traffic Control aims to improve the traffic distribution at intersections to decrease vehicles’ travelling time. Although this subtask will be explained further in the section 3.5, its logic structure can be described as updating its internal state of the traffic network, generating new paths based on the new information, allocating officers to guide vehicles, and updating the navigation map of vehicles. After allocating the
destination of police officers, the Navigation system creates their global path, defines their way points, and calculates their final velocity. Nevertheless, because of the number of evacuees, their navigation system has to be independent of the quantity of vehicles. Consequently, it should renovate their destination based on their latest position. The last but not least component is the behaviour management. It intends to solve the crashing problem and emulate the break an accelerate drivers’ attitudes. Therefore, this sub process comprehend calculate the speed, apply drivers behaviour, and update the final speed of each agent. However, due to the computational time and the deep relation between the Manager and the User interface, the described tasks are distributed in such a way that the visualization and performance of the system remain unaffected.

Summarizing, the second module updates the state of the simulation for every time step, contains the methods for the coordination, navigation, control, and behaviour of the cars, and it has strong relationship with the visualization because of the computational time of each sub process.
3. Methodology

3.4 User Interface

Users always have an objective when they are interacting with an interface. Therefore, the usability of an application depends on the extent to which their goals were completed efficiently, satisfactorily, and effectively. Consequently, this has to be intuitive and simple to use as possible to guarantee a straightforward relation between the system and the operator. According [19] the simulation process has to be split into the start, settings, environment, simulation, and results stages that can be accessed through tabs; these should be organized in a chronological order and a new tab might be unlocked when all the conditions of the previous step have been accomplished; additionally, the visual representation of objects and their location could be updated by Windows Presentation Foundation drawing visual class and rendering pipeline, respectively.

3.5 Traffic Control

As stated in the Chapter 2, some of the limitations of the up to date control strategies are the budget, lack of flexibility, power electric dependency, delays, and dynamics of the problem. Whereas actuated signals guiding implementations time is minimum, their disadvantages are their price, power electric dependency, and installation in every intersection of the network. On the other hand, although police officers approach is cheap, power electric independent, and relocatable, the delay between the guiding instruction and its execution may cause bottlenecks and traffic in the roads. Therefore, there might be two solutions to this problem. The first suggestion is to design a low cost, and efficient actuated signals, but this is not the objective of this project. The second solution is to designate officers to control and define a method to predict traffic at intersections. These might eliminate the drawbacks of current approaches which are focused on correcting the problem instead of preventing it. As a consequence, a predictive methodology illustrated in the Figure 3.1 is going to be explained.

The process begins with the prediction stage. The requirements for this phase are that it should be as independent as possible of the real time information from sensors or the surveillance system, the time interval between the present and the forecast must be enough to deploy officers, and the output data has to reflect the future states. Consequently, the model proposed by [31] to infer length of queues for call centres is applied. Nevertheless, for the simulator, the model is going to predict the length of queues at intersections depending on the flow direction of the roads. This model is presented in Section 2.6.

Having inferred the future state of traffic at intersections, junctions with the longest predicted queues have to be rerouted. However, the rerouting process is subject to the number of available police officers and the future cost of keeping the same path. After defining the new paths, the strategy is applied. This step consists of two stages:
resource allocation and redirecting vehicles. Firstly, officers have to be allocated depending on their travelling time to reach the new locations. Then, when the rerouting team has arrived to its destination the flow of the path has to be changed. This process should remain or finish based on the traffic rates inferred for a new prediction. Summarizing, with the execution of the prediction, path generation, and reroute application methods the control strategy should be able to improve the traffic distribution at junctions and minimize the total evacuation time.
Chapter 4

Implementation

Once described the functionality of each component of the application, the next step is to explain the techniques and tools employed to implement the tasks and subtasks detailed in Chapter 3. Therefore, this chapter begins defining the software package and describing the programming model of the simulator. Then, the source of information for the digital maps and the acquisition module are analyzed. Next, the algorithms for navigation and coordination of police officers and vehicles are outlined, and in the last section the behaviour and control strategies applied to redistribute the traffic at intersections are examined.

4.1 Software

The software was chosen based on the time constraints, previous applications, hardware, and project objectives. As stated in Section 2.7 a GEFORCE NVIDIA graphic card was selected to implement the simulator (the reasons to make this decision are given in the Section 4.2). Consequently, CUDA toolkit 5.5 was employed as the computing platform to exploit the advantages of the GPU. However, it is a low level language which makes difficult programming. As a result, CUDAfy.NET V 1.25 was combined with CUDA to provide flexibility and simplicity to the parallel programming model. Besides, Microsoft Visual Studio Ultimate 2012 Version 11.0 Update 3 was the development environment because it is compatible with the newest Windows8 platform and allows utilizing C# 2012 Microsoft .Net Framework 4.5 as a programming language. Although the lack of experience working with object programs, C# was preferred due to the availability of a crowd evacuation application software /citeRLJDisertation and the advantages of the object oriented programming. This environment granted a simple software maintainability, faster development, and reusability of classes.
4.2 Programming Model

It was decided that the best technologies for the simulator are the Nvidia Graphics Processing Units because its GPU computing allows separating the architecture of the application into serial and parallel subprograms. This improves the speed of the software which implies that more subtasks described in Subsection 3.3 can be executed in a time step. Moreover, applications such as the simulator developed by [19] for crowd modelling in urban areas, the path-finding algorithm implemented by MacNally [16], and the large scale crowd simulation for buildings presented by [37] demonstrate the advantages of employing this technology. Besides, it is accessible to the students of the university because the desktop computers of the laboratories have these cards. Thus, CUDAfy libraries and tools provide flexibility and simplicity to the coding [38]. Consequently, the programming model required to exploit the benefits of the GPUS has the following structure.

Every function requires input data. As a consequence, the information that was handled in both GPU and CPU was defined in the host and in the device, respectively. This initialization was performed before calling the kernels to guarantee that the memory was allocated for every array. Therefore, the functions employed to pass the information from the host to the device and vice versa were \( \text{dev}_n \text{ < variable name > } = \text{gpu.CopyToDevice(host}_n \text{ < variable name > )} \) and \( \text{gpu.CopyFromDevice(dev}_n \text{ < name > , host}_n \text{ < name_of_the_variable > } ) \), respectively. Nevertheless, the variables managed only by the GPU were instantiated by the \( \text{dev}_n \text{ < variable name > } = \text{gpu.Allocate < data_type > (length_of_the_array) [19].} \)

After loading the data, the number of threads and blocks needed for the method was defined. For this application, the number of threads was equal to the quantity of evacuees, police officers, or intersection. As a result, the quantity of blocks was the top integer of the number of threads requires divided by 512 (maximum number of threads per blocs). These threads were launched by the \( \text{gpu.Launch(blocks,threads). < name_of_the_method > (variables) function.} \)

Thus, to avoid the execution of unused threads, they were limited within the method. Before executing the threads, their ID was defined utilizing the code utilized by [38]. To illustrate, Figure 4.1 exhibits the code for a method employing Cuda.

```
[Cuda]

public static void NumberOfCars(GEThread thread, int[] cars, int[] areaBuildings, int sizeCar)
{
    int threadId = thread.blockIdx.x * thread blockDim.x + thread threadIdx.x;
    if (threadId < areaBuildings.GetLength(0))
    {
        cars[threadId] = (int) areaBuildings[threadId] / sizeCar;
    }
}
```

Figure 4.1: Cudafy Programming model.
4.3 Data Source and Acquisition Module

This section begins with the description of the source of information of the application and ends detailing the logic of the implemented acquisition module. The Open Street Maps Foundation\cite{17} was chosen as the data source of the application because it satisfied the features described in the Section 3.2. It was accessible because the maps were downloaded from the webpage of the organization for free. Thus, it was wide due to the maps of most of the urban areas of UK were available. Besides, the data of the .osm files were organized in a vector format which simplified the functionality of the simulators data acquisition module.

The data acquisition implementation was taken from the simulator developed by \cite{19} due to the fact that it maintained the operability of the information because it integrates the information using an object modelling representation approach; additionally, it includes the spatial data conversion function that compensates the earth curvature.

4.4 Navigation System

Although vehicles and officers were in the same environment, their opposite roles during evacuation scenarios led to the implementation of two navigation approaches. Consequently, the description of the path finding algorithms begins explaining vehicles navigation system and ends presenting the method applied for police officers.

4.4.1 Navigation System of Vehicles

As affirmed in Chapter 2 some of the drawbacks of the optimization methods are the memory scaling with the number of vehicles and undefined destination points out of the evacuation zone. To avoid this limitation, the construction of the navigation was done applying the Algorithm 1 which was divided according the type of node.

The first stage built the navigation grid for the main nodes. It was based on a breadth first three which had the entry nodes to shelters as root nodes. Thus, the search was constrained to the length of the path to find the shortest ones. To guarantee that every node was set up only once, the solution was represented as a vector with length equal to the number of nodes. Therefore, in the first for loop the shelters nodes navigation point were set up to a negative number according their position in the set of shelters. These constituted the roots of the tree. Then, in the inner loop the navigation points of the adjacent nodes to shelters were changed to the ID of the shelter nodes; the distance between them was saved in the Length vector; and, their children nodes were added to the Child List. Next, the while loop repeated the same logic as the previous process plus
the length constraint. This process was done until there are not more nodes to expand. 
As a result, even though the navigation point of the node had been set up previously, it 
was changed if a new path was shorter.

**Algorithm 1** Algorithm implemented to find the navigation points of vehicles. 

**Data:** SetOfNodes, SetOfExitNodes, SetOfShelters, NumberOfShelters 

**Result:** NavVector

**initialization**

for int i=0; i<NumberOfShelters; i++ do

  NavVector[Shelter.NodeID]=-i-1
  Length[Shelter.NodeID]=0

  for each node in Shelter.ChildList do
    NavVector[node.ID]=Shelter.NodeID
    distance= EuclideanDistance(Shelter,node);
    Length[node.ID]=distance
    ChildList.Add(nodeID)
  end

end

while ChildList.Length>0 do

  for each node in ChildList do
    for each child in node.ChildList do
      distance=EuclideanDistance(child,node)
      if Length[child.ID]==0 or Length[child.ID]>distance then
        NavVector[child.ID]=node.ID
        Length[child.ID]=distance+Length[node.ID]
        NewChildList.Add(child.ID)
      end
    end
  end
  ChildList=NewChildList
  NewChildList.Clear()
end

**initialization ExitNodes**

for each eNode in SetOfExitNodes do

  if NavVector[eNode.ClosestN]==NavVector[eNode.SecondN] then
    NavVector[eNode.ID]=NavVector[eNode.SecondN]
  end
end
The second phase began with the initialization of the exit points of the buildings. This step consisted in instantiating their navigation vector to their closest node. Then, it was replaced if the other node used to calculate the exit point was closer to the shelter. At the end of the execution of the algorithm the output was the navigation vector with the navigation point for each node. Although it was computationally more expensive than the Vector Field Approach suggested by [19], it was executed only when the shelters where defined. Consequently, the time complexity of the main loop of the simulator was unaffected by this algorithm. However, it limited the flexibility of the destination points of cars because the next point was the same for all the agents that arrive at the same intersection.

4.4.2 Police Navigation System

Whereas, paths for cars converged to shelters, police destination points were dynamic because officers had two functions: spread awareness and re-route traffic at intersections. Consequently, the path finding algorithm designed by [19] was integrated because it was based on the $A^*$ search with the Euclidean distance as the heuristic. This method is known for being optimal and complete. Although the algorithm allowed dynamic destinations, the memory required was its drawback. As a result, it was only applied for officers because their quantity was much lower than that of cars. The path finding algorithm is detailed in the Appendix 7.

4.5 Traffic Behaviours

After generating the navigation vector, the remaining obstacles were other cars. Consequently, to follow the logic described in the subsection 2.5.2.3 the algorithm 2 was implemented to modify the speed. This was changed depending on the distance with other vehicles. Although, [30] fixed the safety distance to 7.8 meters, the equation 4.1 proposed by [29] was utilized to calculate it because the equation reflected the relation between velocity and distance.
Algorithm 2 Traffic Behaviour Algorithm.

**Data:** SetOfActiveCars, Threshold, matchingSpeedFactor

**Result:** Velocity

initialization

Calculate safetyDistance

for each neighbour in SetOfActiveCars do

    distance=EuclideanDistance(car, neighbour)

    if distance < Threshold then
        neighboursSpeed+=neighbour.speed
        counter++
    end

    if distance < safetyDistance then
        if neighbour in front? then
            decelerate
        end
        carsInTheFront=true
    end
end

if !carsInTheFront & Velocity<car.desiredVelocity then
    accelerate
end

if counter > 0 then
    Velocity+=matchingSpeedFactor*neighboursSpeed/counter
end

This algorithm was executed in the parallel system where each thread represented a car that was out of the shelters. Consequently, the initialization phase corresponded to calculating the number of threads and blocks required for the CUDAfy method. This process was showed in the Figure 4.1. Then, the variables were instantiated and the safety speed was calculated using Equation 4.1 \[^{29}\] eq. 6. Next, for simulating the behaviour of cars when roads where almost saturated, a matching speed phase was created. This was obtained by averaging the speed of the cars that were closer than a threshold. Therefore, the first and last If loops were utilized for getting the average and applying it to the speed of the car. Then, the accelerating and breaking behaviour adjusted the velocity based on the safety distance. For instance, if a car was in front and the distance car-neighbour was lower than the safety parameter, the speed was reduced. Conversely, if there was nobody in the front, the vehicle increased the velocity unless it was equal or higher than the maximum speed. This avoided the crashing of cars at intersections.

\[
\text{Safety Distance} = -0.0059v^2 + 0.187v + 0.5 \quad (4.1)
\]

where the speed was in km/hour.
4.6 Coordination

This section of the simulator utilized the Awareness Class Hill Climber algorithm approach form [19]. Consequently, the set up and process description of the method are explained.

This class defined the parameters as follows: the chromosome representation as a vector of integers that had a length equal to the number of police officers for spread awareness; every gene was the destination of a police officer and the alleles were the nodes ID of the traffic network; the operator utilized was the mutation; the fitness contribution of each variable was the number of cars that were within a certain range from the node and whose awareness attribute $a \in [0, 100]$ was $< 100$; the fitness function was the total number of cars that could become aware if the officers were assigned to the nodes of the genome; the number of generation was 20; and, the population size was 10.

The process began initializing the population. Then, they were evaluated using the fitness function. After creating a new generation applying mutation to the fittest individual, the old generation was replaced by the new population. Next, the evaluation and reproductions were repeated for 10 generation. At the end, the best individual of the last population was chosen.

As a result, police were allocated to the best genome destinations.

4.7 Control

This component is focused on the guidance of traffic at intersections during the evacuation. In a real scenario, drivers tend to use the shortest paths from its current location to the closest shelter. However, if the evacuation paths are chosen using these criteria, jams may generate at intersections depending on the number of cars in the network. Rerouting is a common mitigation strategy but its effectiveness depends on the strategy. Consequently, its basis and two approaches which were implemented are explained in the following subsections.

4.7.1 Description

The objective of redirecting vehicles during evacuation is to decrease the network clearance time by avoiding congested roads. However, due to the number of junctions in urban areas, limited number of police for guiding, and the uncertainty of the problem, it is complex to define a model that provides an initial reference to managers for taking decisions.
4.7.2 Dynamic Control Approach

In order to select and allocate officers, the dynamic approach predicts the future state of junctions at intersections and evaluates the cost in terms of travelling time of redirect them. Consequently, the principle of this approach is to decrease the number of jams by eliminating them before they are formed without increasing the network clearance time. The steps followed to achieve this aim are illustrated in Algorithm 3 and are expanded after this.

Algorithm 3 Dynamic Control Algorithm.

Data: SetOfIntersections, OfficersAvailable, setOfShelters, timeStep, minTimeStep

Result: LowerEvacuationTime

Classify Intersections

while carsInTheRoads ≠ 0 & timeStep ≥ minTimeStep do
  timeStep = 0
  Find Paths
  Predict Traffic
  Calculate SetOfNewPaths
  for each newPath in SetOfNewPaths do
    Calculate pathCost
    Apply Heuristic
    if newPathTravelingTime < currentPathTravelingTime then
      Stop previousRerouting
      Apply Rerouting
    end
  end
end

Because of the relatively large number of intersections in urban areas these where classified according the number of buildings contained by their legs. To avoid any ambiguity in the selection, k-means clustering was utilized because intersections are classified according a mean value. Consequently, the set with the highest mean was selected as the basis for the search.

After classifying the junctions, the paths that include them were founded. This was performed utilizing a recursive algorithm that expands the routes until reach a shelter or a previous visited intersection.

Once generated the actual routes, they were submitted to the Latent Force Model Application developed by [31] in Matlab. This based on the features of roads such as direction of the flow, existence of buildings, and state of their header intersection estimates the queue length at junctions for the next 2 minutes. To integrate the Matlab code into the
4. Implementation

C project, the approach applied was that of using Matlab as a server from C♯. As a result, both programs where debugged simultaneously.

The feedback of the prediction stage was the length of queues. Consequently, nodes were reorganized according this parameter for finding the new paths. This guaranteed that crowded intersection had more officers available for redirecting the flow. Additionally, to improve the traffic distribution the redirection began from the most crowded adjacent intersection which assures that part of the traffic remained using the most crowded predicted junction. Then, neighbouring junctions with the smallest queue were used to expand the path. Plus, infinite loops (driving in circles around the same square) and legs from the crowded junction were avoided.

Before redirecting traffic, the new set was evaluated to guarantee that the evacuation time was lower than using the old configuration by calculating a penalty and a heuristic. The cost of travelling using the intersection $i$ was defined as the relation $\frac{\lambda_i}{u_i}$ presented in the equation 4.2, where $u_i$ is the service rate and $\lambda_i$ is the arrival rate of the junction. This formula was based on the length of the queue equation proposed by [39][eq. 7.15].

On the other hand, the travelling time from any intersection to the closest shelter was determined by the 4.3, where $l_i$ is the heuristic, which is its Euclidean distance to the closest shelter, and $v_i$ is the maximum allowed velocity in urban areas. Therefore, for the new path $l_i$ was the length of the new path plus the heuristic.

\[
\frac{\lambda_i}{u_i} = 0.5(L_i + \sqrt{5L_i}) \quad (4.2)
\]

\[
t'_i = \frac{l_i}{v_i} + \frac{\lambda_i}{u_i} \quad (4.3)
\]

After calculating the travelling times, the flow is changed only if the new path is fastest. Consequently, Stop Rerouting function consist on reallocating officers that were guiding intersections affected by the new routes. Next, the Apply Rerouting step allocates policeman to each path which flow is deviated when all the officers arrive to their destinations.

Then, the process was repeated from the junction configuration updating until the network is empty. This assured that traffic was distributed toward roads with the smallest queue.

The time complexity of the algorithm is evaluated in terms of their sub processes. Consequently, generating the paths implies searching iteratively the next navigation point until a shelter or a previous expanded node is founded. This might require $n$ steps, where $n$ is the number of intersections with more buildings in their legs times the maximum length of the path. This parameter was defined to limit the search in large scenarios. After, finding the paths these have to be arranged according the requirements of the Matlab application. So this represents $n$ steps. When the feedback from the predictive module is received it is organized from the most congested intersection to the least crowd
junction. This needs also $n$ steps. However, generating the new routes and evaluate them costs $nd$ where $d$ is the maximum number of police for rerouting a path. To guarantee that the most crowded junctions have priority, policemen are allocated employing the greedy approach. Nevertheless, this phase only costs $q$, which is the number of available police officers, because junctions were organized after updating the predictions. Finally, the rerouting and stop rerouting phases have a computationally complexity of $r$ which is the number of intersections affected by the new paths. Summarizing the number of steps required execute the algorithm is $4n + nd + q + r$. As a consequence, because of the visualization the execution of this method has been subdivided in the phases described before. To evaluate the time complexity of the Matlab application integrated as a server of the project, the length of the time steps will be measured.

4.7.3 Random Approach

An alternative criterion to select the junctions was the stochastic strategy. This follows the same logic as the previous approach. The difference was that junctions were picked randomly for being redirected. Although, the data of the current state of the network required for this part of the algorithm was null, the strategy stills need to ensure that paths were not forming infinite loops. At the beginning, this approach didnt contemplate this restriction; consequently, the evacuation time was increased more than 10 times those needed to evacuate without any guidance. So, it was decided to implement the cycling constraint.
Chapter 5

Empirical Evaluation

For evaluating the dynamic control strategy, the traffic evacuation simulator developed was utilized. Additionally, the test was performed on a 64-bit Windows 8 machine that has a processor i5 3230, RAM memory of 8GB, and a NVIDIA GeForce GT 635M Graphic Processing Unit (GPU), which can manage up to 144 cores. To test the performance of the proposed dynamical model Portswood was chosen as the environment because it has high density of buildings. Consequently, 947 buildings and 1253 navigation intersections were enclosed by this area. Additionally, the model was evaluated with a maximum number of civilian of 2487 which depends on the number of houses and their assigned cars. The number of shelters was two which were chosen depending on their dimensions and their utilization purposes. These were the same for all the evaluations. So, the metrics for the evaluation were the total evacuation time, number of vehicles evacuated in each time step, the state of intersection during the simulation, and density of houses in the environment. As a result, the outcomes are presented depending on the type of scenario.

5.1 Environment With A Concentrated Distribution Of Buildings

To assess the efficiency of the model in urban areas with a high concentration of vehicles, the number of buildings selected was 829 and 2487 vehicles. This emulates the traffic state where most of the people are working and have to leave the buildings due to the emergency. Additionally, to analyze the performance, the outcomes have been organized according the maximum number of cars at intersections, evacuation rate for every time step, and the evacuation time.
5.1.1 Traffic Distribution

This is focussed on the distribution of vehicles in the network. Therefore the metric was the maximum number of cars at intersections during the evacuation. Although all the scenarios exhibit a low balance of traffic during the first 3 minutes of the simulation, the dynamic strategy decreases drastically this value after the fourth minute due to the rerouting and new prediction. This is made after three minutes of receiving the last update. Although, the delay due to officers travelling time remains, its effects are mitigated by the preventive nature of the new technique. On the contrary, the random strategy results are almost similar to those of the uncontrolled network. The reason may be the lack of feedback from the environment to the model. Additionally, it should be emphasized that there is going to be crowded junctions next to shelters during the evacuation because all the vehicles are directed toward them. The results of the distribution are illustrated in Figure 5.1.

![Figure 5.1: Maximum number of cars at intersections.](image)

5.1.2 Evacuation Rate

The evacuation rate has a strong dependence with the traffic distribution because congested roads makes difficult to arrive to destinations. Consequently based on the analysis of the previous section and data from Figure 5.2. It was expected that the new strategy outperforms the other control techniques due to its lowest value after applying the rerouting. This is confirmed by the increasing evacuation rate after the fourth minute. Nevertheless, the third prediction made around the eight minute is less evident due to the length of the evacuation and the fact that most of the cars are stacked at jams in the closest intersections of shelters. On the other hand, the stochastic approach performance
5. Empirical Evaluation

<table>
<thead>
<tr>
<th>Approach</th>
<th>Average ET</th>
<th>Maximum ET</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic</td>
<td>7:00</td>
<td>09:20</td>
</tr>
<tr>
<td>Random</td>
<td>8:17</td>
<td>10:26</td>
</tr>
<tr>
<td>None</td>
<td>8:10</td>
<td>11:47</td>
</tr>
</tbody>
</table>

Table 5.1: Average and Maximum evacuation time in format mm:ss for a environment with high density of traffic.

presents a similar pattern in the state of intersections as the uncontrolled network. The random rerouting avoids the high congestion rates achieved by the strategy without traffic guidance.

5.1.3 Total Evacuation Time

Based on the previous information, the outcomes, illustrated by Table 5.1, show the improvement on the evacuation management if the dynamic strategy is utilized. Its average and maximum evacuation times are the lowest of the three approaches.

Summarizing, after analyzing the metrics for the different techniques, it can be stated that the preventive strategy outperforms in scenarios with a high concentration of traffic.

5.2 Environment With A Spread Distribution of Buildings

The second environment assessed the model for scenarios where the traffic density is low. Consequently, it had 447 buildings with 1341 cars. These values are almost half compared to their setup in the first environment.
5.2.1 Traffic Distribution

Although, the number of vehicles was lower than that of the previous environment, the number of cars at intersections is higher for each time step. The reason for this behaviour of the network is that evacuees become aware faster and their evacuation speed is almost the maximum due to the number of obstacles, other cars, is lower. As a result, the concentration of traffic for the same time step is almost twice that of the previous scenario. This behaviour is illustrated by the Figure 5.3.

5.2.2 Evacuation Rate

Though in both environments this parameters decreases its growing rate after 1:30 minutes of raising the alarm, after this boundary the inclination of the curve for the second scenario is lower because cars were able to reach the nodes closest to shelters earlier which increase the congestion of these bottlenecks. Additionally, Figure 5.4 presents that while the evacuation rate of the dynamic strategy is the lowest during the first three minutes due to the rerouting, this guarantees that more vehicles are able to arrive to the destinations due to its improve distribution of traffic. However, it also shows that the improvement in the evacuation rate depends on the probability of having oversaturated junctions. As a result, the initial predictions caused a high evacuation rate, but when there is a lack of saturated intersections the growing rate of this parameter follows the normal trends.

5.2.3 Total Evacuation Time

Despite the lower number of cars at intersections, Table 5.2 illustrates that the total evacuation time was higher if the network was uncontrolled. This is a consequence of the
5. Empirical Evaluation

Figure 5.4: Percentage of car evacuated against time.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Average ET</th>
<th>Maximum ET</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic</td>
<td>6:00</td>
<td>08:40</td>
</tr>
<tr>
<td>Random</td>
<td>7:11</td>
<td>08:42</td>
</tr>
<tr>
<td>None</td>
<td>8:36</td>
<td>11:07</td>
</tr>
</tbody>
</table>

Table 5.2: Average and Maximum evacuation time in format mm:ss for an environment with low density of traffic.

increasing speed of cars in the roads. Nevertheless, if the proposed technique is applied, the evacuation time is less dependent of the velocity because the traffic distribution is better.

5.3 Execution Time of the Prediction

Section 4 states that the C# project utilizes the Matlab code as a server to integrate the predictive application. Consequently, its execution time was reflected in the main execution time of 25 seconds, approximately. This is also illustrated in the evacuation rate and traffic distribution figures in the intervals of no growing tendency. These are from 0:00 to 0:25, around 3:00 to 3:25, and 5:58 to 6:22. The reason for these outcomes is the drawbacks of the Matlab code which handles efficiently small paths. Summarizing, even though the number of cars is lower if traffic is not well distributed, the evacuation time may be higher that having more cars in the area. Additionally, the cost of having a better control strategy is reflected in the execution time of the application.
Chapter 6

Discussion

Having outlined the aims of the project in the Chapter 1 and presented the results of the implementation in the Chapter 5. This section discussed the outcomes of the projects and its future work to expand its usability.

6.1 Discussion

The main objective of the project has been accomplished because a dynamic control strategy that minimizes the evacuation time in a disaster scenario and the simulator has been implemented taking into account the criteria explained in the Chapter 1. Consequently, the findings, and limitations are presented in this section.

The dynamic control system utilizes police officers as a guiding mechanism because they are an available resource that is not affected by the lack of power electricity. Additionally, policeman provides calm to drivers. This makes them more likely to follow guiding orders. Moreover one of the more significant findings to emerge from this study is that control strategies do not require real time information to have a good performance. For instance, the prediction module requires data such as the number of cars at intersection and the direction of the traffic flow that can be easily inferred. This provides flexibility to the model. Besides, the importance of the travelling time was illustrated in the decreasing evacuation time achieved by the new technique.

The results of this investigation suggest that the new control technique is efficient for scenarios with a high probability of having jams at intersections. That is why it can be widely applied in the traffic evacuation. On the other hand, the multi-agent system handles real world scenarios and provides metrics that facilitate the evaluation of the control strategy. These metrics include the total evacuation time, number of cars at shelter during the evacuation, and the state of junctions. This allows checking that the traffic distribution has been improved. The most important limitation lies in the fact that the predictive application is not able to handle large paths. This might be due
to the model requires to infer additional parameters such as the injection rate of cars to road to provide a good estimate. The effect of this drawback is the high execution time due to routes has to be partitioned in smaller sub paths for getting reasonable predictions. This limits the application of the strategy in larger scenarios.

6.2 Future Work

A number of possible future research areas have been identified through the development of this project. In terms of usability, further research needs to be done for handling large scale scenarios. One potential improvement is to code this application inside the C# project to exploit the parallel computing of the GPU. This might eliminate this algorithmic bottleneck. For emulating more accurately real world evacuation scenarios, the location of the incidents should be included in the simulator. This could allow redirect jams close to shelter toward another safe zone without guiding cars toward the incident. Additionally, the behaviour of cars could be more dynamic if vehicles are able to choose from a set of paths because in real scenarios drivers may not follow the shortest route. Moreover, this project can be implemented utilizing software packages that do not depend of the operative system because a considerable amount of time was spend trying to update the application to the newest releases of Windows. Finally, this new strategy performance should be evaluated against other control approaches such as dynamic programming models in terms of usability, computational time, data required, and traffic distribution.

6.3 Conclusions

The system implemented overcomes some of the limitations of the traffic evacuation management.
First, the evaluation scenario of the strategy defines specific points as evacuation zones. This guarantees that people can receive opportune medical care and further information about the incident.
Additionally, the real time information required for the model is accessible and easily inferable which separates its performance from the availability of data from sensors.
Moreover, using officers as a guiding mechanism not only ensures that drivers follows the orders but also provides calm and confidence to the evacuees. This also ensures that roads will be controlled even thought there is not electricity.
Consequently, the proposed new control technique employs available resources, utilizes few real time information, and its outcomes are better that the random approach.
Appendix

This section illustrates the environments utilized to evaluate the performance of the dynamic strategy. Consequently Figure 7.1 and 7.2 represent the scenarios with high density of buildings and spread distribution of houses, respectively.

Figure 7.1: Environment with a high density of buildings.
Figure 7.2: Scenario with a spread distribution of houses.
References


REFERENCES


