PHD

Computer-generated Circulation Diagrams

Kontovourkis, Odysseas

Award date:
2009

Awarding institution:
University of Bath

COMPUTER-GENERATED CIRCULATION DIAGRAMS

Odysseas Kontovourkis

A thesis submitted for the degree of Doctor of Philosophy

University of Bath

Department of Architecture and Civil Engineering

March 2009

Odysseas Kontovourkis

Acknowledgements

I would like to thank my supervisor, Dr. Chris J. K. Williams for his great help and support during my research studies and especially for his valuable advice in the theoretical and practical parts of this work.

I would like to thank the British Council for awarding me a British Chevening Scholarship by the Foreign and Commonwealth Office (UK) in the first year (2004-2005) of my studies.

I am particularly thankful to Dr. Gonzalo A. Garcia for his useful feedback on various stages of this work and especially for his corrections of chapter 5.

Last but not least, I would like to thank my family for their continuous support, patience and kindness. This work would not have been done without their encouragement.

Abstract

The way in which computers are used is important in the theory, philosophy and practice of architecture. Architects are already using computers to construct complex three dimensional geometric models of their buildings and are beginning to analyse these models using environmental and structural software, a development which raises new questions about the role of architects and engineers.

This dissertation puts forward the hypothesis that architects will at times need to be actively involved in computer programming by writing or modifying software. The hypothesis is based on the assumption that the form of a building and its spatial configuration are influenced by the nature of the design process itself. If architects are to have a complete and subtle control over design and to identify their own personal aesthetic language, they must also have control over the design process including the way computer software tools are developed and used.

The hypothesis is tested using the example of a ferry terminal, a building type whose function is largely dominated by passenger circulation. Even thought passengers have a very straightforward aim to reach their final destination, the rules governing the way passengers move around the building are complex and a single computer program will not be able to cover all possible aspects of such behaviour. Thus, architects must have the freedom to formulate different rules and study the effects they have on their design. This particularly applies in non-emergency situations when each individual moves inside the building according to different needs and desires.

A program was written which runs in real time so that the architect can see the effect of changing the parameters that control the process. The program can be used as evaluation mechanism to study the performance of postulated design or it can be use as creative mechanism where the design may emerge out of the process in the same way that animals create paths in the woods. Either way, the general aim is to optimize the design according to criteria over which the architect, again, must have complete control.

Contents

List of figures

Chapter 1: Introduction

Chapter 2: The design process

Chapter 3: The design process and computers

Chapter 4: Circulation design

Chapter 5: Mathematical model and computer code

Chapter 6: Case study, a ferry terminal

Chapter 7: Conclusions

List of tables

Chapter 2: The design process

Chapter 4: Circulation design

Chapter 6: Case study, a ferry terminal

List of symbols

Symbol	Description	Unit
C_{parallel}	Sign strength parallel to sign	Force/mass, N/kg
$C_{\text{perpendicular}}$	Sign strength perpendicular to sign	Force/mass, N/Kg
C_{attract}	Attraction over mass factor	Force/mass, N/Kg
C_{repel}	Repulsion over mass factor	Force/mass, N/Kg
$C_{\text{obstacle repel}}$	Obstacle repel factor	Force/mass, N/Kg
α_{sign}	Sing-person distance factor	Length, m
$\alpha_{\text{interperson attract}}$	Attraction distance factor	Length, m
$\alpha_{\text{interperson repel}}$	Repulsion distance factor	Length, m
$\alpha_{\text{obstacle repel}}$	Obstacle-person distance factor	Length, m
x, y, z	Coordinate axis	Length, m
L	Length	Length, m
t	Time	Time, second
$\mathbf{f}_{\text{attract}}$	Inter-person attraction force	Force/mass, N/Kg
$\mathbf{f}_{\text{repel}}$	Inter-person repulsion force	Force/mass, N/Kg
$\mathbf{f}_{\text{parallel}}$	Sign force parallel	Force/mass, N/Kg
$\mathbf{f}_{\text{perpendicular}}$	Sign force perpendicular	Force/mass, N/Kg
$\mathbf{p}$	Unit vector	Dimensionless
$\mathbf{q}$	Unit vector	Dimensionless
m	Mass of person	Mass
Note: $\text{Force} = \frac{\text{mass} \times \text{lenght}}{\text{time}^2}$, $1\text{N} = 1\text{kg} \times \text{m/sec}^2$		

Chapter 1: Introduction

1.1 Computers and people

Computers have become an integral part of everyday life influencing almost all human activities. Digital devices like bank cashpoints, computer games, e-mail, the Internet, word processors, aircraft autopilots, engine management systems, all influence people's thoughts and behaviour. As with all advances in technology, certain skills become worthless and new skills come to the fore. For example, the ability to type a page without making a mistake, once so valuable, is rendered worthless by word processors. It is no longer necessary for an architect to be able to draw with a pen or for an engineer to do calculations with only a slide rule, although it could be argued that these skills are still valuable, particularly in the initial stages of the design process.

Moore's law [1965] states that computer power doubles every two years, so that now computer speed and cost are rarely a limitation on their use. On the other hand 'software bloat' as described by Wirth's law [1995] means that much of this power is wasted. As more and more features are added to programs, familiarity with a particular piece of commercial software becomes a skill in its own right and users understandably modify their use of programs to demonstrate their skills, from bullet points in a MicrosoftWord document to freeform design in architecture.

This raises the question of how a harmonious balance between the human and digital worlds can be achieved. When the computer performs a relatively mechanical task such as typing, flying an aeroplane or architectural drawing, it has many consequences for the people involved, but at least it is clear what the computer is doing. But when the computer is involved in a decision-making process that requires judgement, its role is more insidious.

In engineering it used to be necessary for the engineer to have a simple mental model of how the object they were designing should work. Hand calculations were based on this model. The engineer decided how it would work and ensured that it would do so. Now the simple model is unnecessary, at least from the point of view of analysis, since the analysis can be done by computers 'number crunching'. Engineers are taught to be distrustful of computer calculations because they are sometimes wrong, either because of the input data or the software itself. Computers have been used for more than forty years as part of the decision-making process in engineering and engineers are to a greater or lesser extent culturally and intellectually equipped to use computers.

Until relatively recently the majority of architects only used computers as electronic drawing boards, making it easier to produce, modify, store and distribute drawings. Drawings were still done line by line and modified line by line. Now parametric design is increasingly used. Parametric design [Fagerstrom, 2007] is the construction of a logical hierarchical structure by which each part of a drawing depends upon a small number of independent parameters. Changing any one of these parameters causes the whole drawing to change, updating automatically.

However parametric design is essentially deterministic in that a change to any one of the parameters has a unique and predictable effect. This makes it valuable for the production of working drawings, but less so at the conceptual level during which ideas such as analysis, optimisation, emergence, and even chaos or catastrophe (all mathematical concepts) might be more appropriate. In these fields the results of the computation are not easy to predict and the most interesting outcomes for architects are those that go against what was expected.

An algorithm is a set of rules for accomplishing a certain task [Terzidis, 2006]. Both the rules and the task have to be well defined, that is defined unambiguously. Algorithms are often visualized by flow charts such as the ‘getting up’ flowchart in figure 1.1.

Figure 1. 1: ‘Getting up’ flowchart

This is a very simple example and there might be a whole range of other factors that might influence whether or not you get up, such as a friend coming to stay for the weekend. Also, getting up is far from trivial exercise involving all sorts of decisions about what to wear and what to have for breakfast.

Because architectural design is such a complex task, architects, builders and governments have attempted to set out systems of rules from Vitruvius' *The Ten Books of Architecture* to the *England and Wales Building Regulations*. However, these rules require human intelligence for interpretation, even when applied to simple physical things like damp-proof courses. A computer has no intelligence and no understanding of the physical world of human needs, mores and taboos.

The success of computation in engineering is due to the fact that metals, concrete, fluids and even soils and timber behave in relatively simple ways that can be described by algorithms. The same applies to heat flow, acoustics and all other purely physical processes. However, Hubert and Stuart Dreyfus [1986] argued in a paper entitled *Why Computers May Never Think like People* that there are fundamental differences between a computer and a human brain. In architecture, since the first development of computational tools in the early 1960's the ability of computers to address architectural design problems has been very limited. Architects such as Alexander [Alexander, 1965, cited by Cross, 1977, p. 58] criticized such applications:

> 'A digital computer is, essentially, the same as a huge army of clerks, equipped with rule books, pencil and paper, all stupid and entirely without initiative, but able to follow exactly millions of precisely defined operations […] In asking how the computer might be applied to architectural design, we must, therefore, ask ourselves what problems we know of in design that could be solved by such an army of clerks […] At the moment, there are very few such problems'

However, it might be possible to write an algorithm that describes certain limited types of human behaviour such as in the evacuation of a building in an emergency [Okazaki and Matsushita, 1993].

Thus any algorithm to be used in analysing human needs and behaviour must be a minute subset of all the possible rules. Part of the architect's role is to decide upon a unique bespoke set of rules to be adopted for a particular design, whether for a house, theatre, hospital or church. If these rules are to be incorporated in an algorithm for some sort of computer analysis, then the architect must be free to choose the algorithm. This might be possible by some sort of 'drop down menu', but since the number of possible rules and their combination is effectively infinite there are severe limitations to this approach. If it is not feasible to select the algorithm from a list, there is no alternative but to create one's own. Coates and Schmid [1999, p.653] wrote:

> ‘Baroque rulesets are self-defeating, they already specify the majority of the problem solution, and anyway they take too long to program, much better to write small programs that you can chunk away and try again with another terse summing up of the way to behave’

Design is a team activity and no one person has the time, knowledge and experience to undertake all tasks. If some computer programming is to be done to describe an algorithm used for design, then the architect must either write it himself or herself or have a close working relationship with whoever does write it. This hypothesis will be tested in this dissertation by the computer analysis of human circulation.

1.2 Circulation diagrams

People are either stationary relative to the Earth (or some machine in which they are travelling, such as a ship) or they are moving. The word circulation is used to describe such motion, but it does have the suggestion of a purely mechanical motion whose sole intent is to get from one point in space to another. The implication is that time spent circulating is wasted and should be minimized so that time can be spent on more fruitful and rewarding activities such as watching television or answering e-mails.

However, if circulation were a purely practical, technical matter of people moving from one place to another in the minimum time, it would be best solved by mathematicians using techniques such as graph theory as exemplified by the problem of the Seven Bridges of Königsberg[1]. A decision problem can only solved mathematically if there is an optimum solution; that is if there is an agreed procedure for comparing any two proposed solutions and deciding which is better.

But circulation is much more than that, the aesthetic impact of a building or a city or a natural landscape is intimately connected with the time-ordered sequence of how it is experienced, and no two people will experience it in the same way. And no single person will experience it the same way twice. A building is defined by the way people interact with it, by what they see, hear and how they move about. Thus the rules of circulation will vary from building type to building type, from culture to culture, and with the views of the individual client and architect. The circulation diagrams might define the building as shown in figure 1.2.

---

[1] The city of Königsberg (now Kaliningrad, Russia) is on a river with two islands linked to each other and the banks by seven bridges (now there are more). Euler showed that it is not possible to cross all the bridges without crossing at least one of them more than once.

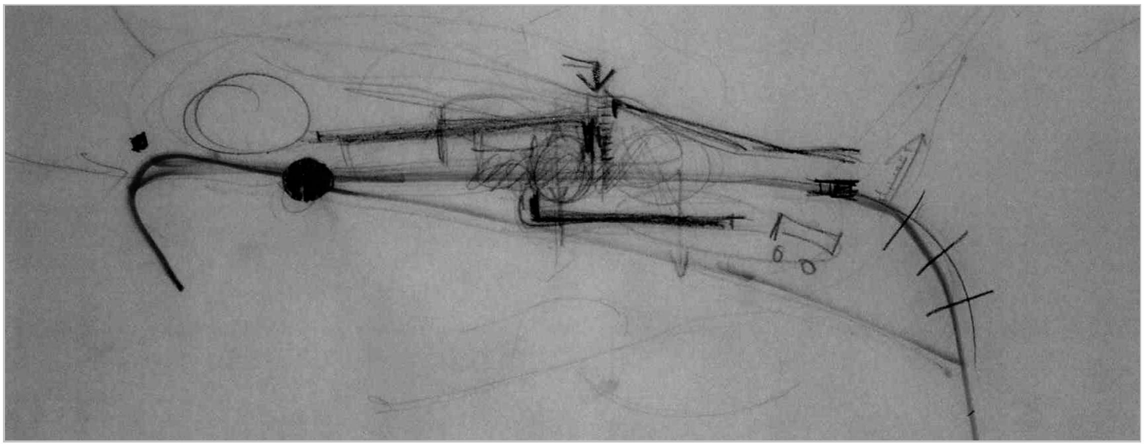


Figure 1. 2: Circulation diagram used for the organization of a building [Kaklamani, Kontovourkis, and Pavlou, 1999]

Architects are trained to make decisions where the problem is ill-defined, where there is no clear idea of what is optimum. But nevertheless they are also taught to make use of scientific and mathematical methods, even if the design decision-making process is not strictly logical.

This dissertation describes the process of writing and testing relatively simple computer programs in which the architect is free to modify the algorithm or even add in completely new rules. Issues such as emergence and chaos are examined, in which the effect of changing rules have unpredictable effects.

A secondary aim of this dissertation is to introduce the techniques of computer programming to architects so that they can have a better understanding of what is and what is not possible with computers.

1.3 Structure of the dissertation

This dissertation is interdisciplinary in that it considers questions of architectural philosophy and theory, mathematics, computer programming and human psychology in the way in which people move around a building. This means that the review of the literature must be selective and cannot examine all possible relevant works. Rather than have one chapter entitles ‘literature review’, the literature is discussed when appropriate in the chapters that follow:

Chapter 2: The design process is intended to be a brief introduction to the theory and philosophy of design with a view to establishing a framework for the dissertation as a whole.

Chapter 3: The design process and computers discusses how computational approaches have developed to aid the process of design.

Chapter 4: Circulation design examines the issue of circulation within and between buildings in general terms.

Chapter 5: Mathematical model and computer code describes the way in which the circulation modelling computer program was developed.

Chapter 6: Case study, a ferry terminal examines the results from the computer program.

Chapter 7: Conclusions draws general conclusions.

Chapter 2: The design process

2.1 Design space

In order to discuss the use of computers in the design process we need a conceptual model or framework. We can then examine how a computer might be used at different locations within the framework.

Let us use the phrase ‘design space’ to mean all possible solutions to a design problem. This is not a new concept, optimisation techniques use a ‘solution space’, the set of possible solutions to a problem. Design space is multidimensional in that many parameters (dimensions, materials etc.) can be changed. A point in this space corresponds to a particular design. A ‘scheme design’ is a region in design space with fuzzy edges since at the scheme design stage the exact location in design space is not clearly defined. Thus the design process involves choosing a region in design space and shrinking the region until it corresponds to the point of final design.

Clearly some locations in design space are ‘better’ than others according to innumerable criteria: function, aesthetics, cost, environmental impact, structural safety, comfort, etc. etc. If it were possible to apply numerical values and weightings to how well each of these criteria were satisfied, then they could be added and an overall ‘suitability rating’ found for any candidate point in design space.

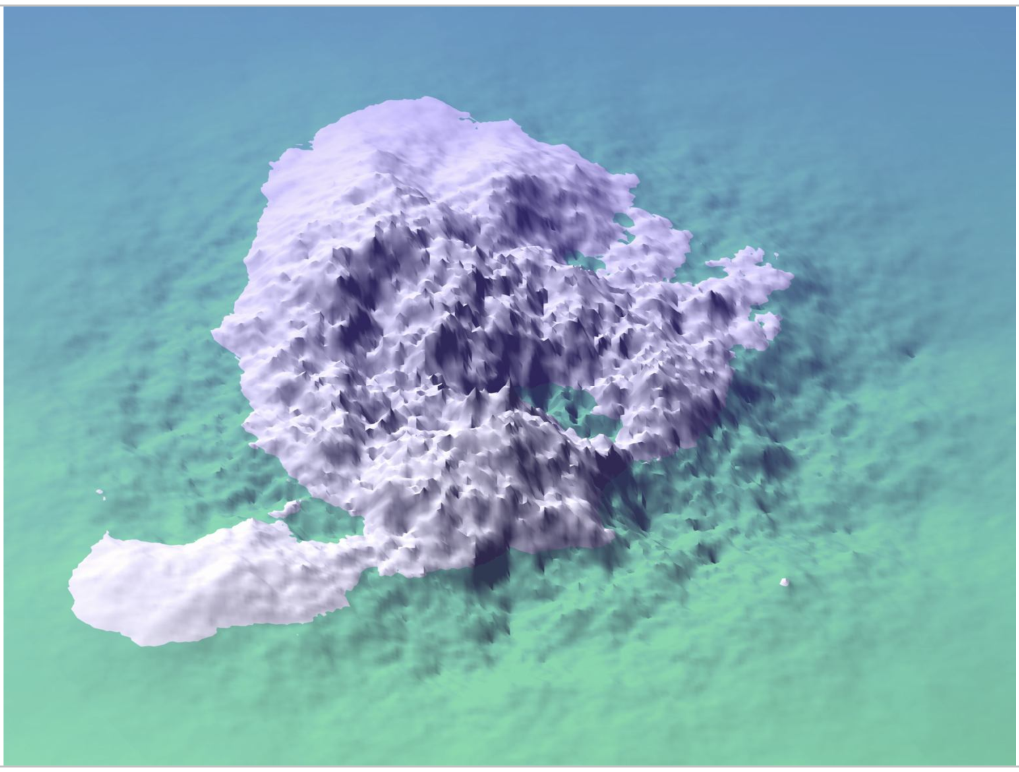


Figure 2. 1: Design space

In order to grasp this concept visually, let us imagine that the design space were only two dimensional. Then the suitability rating can be used as the third dimension to produce a ‘design landscape’. Then the ‘best’ design is the highest point in this landscape.

The idea of moving about a landscape is common in the field of optimization, but in architecture the form of the landscape is not smooth but fractal-like with craggy peaks and chasms. The design landscape shown in figure 2.1 has some points ‘under water’ meaning that they have failed some crucial test so that the design is not fit for purpose and may even endanger life. Unfortunately it is easier to show that a design is not fit for purpose than to prove that it is. In fact it is not possible to prove beyond doubt that a design is fit for purpose.

Equally unfortunate is the fact that we cannot ‘see’ the whole of design space as it is shown in the ‘aerial photograph’ in figure 2.1. We have no map and blindly stumble around more or less at random, sometimes jumping long distances from one possible scheme to another.

The estimation of numerical values and weighting functions is problematic (and highly subjective) and is never done, but the concept is useful in trying to understand the design process. It means that any design process with purports to produce an ‘optimum’ should be treated with skepticism.

If we accept this model, architectural education is about teaching students to navigate this design space. Students are taught how to:

i. Decide where to go in design space
ii. How to specify where they actually are - either a point or usually some larger region – by producing drawings and models
iii. How to decide how high up they are - how suitable is the design?
iv. How to decide where to move to next and/or to shrink their region

2.2 Conceptual framework

Our conceptual framework is based upon navigation around design space. Figure 2.2 shows a conceptual framework proposed by the architect Michael Brawne [1981] based on the work of the philosopher Karl Popper [Popper, 1972]. The diagram was drawn by the author as a visual interpretation of Brawne's [1981] text. So our aim now is to construct a framework using these ideas and that of design space.

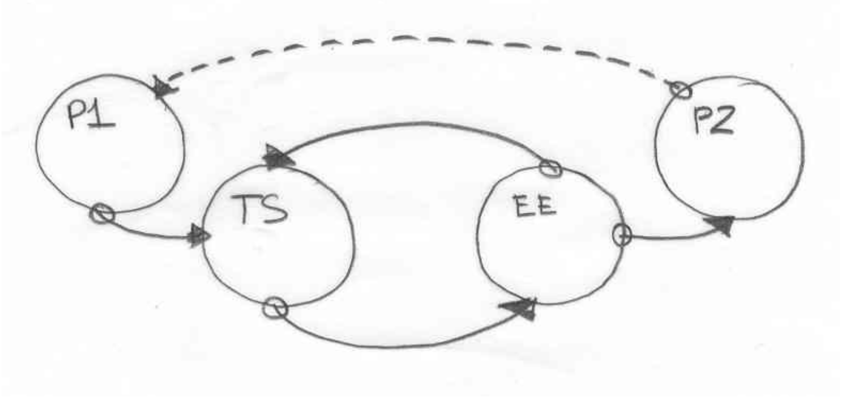


Figure 2. 2: Hypothetical representation of Popper's idea of design process and cyclic process of 'hypotheses' and 'critical evaluation'

Karl Popper described the scientific process in general as a sequence: P1->TS->EE->P2 where P1 is the initial perceive problem, TS is the tentative solution to it, EE is the process of error elimination, and P2 is in one sense a solution but in another sense a new problem.

The design process can be seen to be cyclic moving between alternate phases of 'hypotheses' and 'critical evaluation'. According to Steadman [1979], the modern design process and Popper's generalized trial-and-error scheme for the creative processes in science have many connections. Brawne [1981] considered that the above sequence is close to the design process because the final product becomes a part of a known solution, it can be starting point for a new sequence and it can be seen as a model to be modified. In figure 2.2 the tentative solution (TS) can be seen as the 'hypothesis' and error elimination (EE) as the 'critical evaluation'.

During the late 1950's and the 1960's researchers tried to describe the creative problem solving process as logical structure of activities [Rowe, 1987]. Different representations were investigated based on the general idea that design can be regarded as a series of stages such as analysis, synthesis, evaluation, etc.

Asimow [Rowe, 1987] described two structures in the design process: a vertical structure which consisted of all activities and a horizontal structure in the form of decision-making cycle as can be seen in figure 2.3a. Archer [Rowe, 1987], proposed a similar 'operational' model of design where the design process was seen as a sequence of activities defined by their orientation and by the general type of task involved. Programming, data collection, analysis, synthesis, etc. were included in the forward path of the design process and evaluation was on the backward path creating design loop as shown in figure 2.3b.

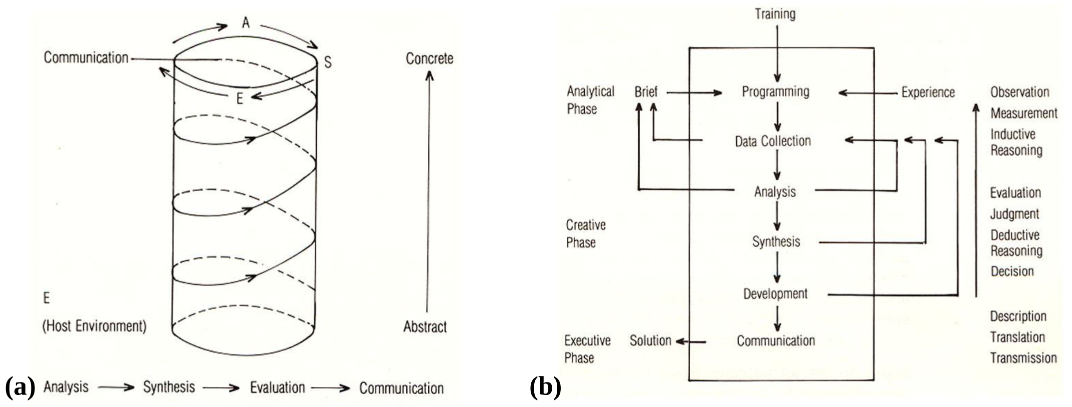
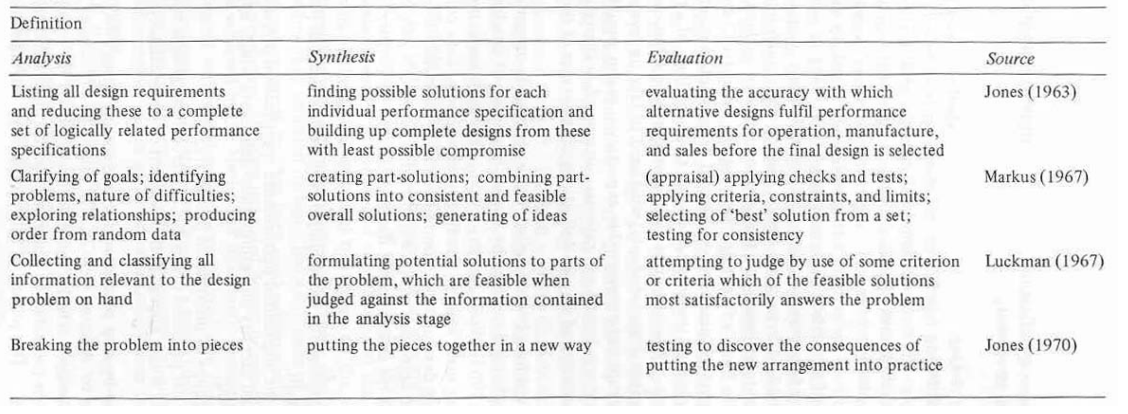


Figure 2. 3: a. An iconic model of a design process [Rowe, 1987], b. Archer's model of the stages of a design process [Rowe, 1987]

A similar approach of analysis, synthesis, and evaluation was followed in Jones' Method of Systematic Design [1963, cited by Cross, 1977].



Definition			
Analysis	*Synthesis*	*Evaluation*	*Source*
Listing all design requirements and reducing these to a complete set of logically related performance specifications	finding possible solutions for each individual performance specification and building up complete designs from these with least possible compromise	evaluating the accuracy with which alternative designs fulfil performance requirements for operation, manufacture, and sales before the final design is selected	Jones (1963)
Clarifying of goals; identifying problems, nature of difficulties; exploring relationships; producing order from random data	creating part-solutions; combining part-solutions into consistent and feasible overall solutions; generating of ideas	(appraisal) applying checks and tests; applying criteria, constraints, and limits; selecting of 'best' solution from a set; testing for consistency	Markus (1967)
Collecting and classifying all information relevant to the design problem on hand	formulating potential solutions to parts of the problem, which are feasible when judged against the information contained in the analysis stage	attempting to judge by use of some criterion or criteria which of the feasible solutions most satisfactorily answers the problem	Luckman (1967)
Breaking the problem into pieces	putting the pieces together in a new way	testing to discover the consequences of putting the new arrangement into practice	Jones (1970)

Table 2. 1: Systematic design is a three-stage process, demanding analysis, synthesis and evaluation [Page, 1963 cited by Cross, 1977]. Some of the definitions which have been given for these three stages [Cross, 1977]

Cross [1977] stated that:

> 'Systematic design was not meant to be a replacement of intuition by logic, but a synthesis of the two: this is much the same argument that has been used subsequently in connection with the development of computer-aided design'

The analysis-synthesis-evaluation model of the design process has been widely used in engineering providing a feedback loop mechanism where designers are able to move back in an iterative manner in order to refine and improve the design until requirements are satisfied [Braha and Maimon, 1997]. Analysis is the stage where designers define and understand the goal or functional requirements. In the synthesis stage solutions are found and evaluation is the process of assessing the validity of the solution based on functional requirements stated in the analysis stage. A chart describing this process is shown in figure 2.4. Note that there is no unique meaning for these words, using this terminology ‘structural analysis’, as understood by engineers, would be described as ‘evaluation’.

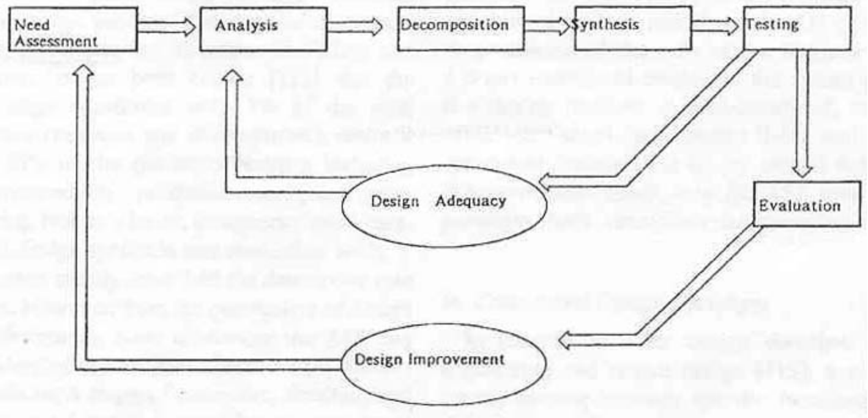
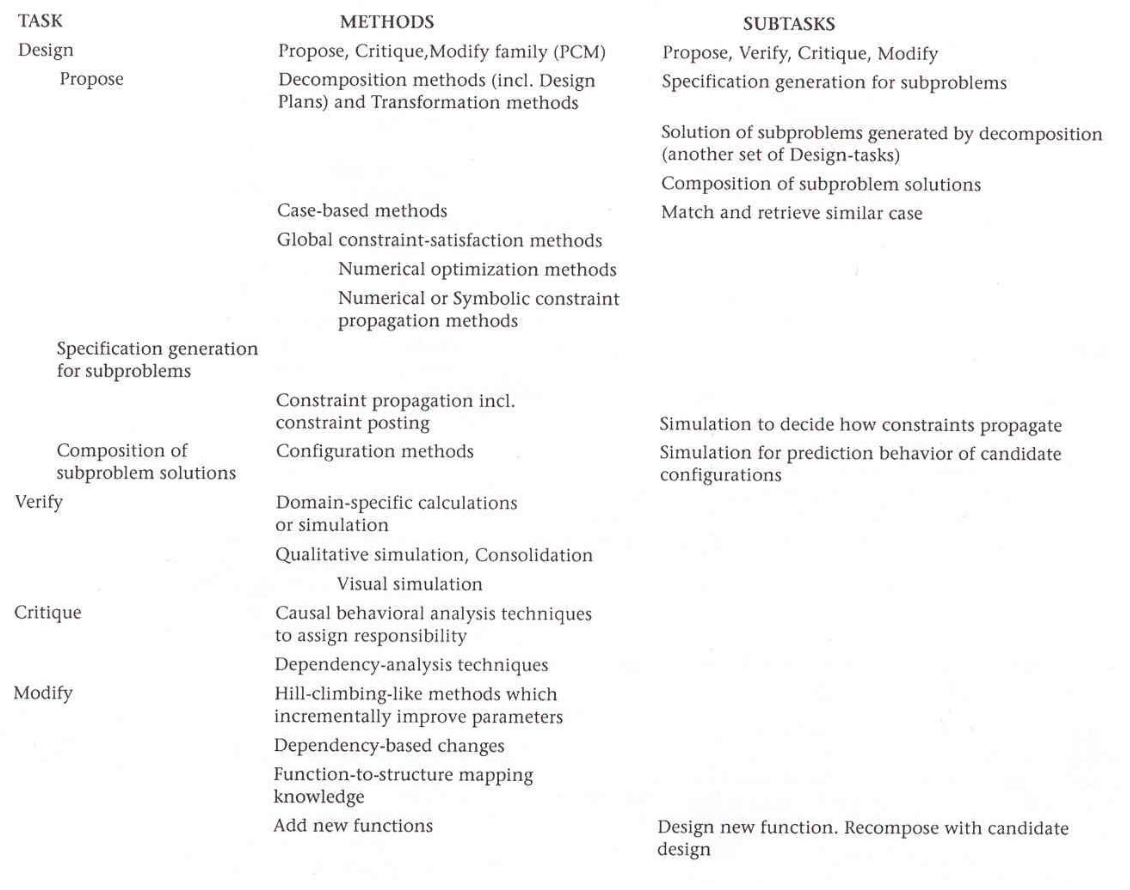


Figure 2. 4: Comprehensive model of the design process [Braha and Maimon, 1997]

The Task Structure for Design (see table 2.2) is a similar cyclic method proposed by Chandrasekaran [1990] in the paper *Design Problem Solving: A Task Analysis.*



TASK	METHODS	SUBTASKS
Design	Propose, Critique,Modify family (PCM)	Propose, Verify, Critique, Modify
Propose	Decomposition methods (incl. Design Plans) and Transformation methods	Specification generation for subproblems
		Solution of subproblems generated by decomposition (another set of Design-tasks)
		Composition of subproblem solutions
	Case-based methods	Match and retrieve similar case
	Global constraint-satisfaction methods	
	Numerical optimization methods	
	Numerical or Symbolic constraint propagation methods	
Specification generation for subproblems		
	Constraint propagation incl. constraint posting	Simulation to decide how constraints propagate
Composition of subproblem solutions	Configuration methods	Simulation for prediction behavior of candidate configurations
Verify	Domain-specific calculations or simulation	
	Qualitative simulation, Consolidation	
	Visual simulation	
Critique	Causal behavioral analysis techniques to assign responsibility	
	Dependency-analysis techniques	
Modify	Hill-climbing-like methods which incrementally improve parameters	
	Dependency-based changes	
	Function-to-structure mapping knowledge	
	Add new functions	Design new function. Recompose with candidate design

Table 2. 2: The Task Structure for Design [Chandrasekaran, 1990]

The method follows the sequence propose-verify-critique-modify. In this case the proposal is evaluated in ‘critique’ stage and then is modified. The process iterates in ‘verify’ stage till the design proposal satisfies the design goals [for further details, see Chandrasekaran, 1990].

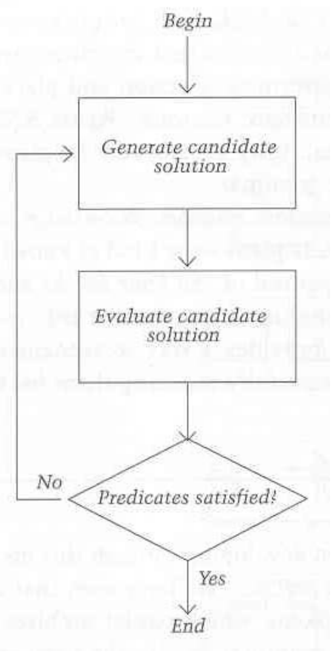


Figure 2. 5: The basic trial-and-error structure of a design process [Mitchell, 1996]

The application of computers in the general design process has been examined by William J. Mitchell [1996]. Again general design is seen as trial-and-error process where a computational device can be used as generation, test, and control mechanism as shown in figure 2.5. Different possibilities can be investigated in a collaboration between digital device and human designer.

2.3 Proposed design framework

In the design landscape introduced at the beginning of this chapter the ‘best’ design is the highest point in the landscape. In order to reach an optimum point and achieve a ‘good’ solution, the process needs to follow a cyclic procedure which moves from the ‘synthesis’ to ‘evaluation’. The ‘synthesis’ part deals with creativity and the ‘evaluation’ part is more objective and is concerned with accuracy.

Figure 2.6 is an attempt at a comprehensive description of the design process. This figure is entitled ‘Design Framework’ and the phrase will be used throughout the dissertation to describe this diagram and developments of the diagram. It includes two loops, the inner one that is refining the design or shrinking the region in design space, and the outer loop that moves the design in new position in design space. ‘Good’ design can be achieved by repeating both loops for a number of cycles.

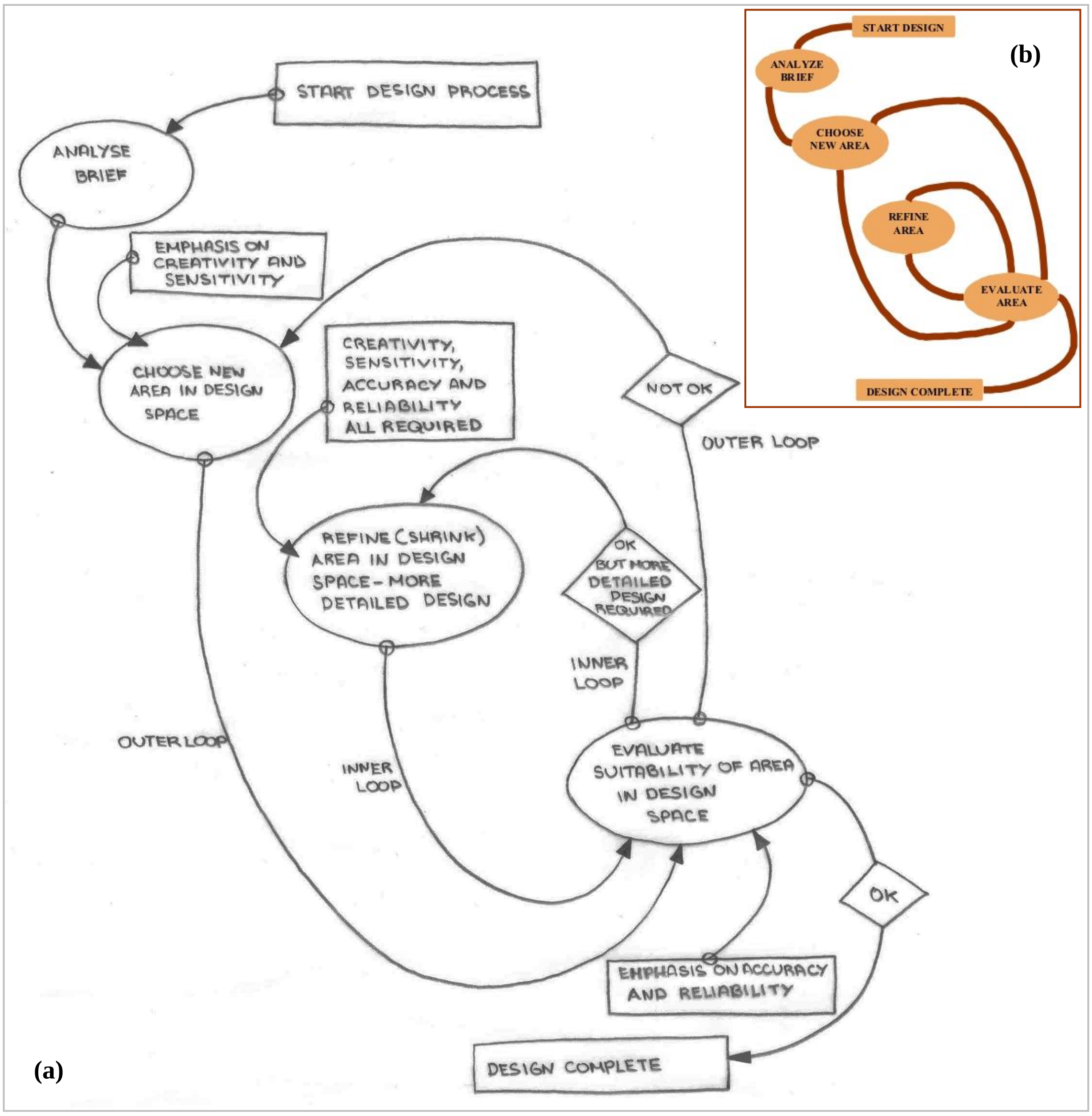


Figure 2. 6: a. Design Framework, b. Abstract representation of Design Framework

Computers can be used for:

I. Locating a point or region in design space (3D models, 2D drawings)

II. Assessing the suitability of a design (engineering and cost calculations)

III. Suggesting new location in design space to be investigated

It is the third of these applications which is perhaps the most interesting and where the ‘rules’ are least defined.

Chapter 3: The design process and computers

In this chapter we will discuss the relation between the design process and the use of computers. In order to achieve this we will try to locate different computational approaches within the Design Framework and see their possible contribution to the overall process. Also, we will try to investigate how different approaches help navigate the design space.

If we accept that searching for a ‘good’ solution in the design space is based on rules of navigation then we need to investigate in parallel different rules that can be applied and see their effect in each design approach.

3.1 Philosophical inspiration

Since the first applications of computers in design, their role has been associated with the practical and rational aspect of design and the design process. However, gradually their role has expanded and computers began to be used as an artistic and creative design tool. Architects like van Berkel and Bos [Feireiss, 1993][Berkel and Bos, 1999], Greg Lynn [1999][2004] use various computational techniques and design processes as part of their inspiration and make connections with philosophical ideas, particularly those introduced by Deleuze [Jormakka, 2002].

Gilles Deleuze (1925-1995) was a French philosopher who has had a major influence on architectural theorists and philosophers. Andrew Balantyne in *Deleuze & Guattari for Archiects* wrote [Ballantyne, 2007, p.8]:

> ‘Through the fog of unfamiliar terminology here, there is a sense of the pervasive provisionality, seeing the world as an open flux of possibilities, that makes Deleuze and Guattari’s writings so appealing to architects, who find themselves called upon to find form of buildings. A precondition for finding form is to be without form, to suspend the condition of having form, so that a new possibility can emerge’

Deleuze himself makes little reference to the use of computers, although in *The Fold, Leibniz and the Baroque* [Deleuze, 1993, p. 14] he wrote:

> ‘Bernard Cache defines inflection – or the point of inflection – as an intrinsic singularity. Contrary to “extrema” (extrinsic singularities, maximum and minimum), it does not refer to coordinates: it is neither high nor low, neither right nor left, neither regression nor progression. It corresponds to what Leibniz calls an “ambiguous sign”. It is weightless; even the vectors of concavity still have nothing to do with a vector of gravity since the axes of the curve that they are determining oscillate around it. Thus

> inflection is the pure Event of the line or of the point, the Virtual, ideality par excellence'

Clearly here Deleuze does not use the word 'Virtual' to mean 'Digital'. Bernard Cache (born 1958) is a French architectural thinker who uses computers to produce architectural objects [1995]. A point of inflection is the point at which the curvature of a line drawn on a plane changes from being positive to negative, that is from being convex to concave. It is independent of the orientation of the curve, whereas the maximum or minimum (highest and lowest points) move as the curve is rotated.

In his translator's introduction to *The Fold, Leibniz and the Baroque*, Tom Conley wrote:

> 'An exquisitely sensuous view of the world is obtained through the curved shapes that Leibniz creates with calculus, and from manifestations of folds that we follow in modern art and poetry'

This poetic relationship between geometry, especially curved geometry, architecture and art, as it is expressed through Deleuze's writing can be possible through the use of computers, if there is an appropriate software. This digital description may simply be a representation (architectural and engineering drawing) of a form that has developed without the use of a computer or the digital description may be the result of an emergence produced by the repeated computation of a system of algorithms.

It is interesting to speculate on what Deleuze would have said about the relationship between computers and architecture (and computer programmes and architecture) and the design process and the use of computers.

3.2 Analysis of the brief

The analysis of the brief was examined by researchers in the early application of computers in design. It is basically the organization of functions and requirements necessary before the architect can start to navigate design space.

In our model the analysis of the brief is located after the beginning of design process and before first location of a tentative area of design space, as shown in figure 3.1. It is the stage where the architectural problem is broken down into clear and separated design sub-problems. As noted above, activities such as structural analysis are not included in our definition of 'analysis' but they are part of the evaluation stage of the Design Framework.

An early example of the analysis of the brief was Christopher Alexander's [1964] work for the determination of functional requirements of an Indian village. The aim was to create an analytical model that would combine functional requirements and group them based on rules of connectivity.

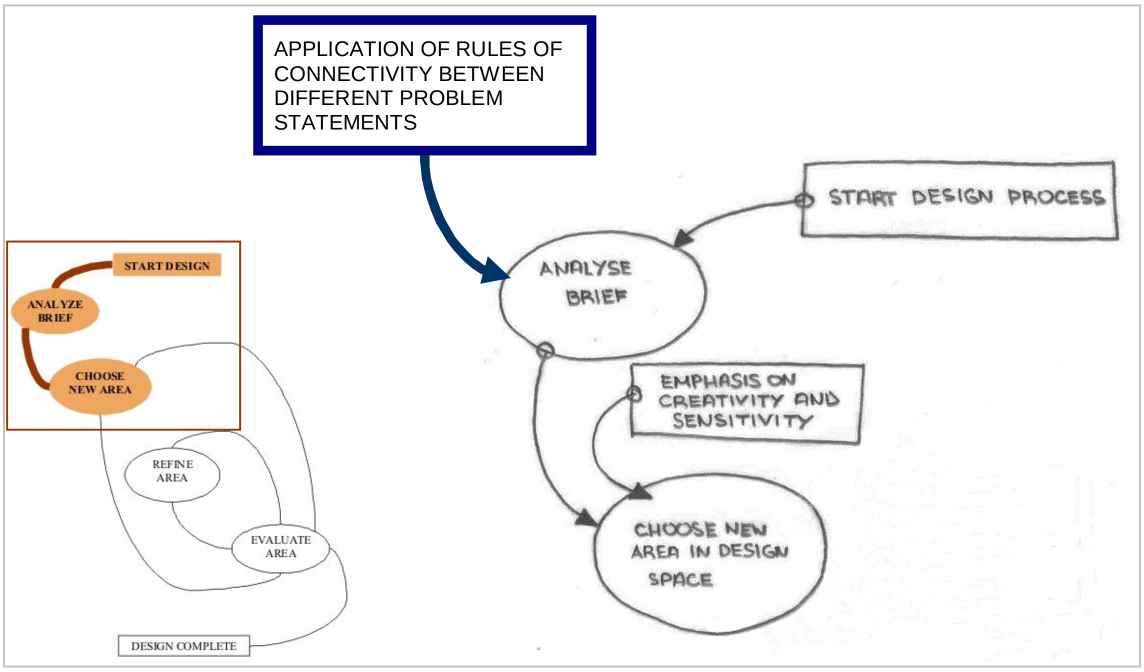


Figure 3. 1: Analysis part of Design Framework

Alexander described the problem by 141 statements of requirements. The relations between these statements were expresses in binary form of 'yes' if there was a relation and 'no' if there was no relation. Then, very closely interrelated groups of statements were identified and basic sup-problems were created. At the end, the isolated sub-problems were combined together to create a single problem statement that included all the other statements.

Rules and algorithm were used to identify the interactivity between the different requirements and then to partition those requirements into groups. As a sequence, the system consisted of groups that were sub-problems of the total problem offering a new interpretation of the organizational structure of the village [Alexander, 1964] [Cross, 1977].

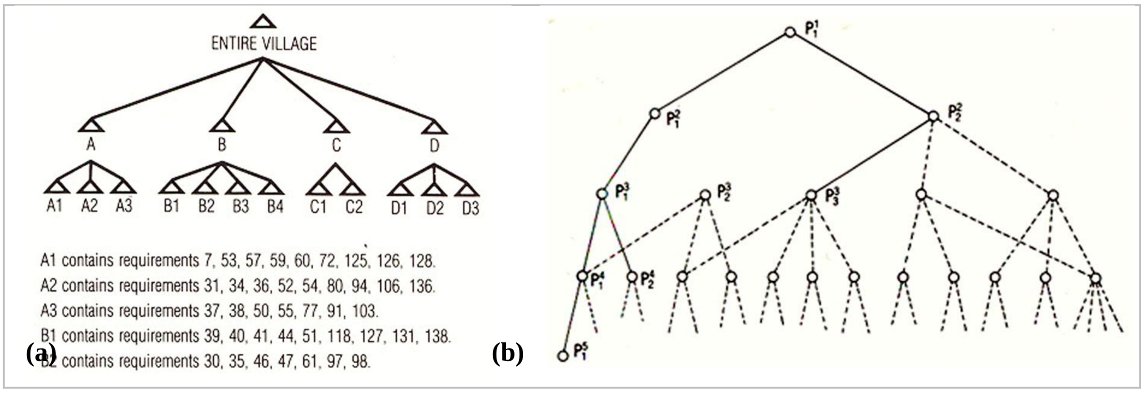


Figure 3. 2: a. Tree structure from Alexander's procedure [Alexander, 1964], b. A hierarchical diagram of a problem space [Rowe, 1987]

Alexander's approach was based on the bottom-up hierarchical decomposition-re-composition method in which a problem is broken down into fundamental components and then the components are recombined according to the problem statement [Rowe, 1987]. The tree structure from the Alexander's procedure is shown in figure 3.2a [for further details, see Alexander, 1964] and a similar example of hierarchical diagram of the problem space is given in figure 3.2b [for further details, see Rowe, 1987].

3.3 An early application of computers in the creative process

In the early development of computer-aided design, computers were used to solve space allocation and circulation problems. Computers were not able to create a complete plan layout but just investigate partial problems such as room arrangement. Whitehead and Eldars' [1964] program was one of the first attempts to generate plan layouts based on building users' movement with the aim of producing an economical design.

Such approaches used a matrix including information on interactions between all possible pairs of rooms. In this case the interactions were journeys made between each pair of the rooms in a certain time. Existing buildings and the various types of users were used to derive the information.

The basic function of the algorithm was to search the matrix in order to find the room with the highest interaction with all other rooms and place it in the centre of the predetermined grid. Then, all other rooms were placed adjacent to each other based on their level of interaction.

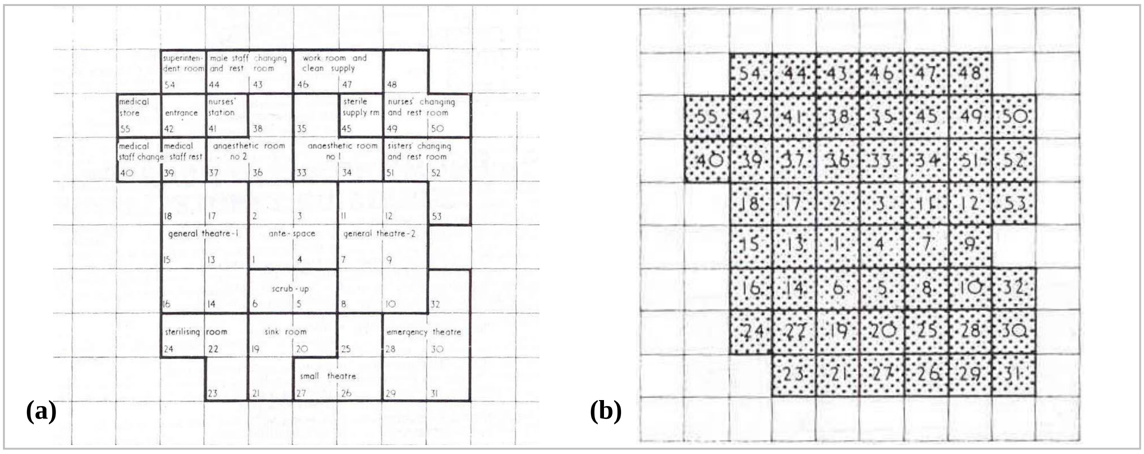


Figure 3. 3: a. Diagrammatic layout derived from the computer, b. Completed location matrix [Whitehead and Eldars, 1964]

The placing of new rooms was based on the idea of trial placing around the edges of the nest of rooms that have been placed, calculating the ‘cost’ that was the sum of the number of journeys multiplied by the distance between the new room and all the previously placed rooms. The new room was placed according to the least cost position, and the program continued to create a plan for the building, until all rooms were placed satisfactory as shown in figure 3.3.

Mitchell [1974, cited by Cross, 1977] and Steadman [1970, cited by Cross, 1977] determined the minimum cost house plans based on adjacency constrains. Using rules (in this case constrains) of dimensions, areas, and adjacencies, the main goal was the design of minimum cost houses that would satisfy constrains in an optimum manner.

Cross’ [1977] view was that such approaches are in the synthesis part of the design process because the main goal was the design of plans.

However, if we try to locate such an approach in Design Framework, it is obvious that this is not purely a creative process because evaluation is also included. The path-finding in the design space starts from the selection of a new area but it moves to evaluate the suitability of area in the design space using weighting functions as shown in figure 3.4.

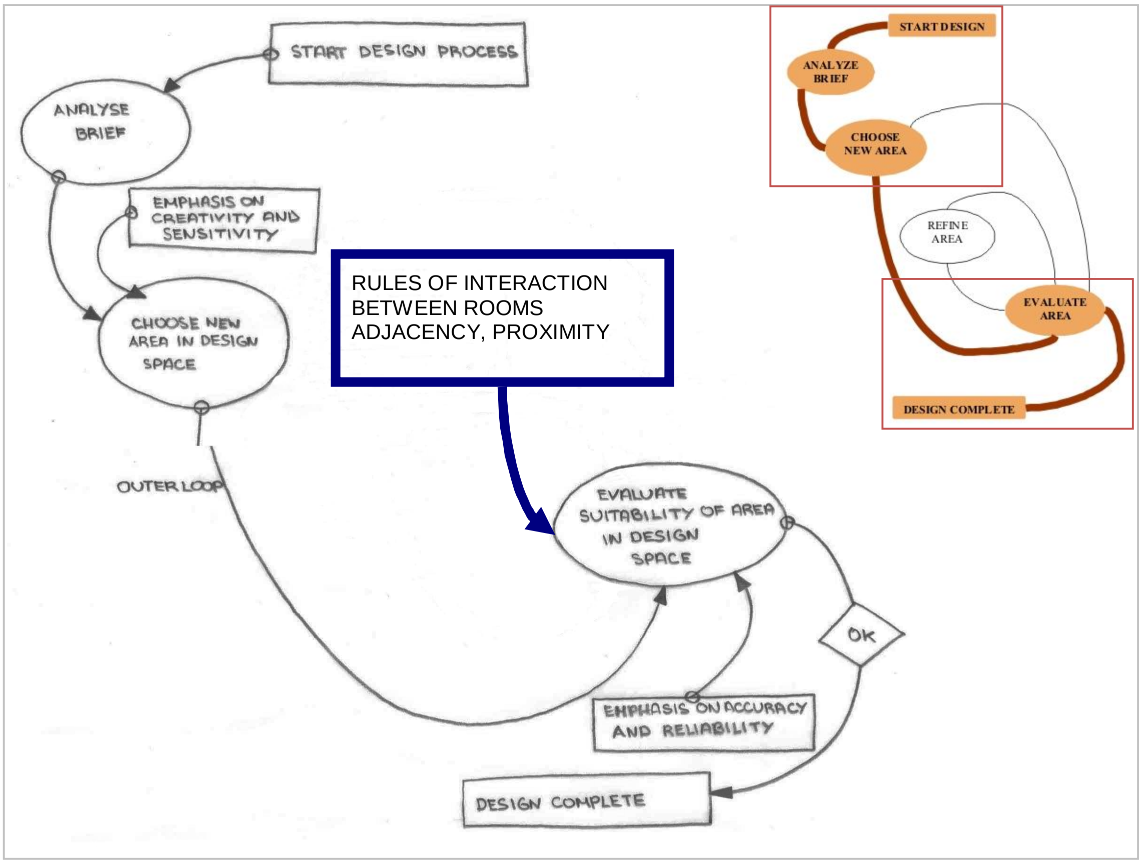


Figure 3. 4: Creativity and evaluation in design process

Here more emphasis is given to evaluation than to creativity. The process seems to be limited allowing little input of creativity and sensitivity. The design solutions are rigid allowing little flexibility in investigating alternative designs.

3.4 Creativity and shape rules

In order to provide design techniques that give more emphasis to creativity, architects are trying to find methods that will allow them to define their own criteria based on shape rules.

One such ‘subjective’ approach with the name ‘structural linguistics’ allowed architects to define their space-planning problem and program the computer to resolve it. It was mainly based on the concept of prototypic space patterns combined with linguistic means of communication with computers. More accurately, the linguistic tools consist of directions on how a sequence of compositions can be executed [Yessios, 1975] [Terzidis, 2006].

Other approaches include shape grammars and fractals. Both use rules to generate design solutions but lack mechanisms to evaluate them.

‘Shape Grammars’ were introduced by Stiny and Gips [1972] influenced by Chomsky [1957, cited by Stiny and Gips, 1972]. Stiny and Gips wrote:

> ‘Shape grammars are similar to phrase structure grammars, which were introduced by Chomsky in linguistics, where phrase structure grammars are defined over an alphabet of symbols and generate one-dimensional strings of symbols, shape grammars are defined over an alphabet of shapes and generate n-dimensional shapes. The definition of shape grammars follows the standard definition of phrase grammars’

Syntactic rules can be applied to generate new forms based on initial rules or they can be used to analyse existing building forms. Shape rules describe how shapes can be transformed into new shapes. Rules consist of an initial shape on the left hand-side of an arrow and a new shape on the right-hand side of the arrow. Each rule describes a transformation action from the initial shape to the new one. The rules can be applied repeatedly and create more complex shapes based on this shape rule. Figure 3.5 shows an example of shape grammar rules [for further details, see Stiny and Gips, 1972].

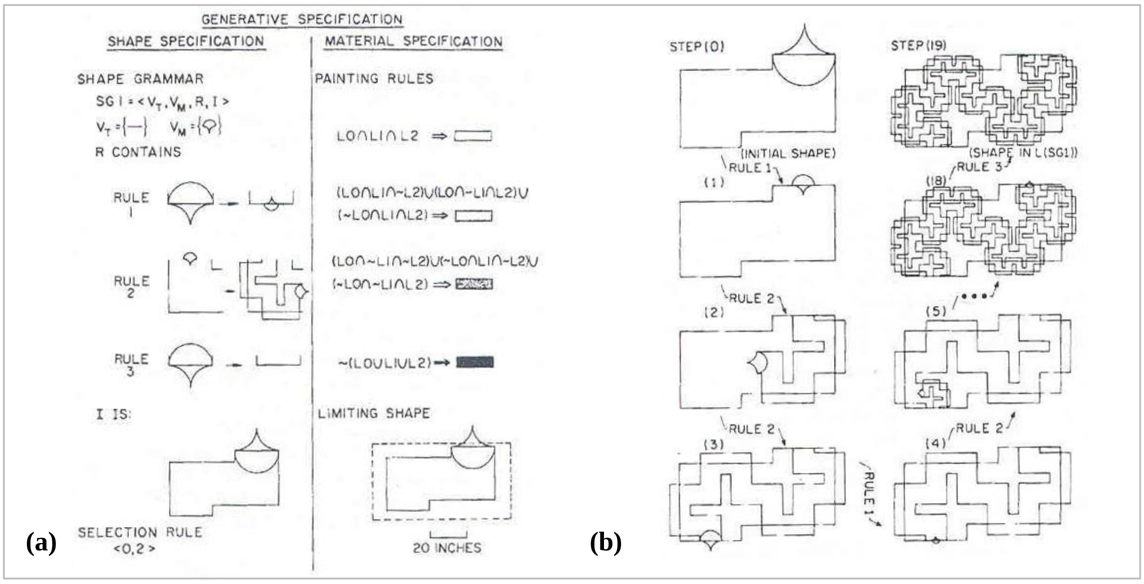


Figure 3. 5: Complete, generative specification of the class of paintings containing *Urform I, II, and III*, b. Generation of a shape using SG1 [Stiny and Gips, 1972]

Rule based approaches similar to shape grammars can be found in ‘fractal systems’ such as Lindenmayer or L-systems. First introduced by Aristid Lindernmayer (1925-1998), the systems are similar to ‘shape grammars’ (set of rules and symbols) and are used to model the growth process of the plant development. They consist of an initial seed cell or ‘axiom’ and a description of the growth rules or the ‘production rules’. Then, strings are created by taking the axiom and substituting the symbols according to the production rules [Flake, 1998].

The use of shape rules in the form of shape grammars or fractals has been found to be useful medium for explaining and generating architectural form due to similarities with architectural or city configuration structures. City and building plans can be interpreted as complex structures, suggesting shape rules are driving the logic behind these configurations [Bovil, 1996].

These approaches can be used in the creative part of the Design Framework as shown in figure 3.6. However, the process needs to pass through stages of ‘revision’ to find the most effective shape rules. Even if the process takes time, according to Flemming [1986] this will ‘lead to a deeper understanding of the issues at hand than is possible with traditional, more intuitive approaches’.

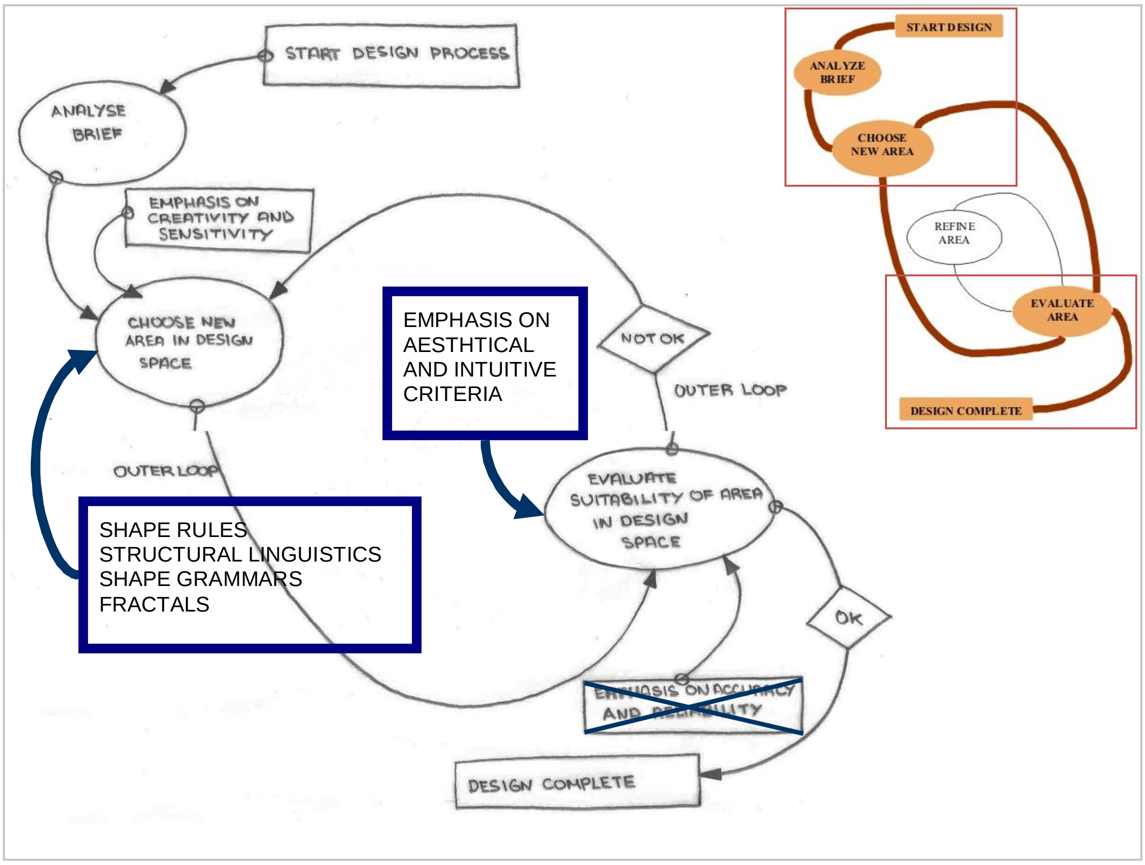


Figure 3. 6: Application of shape rules in Design Framework

The ‘revision’ is done intuitively and the design solutions are seen as ‘jumping’ in the design space to different locations without any mechanism of evaluation apart from aesthetic criteria.

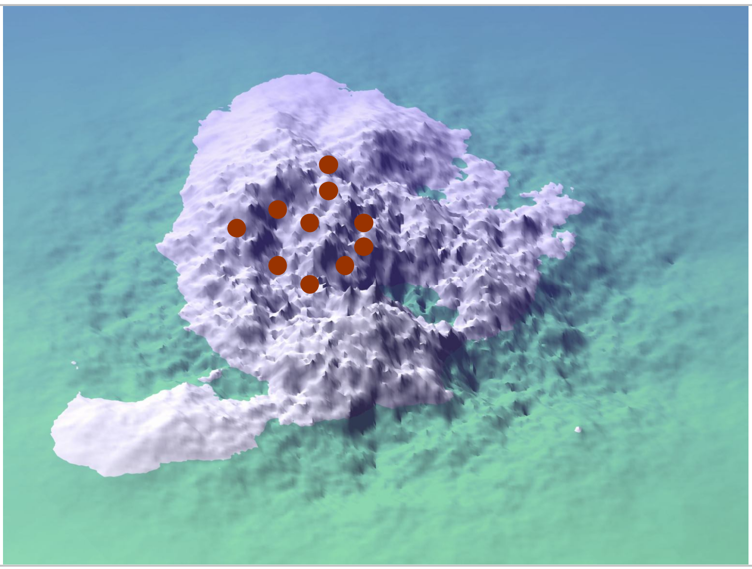


Figure 3. 7: Design solutions in design space based on shape rules

This causes the generation of a large number of solutions randomly located in the design space with different 'suitability ratings' as shown in figure 3.7.

3.5 Evaluation

The application of computers in the process of evaluating the suitability of an area in the design space is well established. In structural analysis is now normal for a computer to solve many thousands of simultaneous equations in a few seconds. This is just an 'ordinary computer' used at home or the office.

Evaluation mechanisms are used as the design is refined. Structural, environmental, pedestrian, or other analysis programs can be applied after the selection of 'design scheme'. However, this should not be taken to imply that the creative and the evaluation part of the Design Framework are independent without any integration or connection. The truly creative designer has an intellectual image of how he or she sees the building working and the evaluation is a confirmation or 'test' of this 'hypothesis'.

One of the drawbacks of computer analysis in areas such as structural behaviour is that a detailed computer model is required before the analysis can be done. This is in contrast to the 'back of an envelope' calculation that can be done using human intelligence.

Negroponte [1970] developed a more interactive computational approach allowing architects to communicate with the program while parts of the design were being completed. The program is named URBAN5, an 'urban design partner', because the architect and the machine cooperate for the production of building schemes.

The architect was able to ‘explain’ the design scheme to the machine as the design was in progress and the machine kept the designer informed of any conflicts and inconsistencies between the design criteria and the current state of the scheme. In the purely evaluation part of the process the program was able to measure important performance variables. One such variable was the production of patterns of pedestrian movement through public buildings, for instance airports. The performance of a building was predicted by simulating the behaviour of the building’s occupants in a given set of circumstances [Cross, 1977].

3.6 Parametric design

All design is parametric in the sense that various independent parameters control the design. In architecture the parameters include the function of the building (dwelling, concert hall, school, swimming pool) the size of the site and the amount of money available. However the phrase ‘parametric design’ is now taken to mean automated parametric design in which a computer program automatically updates the design as the parameters change.

A typical example may involve the length of a building’s frontage (L), the width of the cladding panels (w) and the number of cladding panels (n). These quantities are related by the rule $L = nw$ and the values of any two of the quantities determines the value of the third. Thus any two of the quantities can be chosen as independent parameters and the value of the third is not independent.

Clearly if the parameter ‘function’ were to change from ‘concert hall’ to ‘school’ it would be not be so easy for the design to be automatically updated. Thus parametric design is appropriate for detailed design, after the scheme design is complete and the design is more or less well defined.

Parametric design actually predates programs such as AutoCAD and firms like Ove Arup and Partners (now Arup) were using parametric design in the middle of 1970’s using punched cards as the input medium (reference: private communication from Williams, C.J.K.). It became established in the automobile and aerospace industries, and only recently has moved across to architecture. Parametric design does require the use of a scripting[2] or programming language to describe the logical structure. Alternatively the logical structure can be input and expressed graphically as, for example, in Bentley

---

[2] A ‘scripting’ language is used in conjunction with another program. A programming language produces a program which can run on its own. Computer code or source code are the human-readable statements that make up a computer program. Hence the verb ‘coding’ to mean computer programming.

System's GenerativeComponents. Thus architects are getting drawn into computer programming.

In mathematics a parameter is a quantity that can vary, causing other quantities to change their values. It can be confusing because what is described as a parameter may vary with the context. The following example is intended to show the ways in which parameters are used in mathematics. Figure 3.8b shows a family of superellipses (Lamé curves) that are defined by the parametric form as shown in figure 3.8a.

$$|x| = a|\cos\theta|^{\frac{n}{2}}$$

$$|y| = b|\sin\theta|^{\frac{n}{2}}$$

(a)

(b)

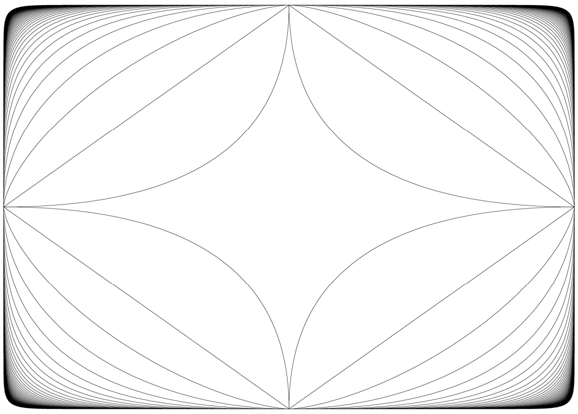


Figure 3. 8: a. Parametric forms of superellipses, b. Superellipses [Williams, 2008]

On a particular curve α , b and n are constant and the value of the parameter θ defines a point on that curve. In all the curves the constants α and b have the same value but the parameter n changes from curve to curve, starting at $n=0.5$ and increasing in increments of 0.5. The special case $n=1$ is a rhombus (or square if $a=b$) and $n=2$ is an ellipse (or circle if $a=b$).

n is constant on any one curve and if we change n , the curve changes shape. In parametric design we vary the parameter n to choose the particular curve that we want. Having made that choice, n in a sense ceases to be a parameter and is instead a constant. Having chosen the value of n we vary the value of the parameter θ to calculate the values of x and y to draw the curve.

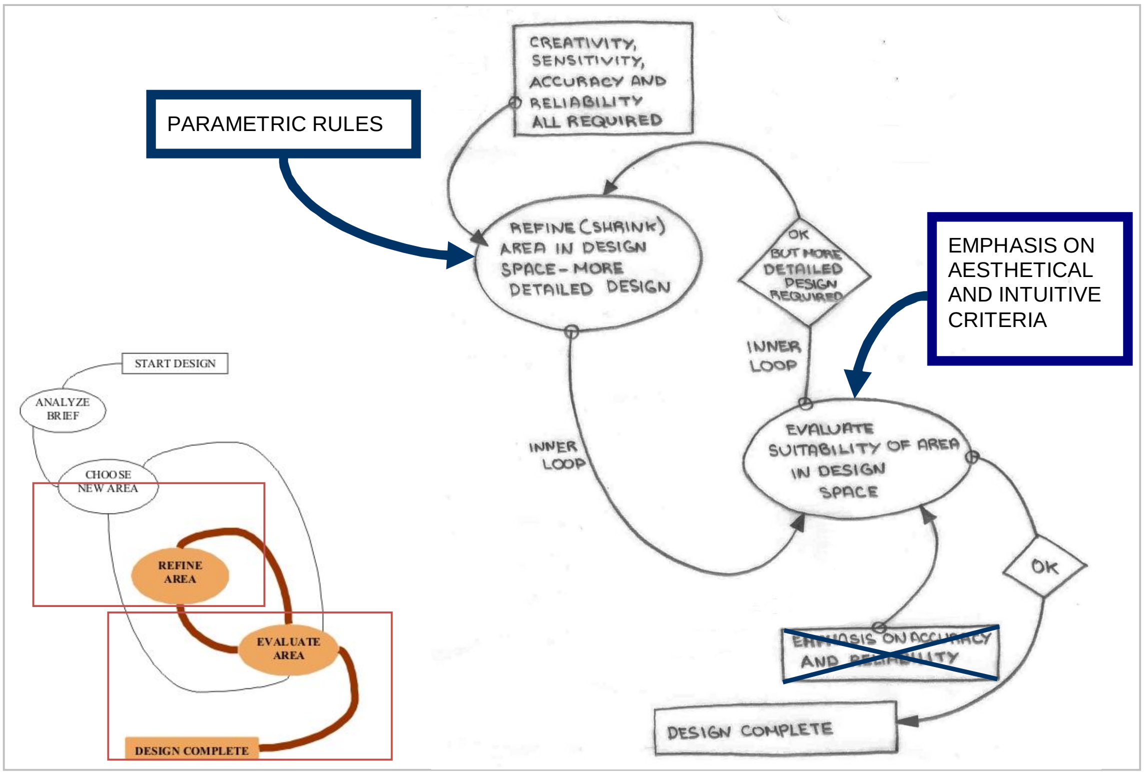


Figure 3. 9: Parametric design in Design Framework

Parametric design is in the inner 'detailed design' loop of our Design Framework as shown in figure 3.9. One of the current areas of interest in parametric design is the automation of the evaluation process, so that data for structural and environmental analysis is automatically generated.

3.7 Generative processes

A computer can be used to undertake various tasks, word processing, drawing (with or without parametric design) or structural analysis. In general the more 'mechanical' the process is, the more successful the application is.

Generative design processes are processes that create a 'new' design or at least a modification of same existing design. Unexpected patterns 'emerge' through the computational process. Architects and artists use generative processes to generate forms, to write music, and so on.

Iannis Xenakis[3] (1922-2002) was an architect and engineer who studied in Athens and then worked for Le Corbusier. But it is as a modernist composer that Xenakis is best known. He wrote computer programs to generate music that was then played by musicians with conventional instruments.

---

[3] Xenakis received many awards, including an honorary doctorate from the University of Bath in 1997, proposed by the Department of Architecture and Civil Engineering.

While Xenakis was working in Le Corbusier’s Paris Studio he designed the Philips Pavilion for the 1958 Brussels World Fair. Figure 3.10b shows Xenakis’ conceptual sketches.

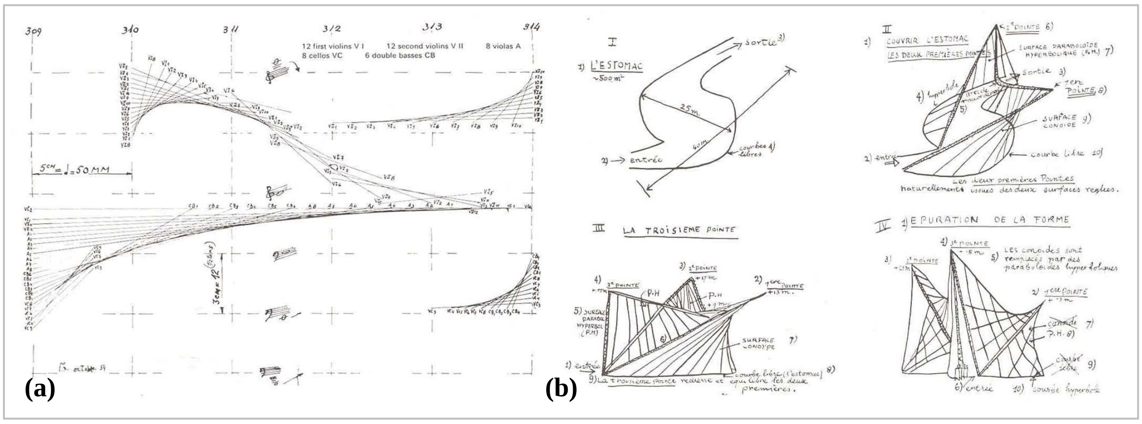


Figure 3. 10: a. Diagram of glissandi from Metastasis, 1954 by Iannis Xenakis, b. Conceptual sketches by Xenakis showing the development of the Pavilion’s geometry, 1956 [Treib, 1996]

The Pavilion was a hyperbolic structure based on the diagrams of glissandi from ‘Metastasis’ [Xenakis, 2001], one of his most famous musical pieces composed in 1954, see the diagram, figure 3.10a. Correlation between ‘Metastasis’ and the Pavilion can be found in Xenakis’ theory of meta-art. Xenakis [2001, p. 1] wrote:

> ‘Art, and above all, music has a fundamental function, which is to catalyze the sublimation that it can bring about through all means of expression. It must aim through fixations which are landmarks to draw towards a total exaltation in which the individual mingles, losing his consciousness in a truth immediate, rare, enormous, and perfect […] But this transmutation of every-day artistic materials which transform trivial products into meta-art is a secret. The possessed reach it without knowing its mechanisms’

The music of causal, according to Xenakis, was strongly influenced by the philosophical thinking of Pythagoras and Plato (Timaeus)[4]. James Harley [2004, p. 10] in *Xenakis: his life in music* writes about Xenakis’s influence and theory in musical composition:

> ‘At that time composers were grappling with the problem of how to create a new music. The serialist solution, derived from Arnold Schoenberg via Anon von Webern, was to design the shape of the composition from a generative cell, or series. This ‘organic’ approach, rather traditional from today’s perspective, despite all the avant-garde fervor and proselytizing of the time – can be contrasted with the principle of

---

[4] Discussion on reasoning can also be found in Plato’s (428BC-348BC) work *The Republic* [Lee, 2003, p. 235-240] in particular in the dialogue between Socrates and Glaucon in Chapter XXIV.

> juxtaposition, which Xenakis adapted from the architectural model of Le Corbusier. In this approach materials and forms are assembled according to relation established by the Modulor principle. At the same time, Xenakis was interested in dynamic processes or transformations. The title, Meta ('after, beyond') – stasis ('immobility'), refers to the contrast - or dialectic relationship – between movement, or change, and nondirectionality, or standstill'

Xenakis moved beyond 'static' approaches and he tried to find mechanism of creation based on dynamic principles. The creation of his musical pieces has been characterized as 'random' and 'stochastic' [Xenakis, 2001]. Harley [2004, p. 10] writes about 'Metastasis' and Xenakis's ideas of creation:

> 'The piece itself is utterly original. Xenakis's conception of originality supposes that creation must start from nothingness [Xenakis, 1994a cited by Harley, 2004]. The music begins on a sustained single note, as much out of nothing as is possible'

Xenakis's conception is similar to the idea of generative processes where a computer is used as a mechanism of 'new' creation. In the same way Xenakis used the computer as a 'tool' to help him generate his music but the computer did not generate the music on its own.

Nature has inspired architects, mathematicians, engineers and computer scientists. A number of computational methods have been developed mimicking natural mechanisms of growing or adapting to the environment. Such techniques might assist design in a more active way than conventional approaches.

The possibility of applying these techniques to architectural design raises the question as to whether architecture can be considered as part of the natural environment.

Philip Steadman [1979] in *The evolution of designs: Biological analogy in architecture and the applied arts* investigated the relationship between biological analogy and architecture. Taking examples from Antoni Gaudi, Luis Sullivan and Le Corbusier Steadman stated that organic analogy is a central theme in design in the twentieth century.

More than thirty years ago Charles Jencks, [1971] in his book *Architecture 2000: Predictions and Methods,* wrote :

> 'The biological age will have great influence on architects, will provide them with specific tools and analogies and be taken by 'intuitive' tradition to become the *Biomorphic School'*

Figure 3.11 shows an evolutionary tree to the year 2000 created by Jencks [1971]. In this image the word ‘Biomorphic’ is indicated in a circle [see Jencks, 1971, for details].

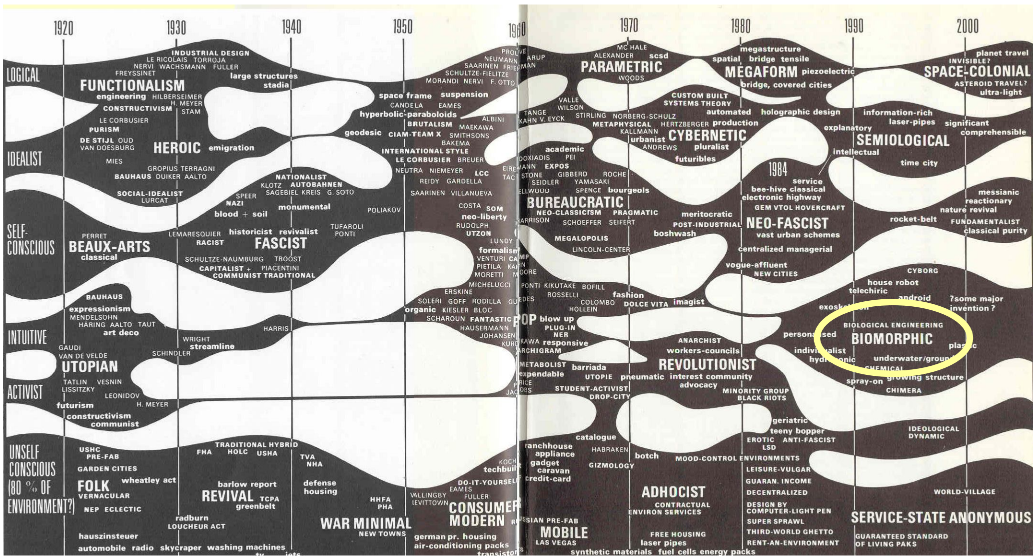


Figure 3. 11: Evolutionary tree to the year 2000 [Jencks, 1971]

This idea has been further developed and digitalized by architectural theorists like John Frazer [1995, p.9] whose pioneering work in the area of evolutionary design has opened new possibilities in the area of computer aided design. He stated that:

> ‘Architecture is considered as a form of artificial life, subject, like the natural world, to principles of morphogenesis, genetic coding, replication and selection. The aim of an evolutionary architecture is to achieve in the built environment the symbiotic behaviour and metabolic balance that are characteristic of the natural environment […] Architectural concepts are expressed as generative rules so that their evolution may be accelerated and tested. The rules are described in a genetic language which produces a code-script of instructions for form-generation […] Computer models are used to simulate the development of prototypical forms which are then evaluated on the basis of their performance in a simulated environment. Very large numbers of evolutionary steps can be generated in a short space of time, and the emergent forms are often unexpected’

Scientists, engineers, architects and artists introduced such generative mechanisms in their work, either as creative or as evaluation mechanisms, since the results are not obvious or predicted, might be ‘new’ or ‘emergent’ and they exhibit complex behaviour.

3.7.1 Complex systems

The term ‘complex systems’ is used in diverse disciplines and its definition varies according to the field. An interesting definition can be found in the website of the

Australian Research Council [http://www.arc.gov.au] since it covers almost all aspects of our world:

> ‘Real-world systems are almost always made up from a large number of interacting components. This leads to complex behaviour that is difficult to understand, predict and manage. Research on complex behaviour is often undertaken by mathematicians, statisticians, engineers, and information and computer scientists. It will contribute to improvements in areas such as the internet, air traffic and transportation control, power systems, robotics, irrigation and land management, defence, manufacturing and finance, as well as ecology and biology’

The Santa Fe Institute [Bonabeau et al., 1999] is an organization dedicated to the study of complex systems applying such methods to a range of fields from economics to biology and physics. Also, institutes like Bath Institute of Complex Systems (BICS) at the University of Bath investigate complex behaviour taking examples from research areas including mathematics, engineering, and computer science.

What are common to all these areas are the properties that accompany complex systems. Herbert Simon [1996, p. 183-184] in *The science of the artificial* explained that:

> ‘I shall not undertake of formal definition of ‘complex systems’. Roughly, by complex systems I mean one made up of a large number of parts that have many interactions. As we saw in the last chapter, in such systems the whole is more than the sum of the parts in the weak but important pragmatic sense that, given the properties of the parts and the laws of their interaction, it is not a trivial matter to infer the properties of the whole’

The characteristics of complex systems can be summarized in the following categories [Goldstein, 1999, p. 55]:

i. Nonlinearity – ‘small cause, large effect’ and nonlinear interactivity

ii. Self-organization – creative, self-generated, adaptable behaviour

iii. Beyond equilibrium – amplification of random events

iv. Attractors – fixed point, limit cycle, and strange attractor

According to Goldstein [1999], self-organization in complex systems refers ‘to the creative, self-generated, adaptability-seeking behaviour of a complex system. Emergent phenomena are novel structures that confer this adaptability’.

3.7.2 Emergence

Williams and Kontovourkis [2008] in *Practical Emergence* make comparisons between ‘emergency’ and ‘emergence’. They wrote:

> ‘The Latin mergo, mergence, mersi, mersum means to dip, plunge into a liquid, immerse. From ex mergo we have emergo, emergence, emersi, emersum: to rise up, to free oneself, to come to light, appear, emerge. Nuttall’s Standard Dictionary [Wood, 1916] gives the two words ‘emergence’ and ‘emergency’ the same definition: a sudden appearance; an unexpected event; exigence; pressing necessity’

Moving from the general to the specific, a definition of the word ‘emergence’ in complex systems is given in Jeffrey Goldstein’s [1999, p.49] work *Emergence as a Construct: History and Issues* in the first edition of the journal *Emergence.* Goldstein wrote:

> ‘Emergence, as in the title of this new journal, refers to the arising of novel and coherent structures, patterns, and properties during the process of self-organization in complex systems. Emergent phenomena are conceptualized as occurring on the macro level, in contrast to the micro-level components and processes out of which they arise. In a wider variety of scientific and mathematical fields, grouped together loosely under the title ‘complex theory’, an intense search is now under way for characteristics and laws associated with emergent phenomena observed across different types of complex system’

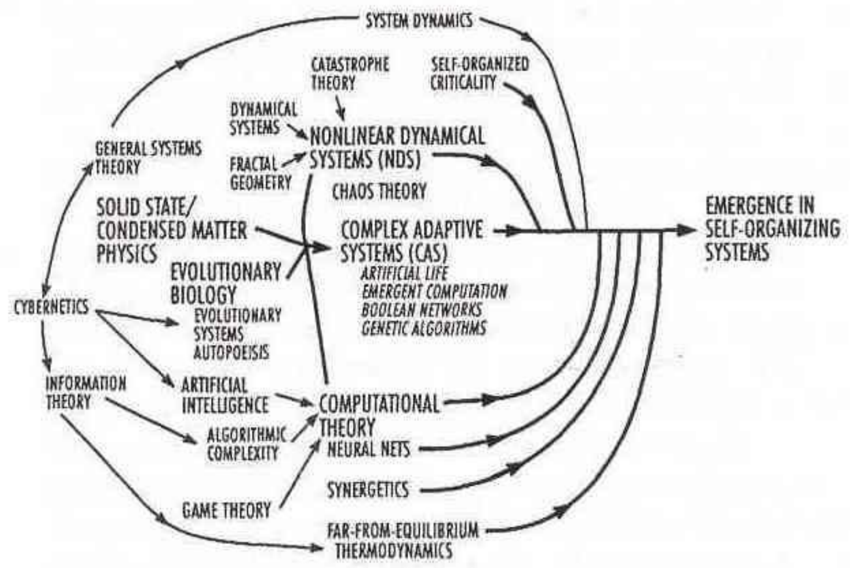


Figure 3. 12: Mathematical and scientific roots of emergence [Goldstein, 1999]

Emergent phenomena can be found in different areas. Figure 3.12 shows a diagram of the mathematical and scientific roots of emergence in self-organizing systems as presented by Goldstein [1999].

Either in physical systems or in computer simulation, emergent phenomena share certain interrelated and common properties that identify them as emergent [Goldstein, 1999]. Such properties include:

i. Radical novelty – features are neither predictable nor deducible from lower or micro-level components

ii. Coherence or correlation – coherence spans and correlates the separate lower-level components into a higher-level unity

iii. Global or macro level – observation of emergent is in macro level

iv. Dynamical – emergent phenomena evolves over time

v. Ostensive – ostensive qualities

Williams and Kontovourkis [2008] gave a number of examples using various computational techniques that produce such emergent properties. These included the flow of a fluid past a cylinder (see figure 3.13) simulated using the method known as Smoothed Particle Hydrodynamics (SPH) [Monaghan, 2005].

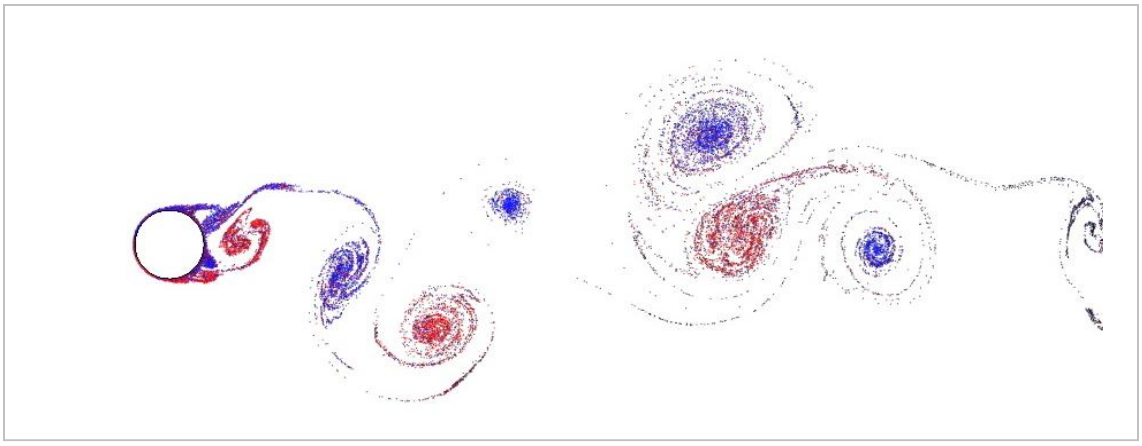


Figure 3. 13: Flow past a cylinder computed using smoothed particle hydrodynamics [Williams and Kontovourkis, 2008]

Each particle used in Smoothed Particle Hydrodynamics represents many air molecules and is given more sophisticated behaviour than that of the molecules in the kinetic theory of gases. Each particle is ‘aware’ of its neighbours and moves in a complicated way in an attempt to model the effect of pressure and viscosity [Williams and Kontovourkis, 2008].

Other systems that exhibit emergence properties can be found in Complex Adaptive Systems (CAS) as shown in the diagram of figure 3.12. Complex Adaptive Systems are special types of complex system that are diverse and made up of multiple interconnected elements that are adaptive in that they have the capacity to change and learn from experience. Examples of Complex Adaptive Systems include Genetic Algorithms and Artificial Life and these techniques have been applied in design to generate 'new', 'unexpected' design solutions.

3.7.3 Genetic algorithms

Charles Darwin (1809-1882) in his work *The origin of species*[5] [1968] and Alfred Russel Wallace (1823-1913) independently introduced the theory of natural selection in 1858. Natural selection is a process by which biological populations are changed over time, as a result of the propagation of heritable traits that affect the capacity of individual organisms to survive and reproduce.

Earlier theories included that of Jean-Baptiste Lamarck (1744-1829) [Burkhardt, 1977, p. 145] who believed that evolution happens according to a predetermined plan and any changes of the population during its life in order to adapt to the environment is passed on to the offspring. However, Lamarck's theory has been abandoned.

Figure 3. 14: Evolution of the Volkswagen Golf, 1974 to 2003 [Available from numerous internet sites]

---

[5] First publish by John Murray in 1859.

Figure 3.14 shows the ‘evolution’ of the Volkswagen Golf over 30 years, starting in 1974. The evolution is more Lamarkian than Darwinian in that experience with one model is passed on to its successor. Competition with other manufacturers (Ford Escort, Focus, Peugeot 306, 307, 308) is intense meaning that each model must be better (or perceived to be so by the consumer) than its predecessor. The family resemblance between the current Golf (foreground) and its great grandparent (far left in the background) is not very obvious, but when each model is compared with its parent, the ‘genetic’ inheritance is obvious. However, the Golf’s genetic make-up is based more on Alec Issigonis’ BMC Mini (front wheel drive, front transverse engine) than Ferdinand Porsche’s Volkswagen Beetle (rear wheel drive, rear engine).

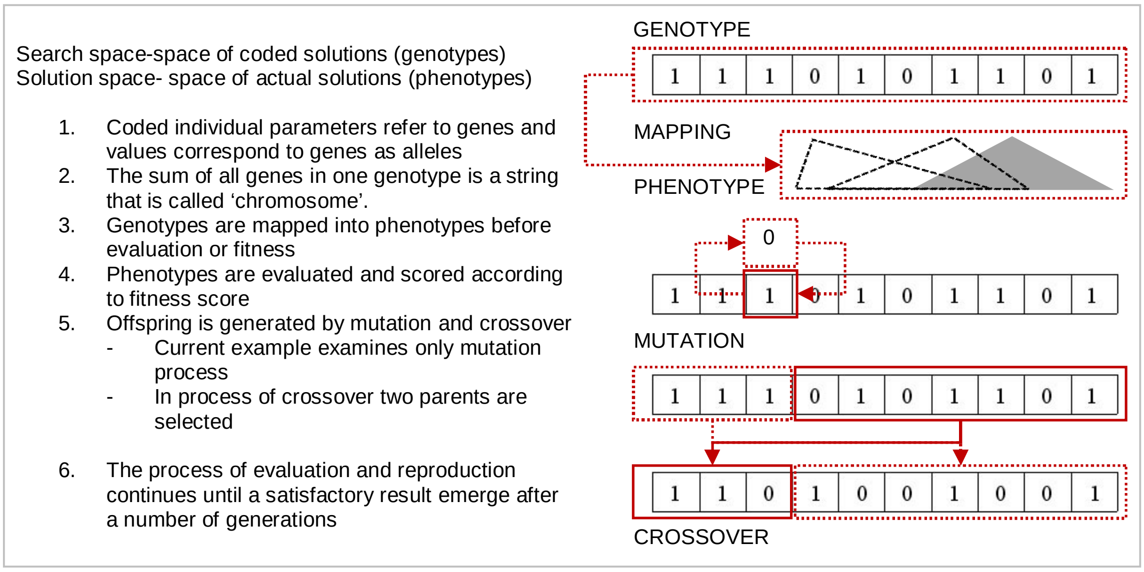
Living organisms are many times more complex and sophisticated than a car or any other man-made object including computers. In the *Blind Watchmaker* Richard Dawkins [1986] describes how Darwinian natural selection can explain the evolution of such complex organisms. The title is chosen because the book is a refutation of William Paley’s (1743-1805) watchmaker analogy, a teleological argument for the existence of God based on order and design in nature.

Genetic Algorithms are computational techniques that have been developed using the Darwinian idea of natural selection where traits are inherited from the parents to offspring and cannot be affected by the environment. The technique appears in the broader category of Complex Adaptive Systems [Goldstein, 1999] since it exhibits emergent properties.

Although computer simulation of evolution started in the 1950s, Genetic Algorithms were not introduced until 1975 in John Holland’s [1975, 1992] work *Adaptation in Natural and Artificial Systems*. Genetic Algorithms are search algorithms that can map any data, design, etc. into ‘virtual’ DNA structures. This structure is the population that can produce offspring, can cross over, mutate and share fitness – increasing favourable traits like a real population found in nature [Flake, 1998]. However it must be emphasised that Genetic Algorithms require a technique for automatically comparing any two candidate solutions and deciding which is better. This is not very meaningful when the two possibilities are very different (chalk and cheese), or when subjective criteria are involved.

Figure 3.15 demonstrates the basic steps of a simple Genetic Algorithm that maximizes the area of a triangle with a given perimeter. The algorithm was written by the author

[Kontovourkis, 2006] and the computer code with explanation can be found in Appendix A.



Search space-space of coded solutions (genotypes)
Solution space- space of actual solutions (phenotypes)

1. Coded individual parameters refer to genes and values correspond to genes as alleles
2. The sum of all genes in one genotype is a string that is called ‘chromosome’.
3. Genotypes are mapped into phenotypes before evaluation or fitness
4. Phenotypes are evaluated and scored according to fitness score
5. Offspring is generated by mutation and crossover
    - Current example examines only mutation process
    - In process of crossover two parents are selected

6. The process of evaluation and reproduction continues until a satisfactory result emerge after a number of generations

Figure 3. 15: Basic steps of Genetic Algorithm that maximizes the area of a triangle [Similar descriptions are available from numerous sources (books, articles, internet, etc) discussing GA]

The results show that Genetic Algorithms are very powerful optimization tools for analysing a population of solutions in design space with different ‘suitability ratings’ according to fitness criteria. However, the population of emergent solutions occupies a certain area in the design space and cannot be characterized as ‘new’ or ‘unpredictable’.

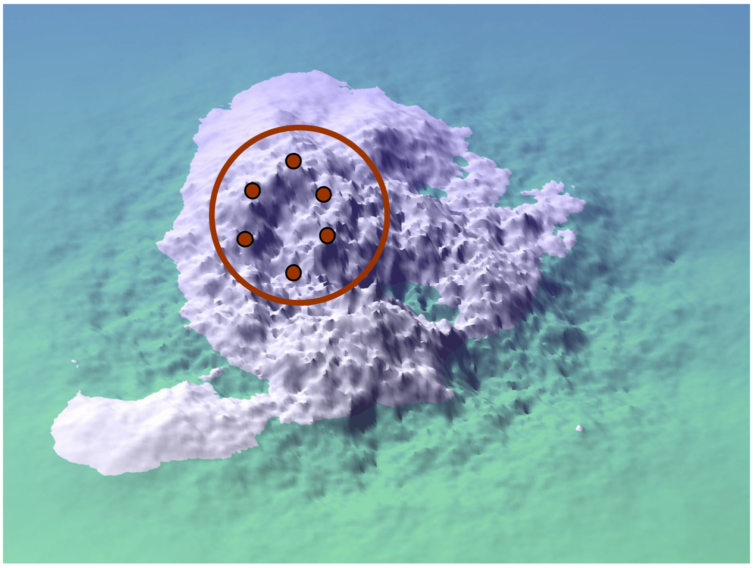


Figure 3. 16: Design solutions in the design space based on genetic algorithms

In nature the evolution of whales from early small land-based mammals is rather surprising, but one cannot imagine a computer program written for analysing the fitness of mice for survival to also include the possibility of whales.

Even if Genetic Algorithms give the impression of breeding new forms the conceptual stage of design needs to be done intuitively. Manuel DeLanda [2002] in *Deleuze and the use of the genetic algorithm in architecture* argued:

> 'The space of possible designs that the algorithm searches needs to be sufficiently rich for the evolutionary results to be truly surprising'

The important part in this procedure is the production of 'virtual' DNA. As DeLanda stated the design solutions need to be treated as 'reproductive communities':

> 'More technically, the idea is that despite the fact that at any one time an evolve form is realized in individual organism, the population, not the individual, is the matrix for the production of form. A given animal or plant architecture evolves slowly as genes propagate in a population, at different rates and at different times, so that the new form is slowly synthesized within the larger reproductive community'

For DeLanda this is 'a population thinking', a philosophical concept that traced to the work of Gilles Deleuze together with other philosophical and mathematical terms like 'intensive' and 'topological'.

3.7.4 Complex systems in architecture

The term 'molecular population' can be found in Deleuze and Guattari's [1987, p. 380] work *A thousand plateau: Capitalism and schizophrenia.* Deleuze and Guattari wrote:

> 'Material thus has three principal characteristics: it is a molecularized matter; it has relation to forces to be harnessed; and it is defined by the operations of consistency applied to it. Finally, it is clear that the relation to the earth and the people has changed, and is no longer of the romantic type. The earth is now at its most deterritorialized: not only a point in a galaxy, but one galaxy among others. The people is now at its most molecularized: a molecular population, a people of oscillators as so many forces of interaction'

Clearly the term 'population' can be found in complex systems where techniques try to simulate the interaction behaviour between 'automata' and their environment at a local level and assess the behaviour of the system as a whole.

One such example is Cellular Automata. John von Neumann (1903-1957) and Stanislaw Ulam (1909-1984) were the first to examine such systems at the Los Alamos Laboratory, New Mexico, in 1940's. The Los Alamos Laboratory was founded to develop the atomic bomb, the Manhattan Project. Cellular Automata form a dynamical system that is discrete in both space and time [Flake, 1998]. It consists of infinite, regular grid of cells, each in one of a finite number of states. Every cell has the same rule for updating and each time step a new generation is produced. Cellular Automata have been used in architecture to explore the growing potential and the spatial organization of cities [Batty, 2003; 2005], the development of forms, buildings, etc. [Krawczyk, 2002].

The Cellular Automata technique predates other simulation methods like Artificial Life. In Artificial Life the cells are replaced by ‘agents’ and the systems are not discrete in space but continuous. The distinction between discrete and continuous is not clear since as a grid is refined it approaches the continuous. Also, numbers in a computer are only stored to a finite accuracy and therefore all calculations are discrete. In Artificial Life stepping through time remains discrete with the time step chosen according how quickly things change.

Artificial Life examines man-made systems that exhibit the behavioural characteristics of living systems like flock of birds and ant colonies. Artificial Life use a ‘bottom up’ approach in contrast to Artificial Intelligence where a ‘top down’ approach is used.

Artificial Life examines global ‘emergent organizational structures’ as results of the interaction between automata and their environment at a local level. A ‘real life’ example (see figure 3.17) is the foraging pattern produced by the army ant species *eciton hamatum* (left) and *eciton burchelli* (right) [Bonabeau et al., 1999] [Otto and Rasch, 1995]. The ant shows a collective intelligence that exhibits emergent behavior.

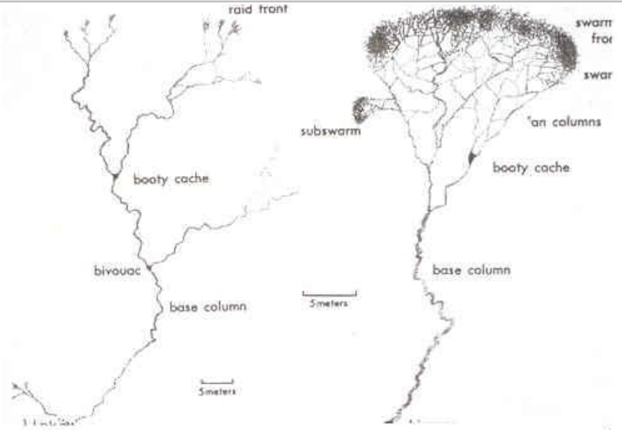


Figure 3. 17: Booty-seeking ants’ path systems [Otto and Rasch, 1995]

Artificial Life phenomena can be best simulated using parallel computers where numbers of automata can perform their actions at the same time. One such software tool is Star Logo [Resnick, 1997] which has been used in diverse areas like chemistry, economics, political science, geometry, and social systems.

Paul Coates, head of the Centre for Evolutionary Computing in Architecture (CECA) at University of East London applied Star Logo software in architectural design for the generation of emergent movement patterns, land use or configuration analysis of space. Specifically, their examples tried to model the way arbitrary agglomerations of space are trafficked by users [Coates and Schmid, 2000].

Swarm Intelligence is an Artificial Life technique based on decentralized, self-organized systems. It was mainly introduced in the field of cellular robotic systems, though it gradually has been applied in different areas, such as sociology.

Neil Leach [2004] in *Swarm Tectonics* drew parallels between Swarm Intelligence and structural design by characterizing structures as ‘self-organized systems’. He wrote:

> ‘Yet, once we reinterpret the computer, not as a nomadic[6] machine, but as a ‘population’ of smaller, nomadic components operating within the logic of swarm intelligence, the possibility becomes more evident. Here we might recognize that structures themselves operate in a highly complex manner. Never as discrete and self-contained as they first appear, structures operate parametrically as ‘self-organized systems’, It would be better to think of their operations in terms of networks or even meshworks’

The potential application of such systems in structural design is demonstrated by Kristina Shea’s [2004] eifForm program in which the design is developed through the repeated modification of an existing design aiming to improve its performance taking into account different factors like structural efficiency, economic of materials, and aesthetics. Structure is ‘self-organized’ since, as Leach [2004] wrote: ‘the ‘designer’ merely established certain defining coordinates, and then unleashes the program which eventually ‘crystallizes’ and resolves itself into a certain configuration’.

In architecture, Carranza and Coates [2000] used Swarm Intelligence techniques to study the interaction of swarm automata with their environment. In their investigation concepts like ‘structural coupling’ were introduced and investigated. They described ‘structural coupling’ as:

> ‘an ongoing mutual co-adaptation without allusion to a transfer of some ephemeral force or information across the boundaries of the engaged systems’

---

[6] The term ‘nomad’ is used by Leach following Deleuze [1993] in *The fold: Leibniz and the baroque.* Nomad an anagram of the term ‘monad’ used by Leibniz. ‘Nomad’ in Greek is translated as ‘νομάδα: ομάδα ανθρώπων, κυρίως κτηνοτρόφων που δεν κατοικούν καπου μονιμα αλλα μετακινούνται απο τόπο σε τόπο για να εξασφαλίσουν βοσκή για τα κοπάδια τους’ and ‘monad’ means ‘μονάδα: καθέ χωριστό στοιχείο ενος συνόλου το οποίο έχει τη δυνατότητα να λειτουργεί αυτόνομα’.

and continued:

> ‘Inside this framework it is interesting to observe is how different ‘forms’ are described by the structural coupling of the automaton and an environment’

Their automata had access to the whole scene’s geometric description and reacted only to flock mates within a certain radius. Swarm automata were encoded based on the computer model of the co-ordinated animal motions of ‘boids’ that were reacting to the geometrical environment through a collision detection algorithm [Reynolds, 1987]. The model was based on simple flocking rules like ‘separation’, ‘cohesion’, ‘alignment’, and additional rule of ‘obstacle avoidance’. Reynolds’ technique is discussed analytically in section 4.4.4

The emergent results showed that:

> ‘The individual agents had a tendency to align with the surfaces of the geometric model of the site. This ended in the emergence of the ‘smoothest’ trajectory on the environment, which in the case of the test model of a site where the meanders of a river. He swarm is able to discriminate the edges of a long wide curvy grove, that is, the geometric form of the river, from any other information such as buildings or building groups or infrastructures’

Figure 3.18a shows a diagram of the swarm.

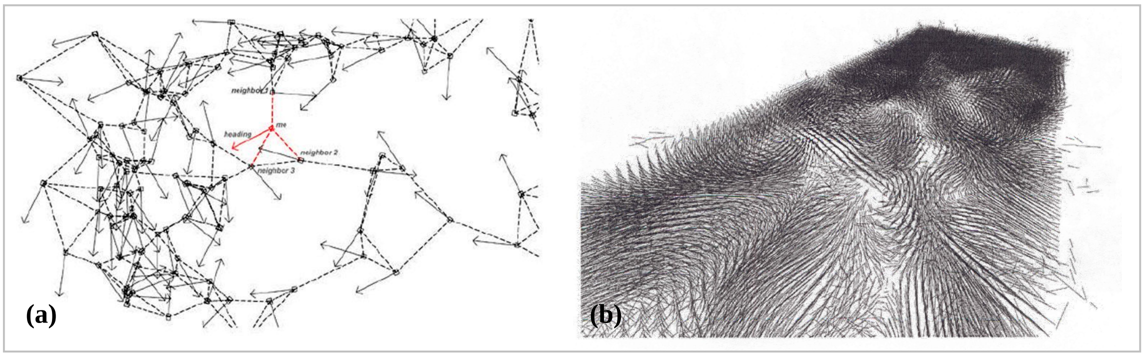


Figure 3. 18: a. Diagram of the swarm. Arrows represent each agent’s heading, dotted lines their closest neighbors, b. Gradients created by sematectonic process [Carranza and Coates, 2000]

‘Stigmergy’ is the mechanism in which a trace left in the environment by an action stimulates the performance of a subsequent action, by the same or a different agent. Stigmergy was first observed in social insects, for example, ants exchange information by laying down pheromones on their way back to the nest when they have found food. The term was introduced by French biologist Pierre-Paul Grassé [1959] to refer to termite behaviour. It is derived from the Greek words stigma ‘sign’ and ergon ‘action’.

The laying down of pheromones is ‘sign-based’ whereas ‘Sematectonic stigmergy’ involves a change in the physical environment, for example nest building where an ant observes a structure developing and adds its ball of mud to the top.

In another case study, again by Carranza and Coates [2000], the relationships between ants, networks and learning swarms were examined. In this example the ability of social insects to learn through their environment was investigated. The process involved ‘sematectonic communication’ and the individuals were interacting not with other individuals but with the environment, which in turn modified the behaviour of other individuals. Figure 3.18b shows gradients created by a sematectonic process. Such systems showed higher complexity than flock models where correlations with networks can be found [Carranza and Coates, 2000].

Krause [1997] described a process to generate architectural patterns using behaviour based Artificial Intelligence. The individuals or design elements were represented as points, polygons, and so on. There were three different types of movement: wander agents move locally, path agents move along a path, and hyper agents move from point to point through the environment. Their behaviour within the system was controlled by applying different rules of interaction or communication. Rules of growth, decay, transformation, social, etc. were applied and results emerged through interaction or communication (sharing the same space). The emergent results showed that highly complex forms were generated from a large number of interactions at a local level between agents in a chaotic environment. The technique was suggested as form generator for the autonomous emergence of architectural principles.

Scheurer [2005] used an agent based system to define the configuration of randomly positioned columns in a large concrete structure. The design problem was divided into known and unknown information. The known information included outlines of the slab, the distance between the floor and the slab, centre lines of the defined walking and cycling paths. The unknown information was the numbers of columns and their minimum diameters, the cost saving, the position of columns in different parts of the structure, such as at the edges, etc.

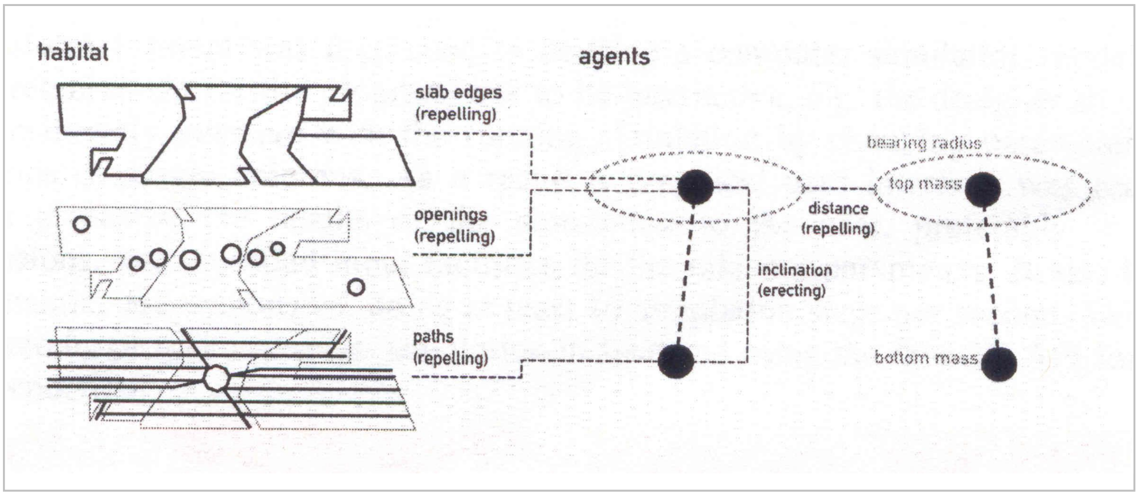


Figure 3. 19: Relations between agents and environment [Scheurer, 2005]

Agents were represented by the columns moving freely within the horizontal planes of the floor and the slab. Their aim was to keep an optimal distance regarding the local spanning capacity of the slab and to keep clear the walking and the cycling paths. Figure 3.19 shows the relation between the agents and the environment.

The dynamic behaviour of the agents was defined by a particle simulation with springs, attractors, and repellers. The agents (columns) were initially distributed randomly, growing or splitting according to their position. The parameters controlling the behaviour could be changed, the strength of repelling force, minimum distance from the path lines etc., thus influencing the spatial arrangement of the system.

3.7.5 Deleuze and the generative processes

Deleuze does not refer directly to any form of digital design process but his philosophical thinking offers inspiration for architects, architectural theorists and artists.

Ballantyne [2007, p. 9] in *Deleuze & Guattari for Architects* wrote:

> ‘What they can help us to do is to keep common sense at bay, take us on a witches’ ride, so that we can see the world as unformed elements acted upon anonymous forces, and by entering into this world of virtualities, it can put us in a position where we might actualize something that has not previously been’

Deleuze and Guattari’s term ‘Virtual’ has been discussed by architects [Lynn, 1999]. ‘Virtual’ can be understood as the abstract conceptual level of design creation that plays an important role in the overall design process, especially in the ‘creative’ part of our Design Framework. It is the abstract representation of ideas or the ‘abstract machine’, according to Deleuze and Guattari [1987].

Deleuze’s analysis of the relationship between the virtual and the real is discussed in *Flying Dutchmen*: *Motion in Architecture* [Jormakka, 2002]:

> ‘The movement from the virtual to the real is creative, for ‘while the real is the image and likeness of the possible that it realizes, the actual [...] does not resemble the virtuality that it embodies’. With no preformed order to dictate the form, the actualization of virtual beings is a creative evolution, an original differentiation of organization. The differentiation is never a negation but a positive creation. Virtual beings as unity unfold and reveals its real multiple differences; differentiation is a temporal process’

If computers are applied in design process, this creative evolution can only be captured by the application of systems that can offer the potential to move from ‘the nothingness to the creation’, Xenakis, quoted by Harley [1994]. Generative processes might offer such emergent design possibilities.

Ballantyne [2007, p.36] found connections between emergent design and Deleuze and Guattari’s philosophical thinking about the ‘body without organs’. Ballantyne wrote:

> ‘The body without organs is a state of creativity, where preconceptions are set aside. It is the state before a design takes shape, where all possibilities are immanent, and one holds at bay the common – sense expectations of what the design should be. When a stimulus or an internal pain prompts a line of flight, then formations assembly, giving the beginnings of a form – a structure, a detail, a *leitmotif*. The aim could be that the design would be entirely immanent in its initial condition, and would emerge as a product of the various forces in play in the milieu. It would not be imposed from outside as a specific form, but would work with the grain of its matter, from within, but also seamless with the milieu and networks extending to its horizon. It can crystallize in various ways, at a molecular level, to aggregate and produce different surface effect when it becomes apparent to the senses in a wider world’

This paragraph does not refer to the application of generative processes in design directly. However, the concept of emergent design that is the result of an internal organization of a system or ‘population’ at a molecular level might be connected with the application of such generative mechanisms at the conceptual design level.

In order to introduce ‘technologies’ in design and achieve the translation from ‘the virtual into the concrete’, according to Greg Lynn [1999, p.40] in *Animate Form*, first we need to ‘interrogate their abstract structure’.

> 'Without a detailed understanding of their performance as diagrams and organizational techniques it is impossible to begin a discussion of their translation into architectural form'

3.8 Generative processes and the Design Framework

An 'active' or 'generative' tool in the design process is a design mechanism that can be represented as a feedback loop. In such a cyclic procedure the investigation moves from the design creation to the evaluation of solutions repeatedly in a short time period providing a large number of alternative solutions.

Techniques based on 'complex systems' can be incorporated in our Design Framework in the 'evaluation' part causing the automatic movement of the design to a new position in the design space as shown in figure 3.20.

Such processes alternate between the selection of the new area in the design space and the evaluation of the design area based on simple 'local' rules till the design solution finds a satisfactory position in the design space. This process clearly combines both parts; the one that emphasizes 'creativity' and the other that emphasizes 'accuracy'.

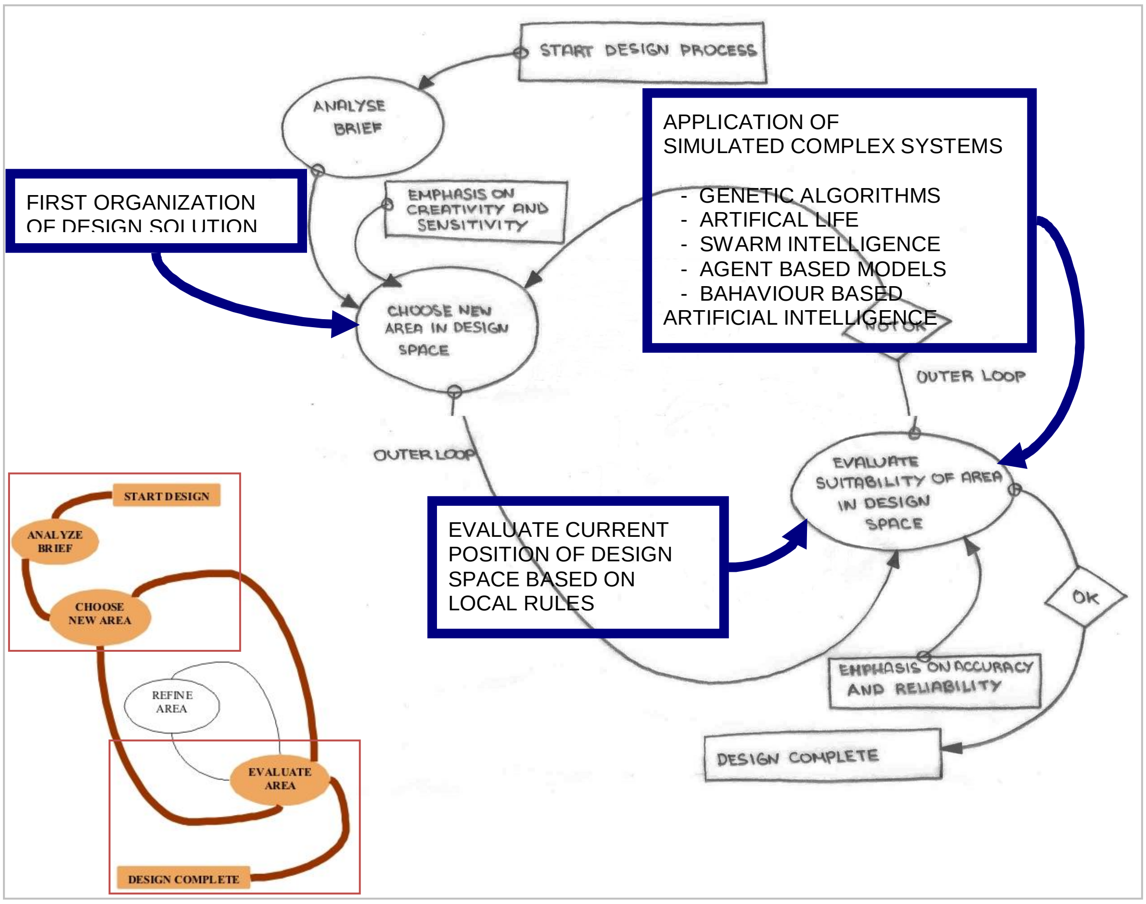


Figure 3. 20: Application of complex adaptive systems in Design Framework

Figure 3.21 shows a representation of how the concepts of emergence and artificial life are incorporated in our Design Framework. Architects and engineers act as agents that navigate design space according to rules in which the agents respond to the 'environment', that is objective criteria, but also to the behaviour of others agents.

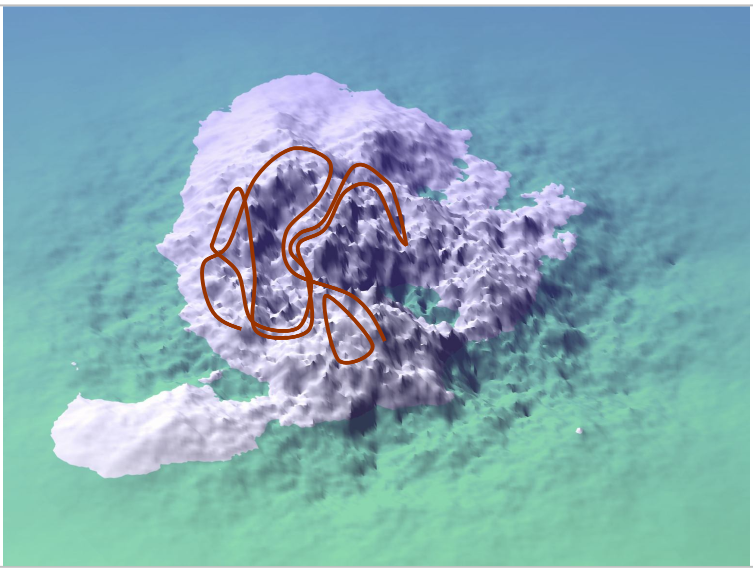


Figure 3. 21: Path finding in design space based on emergence

This chapter has investigated various computational design processes that can provide design solutions at the 'scheme' and 'detailed' design stage. It is the scheme design stage that is of most interest to architects because it is where there is more emphasis on inspiration and conceptual ideas are first realized.

Following chapter will investigate circulation diagrams that unfold the movement in the building environment.

Chapter 4: Circulation design

In this chapter we initially draw an analogy between circulation design and way-finding, on one hand and the navigation of design solutions in our design space on the other hand. We assume that the process of creating ‘good’ circulation design solutions is similar to the iterative process between ‘creativity’ and ‘accuracy’ in order to move and refine the region in design space.

4.1 Circulation

The Oxford English Dictionary [OED Online, January 2008] defines circulation as ‘The transmission or passage of anything (e.g. money, news) from hand to hand, or from person to person (with the notion of its ‘going the round’ of a country, etc.); dissemination or publication, whether by transmission from one to another, or by distribution or diffusion of separate copies’. In architecture and town planning circulation is the movement of people, motor vehicles, trains, aircraft, ships, bicycles, livestock and goods of all sorts.

The design of circulation systems involves many compromises and the same person may view a design as ‘good’ or ‘bad’ depending upon whether they are at that particular time in a car or on foot. Their view may also be influenced by whether they are in a hurry, whether they are alone, whether they undertake the journey regularly, the time of day, the weather, their age, their sex and their state of health. The issues that have to be addressed include:

i. Safety against accidents
ii. Safety against attack or molestation
iii. Provision for people with a disability
iv. Impact on the environment
v. Security of property
vi. Minimization of journey time
vii. Quality of the experience, including views, sense of arrival, awe, surprise etc.
viii. Ease of way-finding (not getting lost).Way-finding is discussed in section 4.4.6
ix. Space required
x. Protection from the elements (sun, wind, rain and snow)

xi. Privacy and social taboos

xii. Social interaction – meeting people

xiii. Cost

xiv. Maintenance

The relative importance of these criteria is subjective. We expect an architect to be able to take all these and other factors into account. On the other hand individual computational techniques can only examine a limited number of factors. One does not expect a computer program that analyses wind flow around a building (which is important for the comfort of pedestrians) to communicate directly with a program which is concerned with building costs. However both programs rely on a definition of different aspects of the geometry of the building.

In this dissertation the emphasis is on the *mechanical* design of pedestrian circulation, which is the topology and geometry of the circulation pathways and the flow of people along these pathways. Topology is the branch of mathematics concerned with the way in which objects are connected, thus a sphere is topologically equivalent to a cube, but a teacup is topologically equivalent to a torus (doughnut) since they both contain one hole (the handle of the teacup). An electrical wiring diagram shows the topology of how the components are connected by wires. Figure 4.1 shows the wiring diagram of a Citroën 2CV, a very simple car.

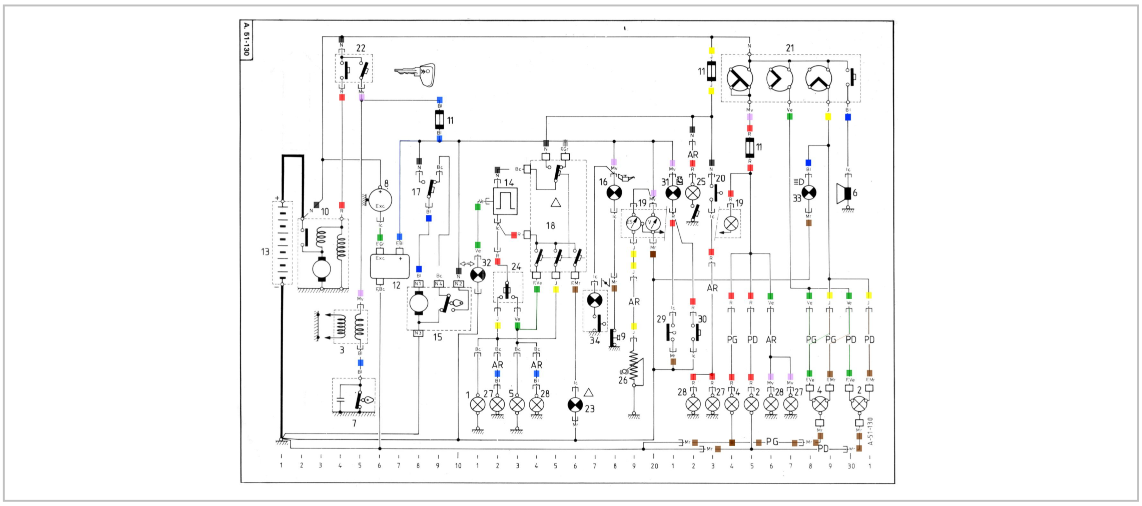


Figure 4. 1: Citroën 2CV wiring diagram [http://www.2cvstuff.com/Files/Wiring/Wiring%20Post%2081.pdf. January 2008]

The geometry of the wiring of a car would include information on the location of the components (battery, switches, bulbs etc.), plus the geometry of the wires as they snake around the body. The London tube map, figure 4.2, first designed by Harry Beck in

1933, is interesting because it is topologically exact, but geometrically distorted for clarity.

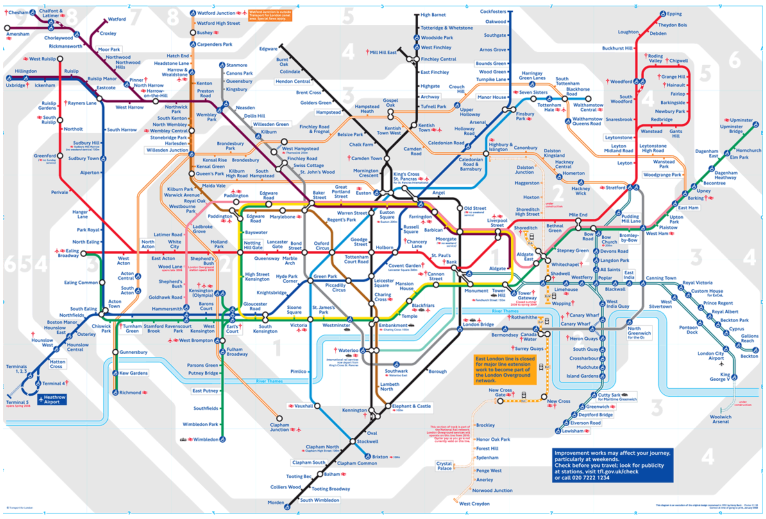


Figure 4. 2: London underground geometrical distorted [Available from numerous internet sites]

A geometrically accurate map such as that by James D. Forrester, figure 4.3, is much more difficult to read.

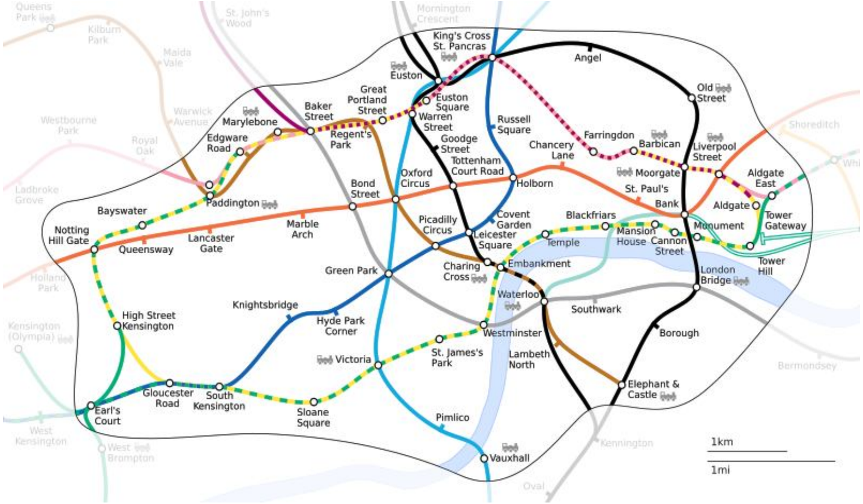


Figure 4. 3: London underground geometrically correct (Zone 1) [Available from numerous internet sites]

In a building certain types of area are designated as 'circulation areas', entrance lobbies, corridors, hallways etc. But this is a very limited definition in that movement of people takes place in all rooms and potentially all areas not occupied by furniture are circulation areas. Similarly entrance lobbies and corridors can contain seating and other furniture and are not used only for circulation.

4.2 The unconnected diagram of Design Framework

We assume that a computational technique applied to circulation design need to satisfy the basic organization of our Design Framework. A circulation strategy is defined, evaluated and refined (the inner loop) or it might be found to be unfeasible and so abandoned and some alternative strategy (new location in Design Space) investigated (the outer loop).

The Design Framework is a topological map, a circulation diagram that consists of various 'nodes' and 'links'. 'Nodes' correspond to different parts or actions of the design process and 'links' represent the possible connections between those actions.

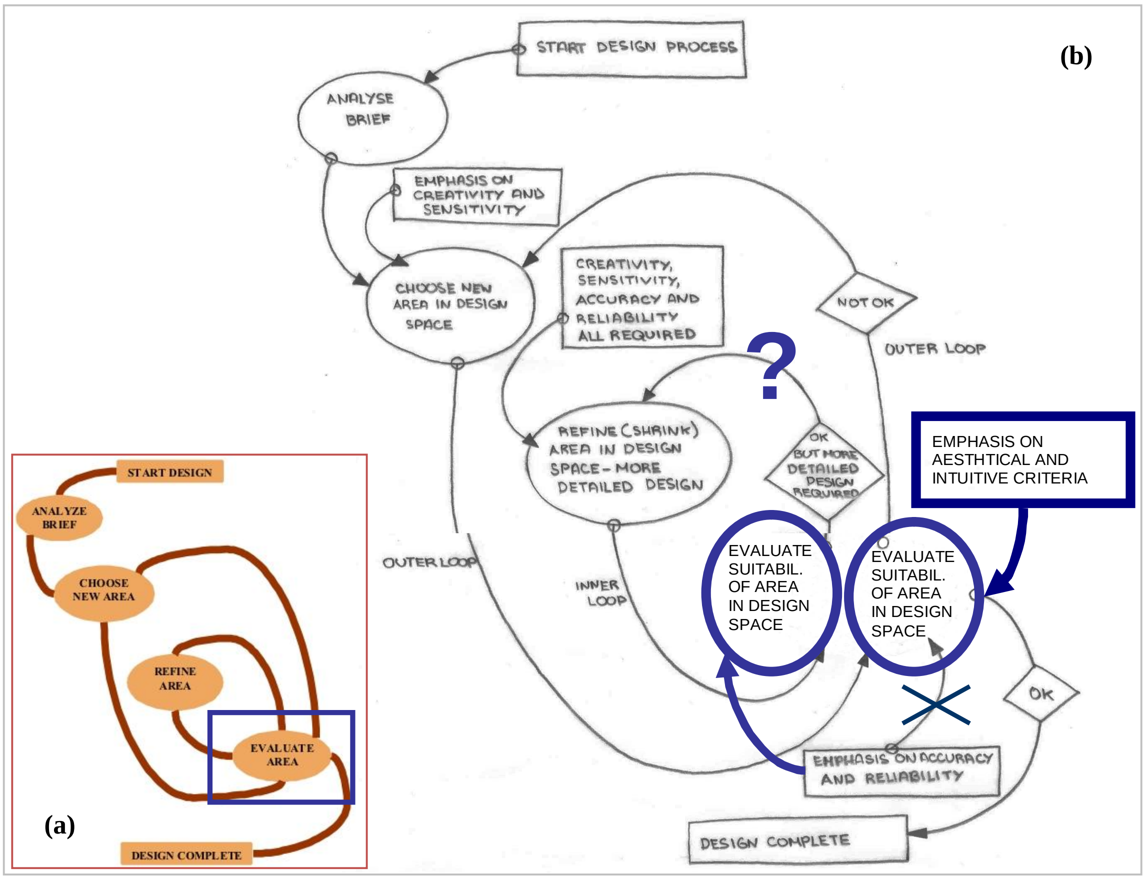


Figure 4. 4: a. Ideal connection in Design Framework, b. Unconnected paths of Design Framework

The network of connections helps the architects navigate the design space in order to find the 'best route' towards completion of the circulation design. The 'outer' and the 'inner' loops are separated mainly because of different evaluation processes (figure 4.4b).

In the 'outer' loop where the 'scheme design' is generated the evaluation is mostly based on a conceptual circulation diagram and the potential use of computer is limited.

In the ‘inner’ loop where the ‘detailed design’ is investigated, computers are used increasingly to analyse the pedestrian movement.

4.3 Creativity and circulation diagrams

In the ‘outer loop’, diagrams are used as the first attempt to navigate the design solution at a fundamental level by trying to solve the problem of the arbitrariness of form and the subjectivity of aesthetics.

The use of diagrams in art and architecture was central to the Bauhaus [Dewrstyne, 1968]; a school in Germany funded by Walter Gropius (1883-1969) and operated from 1919 to 1933. Artists like Vasily Kandinsky [1977] [Düchting, 2000, pp. 37-56] [Whitford,1967] and Paul Klee [Spiller, 1961] influenced the design teaching at the Bauhaus, introducing ideas of abstract diagrams in art giving emphasis to the spiritual, the spontaneous, and the dynamism of lines.

The diagrams introduced in Kandinsky’s [1979] *Point and Line to Plane* or Klee’s [1953] diagrams were examined and analysed by Deleuze [1993, p.14]. In *The Fold: Leibniz and the Baroque* he wrote:

> ‘Inflection is the ideal genetic element of the variable curve or fold. Inflection is the authentic atom, the elastic point. That is what Klee extracts as the genetic element of the active, spontaneous line. It testifies to his affinity for the Baroque and for the Leibniz, and opposes him to Kandinsky, a Cartesian, for whom angles are firm, for whom the point is firm, set in motion by an exterior force. For Klee, however, the point as a ‘noconceptual concept of noncontradiction’ moves along an inflection. It is the point of inflection itself, where the tangent crosses the curve. That is the point-fold’

Philosophical interpretation of diagrams can be found in Deleuze and Guattari’s work. They describe diagrams as accidental, free, random, abstract machines, maps, etc. In *Francis Bacon: The logic of sensation* Deleuze [1981, p. 71] wrote about the use of ‘graphs’ or ‘diagrams’ in Bacon’s work:

> ‘It is like the emergence of another world. For these marks, these traits, are irrational, involuntary, accidental, free, random. They are nonrepresentative, nonillustrative, nonnarative. They are no longer either significant or signifiers: they are asignifying traits. They are traits of sensation, but of confused sensation (the confused sensations, as Cèzanne said, that we bring with us at birth). And above all, they are manual traits’

The abstract representation of ideas in the form of diagram is explained by Deleuze and Guattari [1987, p. 156] as an 'abstract machine'. In *A thousand plateaus: Capitalism and Schizophrenia* they wrote:

> 'The abstract machine is pure Matter-Function – a diagram independent of the forms and substances, expressions and contents it will distribute […] We define the abstract machine as the aspect of moment at which nothing but functions and matters remain. A diagram has neither substance nor form, neither content nor expression'

A similar definition of diagrams is described in Deleuze's [1988, p. 35] work *Foucault*:

> 'On one occasion Foucault gives it its most precise name: it is a 'diagram', that is to say a 'functioning', abstracted from any obstacle […] or friction [and which] must be detached from any specific use'. The diagram is no longer an auditory or visual archive but a map, a cartography that is coextensive with the whole social field. It is an abstract machine. It is defined by its informal functions and matter and in terms of form makes no distinction between content and expression, a discursive formation and a non-discursive formation. It is a machine that is almost blind and mute, even though it makes others see and speak'

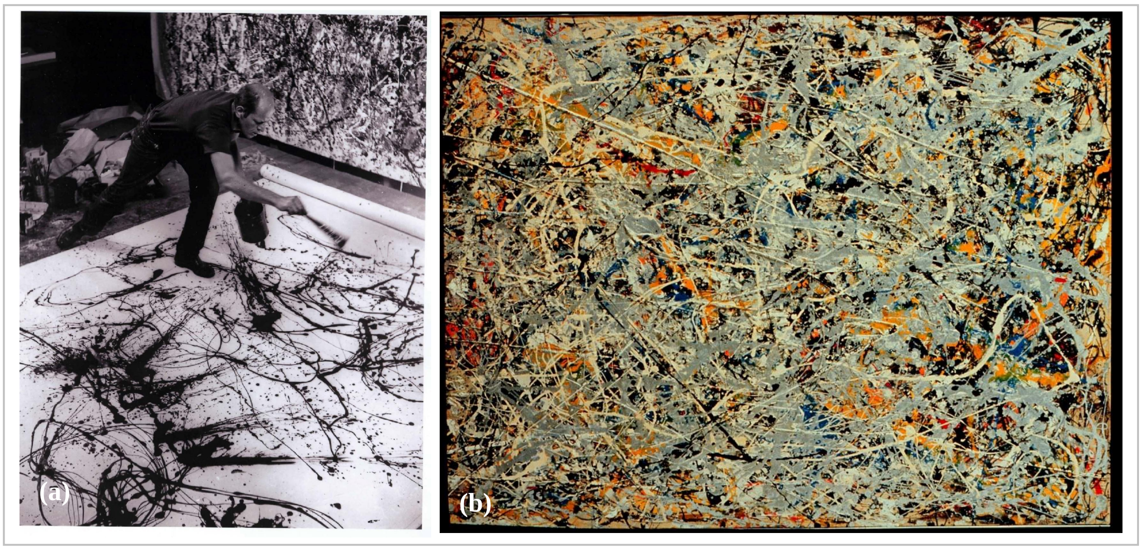


Figure 4. 5: a. Jackson Pollock, b. Jackson Pollock *Number 9* [Available from numerous internet sites]

Artistic representation of diagrams can be found in the work of Jackson Pollock (1912-1956) [Landau, 1989] [7] and Franz Kline (1910-1962). They were members of the abstract expressionist movement centre [1990] [8] in New York in the 1940s and 1950s. Their 'action painting' style has a resonance with the concept of circulation in the

---

[7] Reference to Kandinsky's [p. 153] and Klee's [p. 131] works that have some impact on Pollock's art.

[8] See also Motherwell, R, 1992. *The collected writing of Robert Motherwell.* New York; Oxford: Oxford University Press.

motion of the lines and covering of old lines with new. Jackson poured, threw and flicked paint onto a horizontal canvas.

Kline used a brush and he was said to produce many draft sketches before embarking on his ‘spontaneous’ work. He was influenced by traffic plans, bridges and buildings.

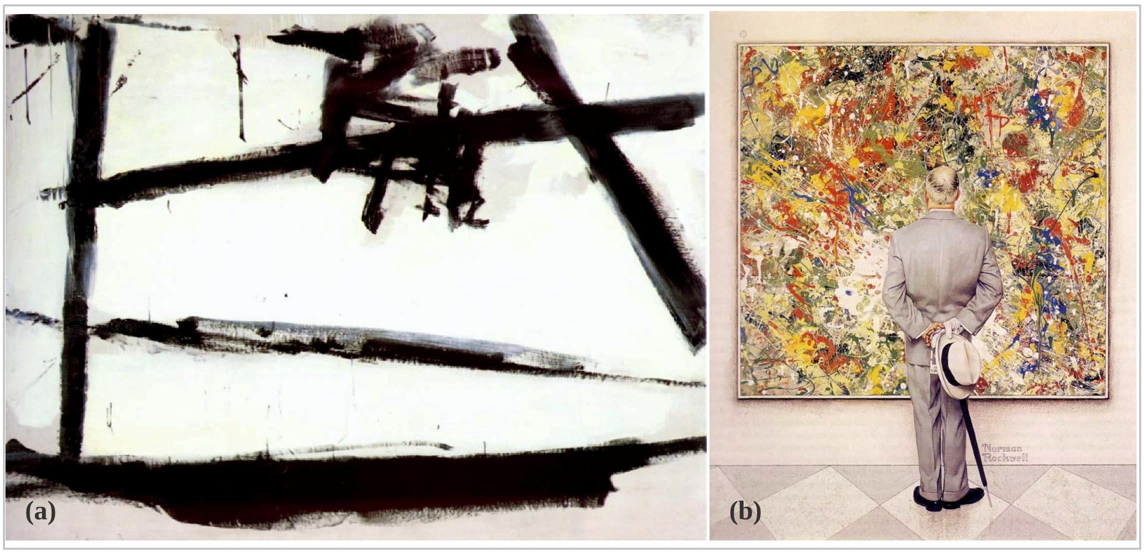


Figure 4. 6: a. Franz Kline *Number 2*, b. Norman Rockwell *The Connoisseur* [Available from numerous internet sites]

Abstract expressionism was not to everybody’s taste as illustrated in Norman Rockwell’s gentle painting, *The Connoisseur* (see figure 4.6d).

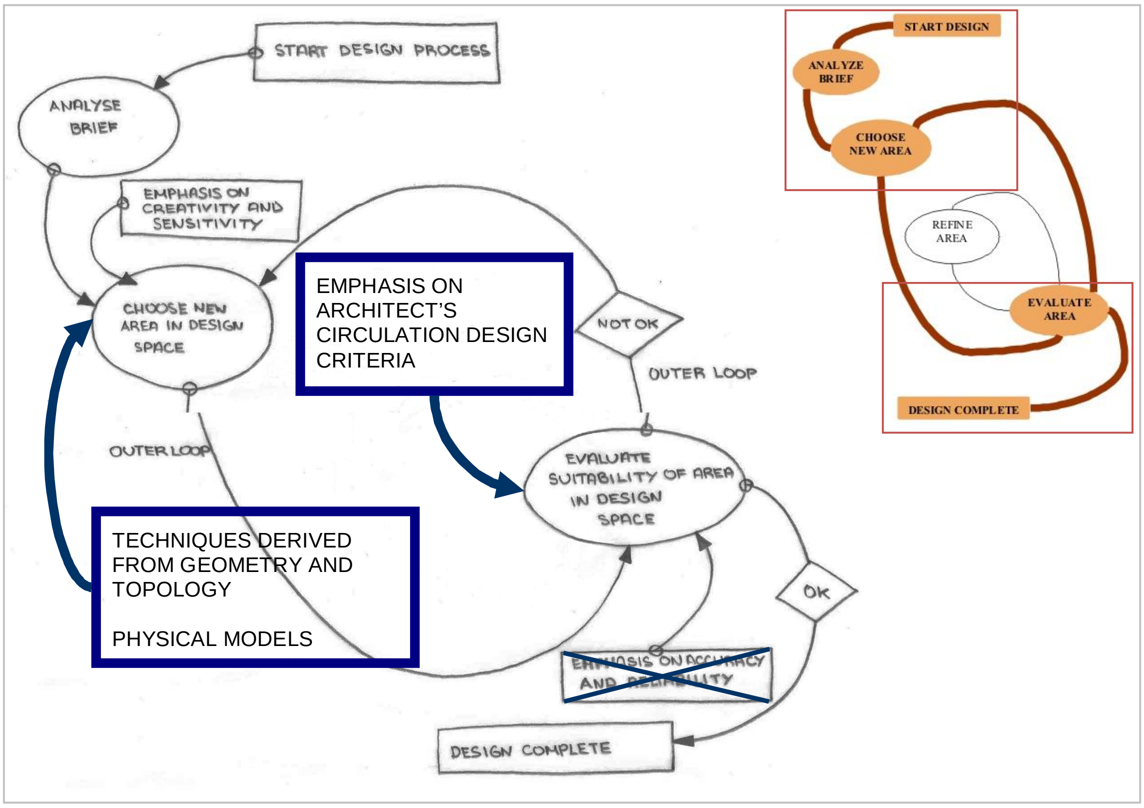


Figure 4. 7: Circulation design emphasize creativity

In architecture, abstract circulation diagrams are located in the ‘scheme design’ stage of our Design Framework (see figure 4.7) The emphasis is on creativity and sensitivity and solutions are evaluated using judgement and consideration of the solution fits in with their basic concept or diagram. The emphasis is not on accuracy and reliability.

Circulation diagrams can take different forms, shape and expressions. They can be only a movement scenario, a sketch drawing, or physical model. They can be applied in the ‘scheme design’ stage as vehicle of design thinking and they can be used to express ideas about the organizational relations between the architectural elements.

4.3.1 Le Corbusier and the experience of movement

Architects may not use actual circulation diagrams for their design investigation but their design may be influenced by various human movement scenarios or ideas of how people experience and move in urban areas and buildings.

Le Corbusier [1986, p.43] in *Towards a new architecture* wrote about the Acropolis in Athens and his movement experience:

> ‘It should not be forgotten that the site of the Acropolis is very up and down, with considerable variations in level which have been used to furnish imposing bases or plinths to the buildings. The whole thing being out of square provides richly varied vistas of a subtle kind; the different masses of the buildings, being asymmetrically arranged, create an intense rhythm. The whole composition is massive, elastic, living, terribly sharp and keen and dominating’

Circulation as human movement experience was introduced in his later work with the term ‘architectural promenade’. The design of Vila Savoye or Maison La Roche offers such movement qualities that make circulation inside the houses surprising, unexpected and intense.

Gaston Bachelard [1958, p. 11] in *The Poetics of Space* describes how we experience different spaces and how our perception of houses, paths, etc shapes our thoughts, memories and dreams. In section *The house from cellar to garret. The significance of the hut* he wrote about paths:

> ‘And what a dynamic, handsome object is a path! How precise the familiar hill paths remain for our muscular consciousness! A poet has expressed all this dynamism in one single line: *O, mes chemins et leur cadence*, Jean Caubere, Deserts (Oh, my road and their cadence) […] When I relive dynamically the road that ‘climbed’ the hill, I am

> quite sure that the road itself had muscles, or rather, counter-muscles […] Each one of us, then, should speak of his roads, his crossroads, his roadside benches'

Marcel Proust [1964, p.255] in *Swann's Way,* the 1st volume of his work *In Search of lost Time* describes his movement experience and the qualities he found by taking two different routes, the Meseglise and Guarmantes 'ways':

> 'No doubt, by virtue of having permanently and indissolubly combined in me groups of different impressions, for no reason save that they had made me feel several separate things at the same time, the Méséglise and Guarmantes 'ways' left me exposed, in later life, to much disillusionment, and even to many mistakes. For often I have wished to see a person again without realising that it was simply because that person recall to me a hedge of hawthorns in blossom; and I have been led to believe, and to make some one else believe in an aftermath of affection, by what was no more than an inclination to travel. But by the same qualities, and by their persistence in those of my impressions, to-day, to which they can find an attachment, the two 'ways' give to those impressions a foundation, depth, a dimension lacking from the rest'

This movement experience has great impact in architectural design since circulation is not only a matter of connecting two destinations with a minimum distance. The movement needs to reflect personal needs and desires that can only be found through the aesthetic aspect of circulation design.

4.3.2 Formal representation of circulation diagrams

Sketch diagrams are used by architects to describe the topological connections between functional units. However, the diagrams are not just topological because they express the initial organization of the design elements and the spatial (geometrical) relationships between the elements [Do and Gross, 2001].

Alexander [1964] used the terms 'form' and 'requirement' diagrams. He wrote:

The form diagram:

> 'summarise aspects of physical structure, by presenting one of the constituent patterns of its organization […] it remains principally a description of formal characteristics'

The requirement diagram is:

> 'intended to summarize a set of functional properties or constrains, like the arrow, or the population density map. This kind of diagram is principally a notation for the problem, rather than for the form'

A third form of diagram is the ‘constructive diagram’ [Alexander, 1964]. It can combine both, form and requirement diagrams. Figure 4.8 shows the example of a street map where thick and thin lines describe the number of cars and the actual size of the street at the same time [for further details, see Alexander, 1964].

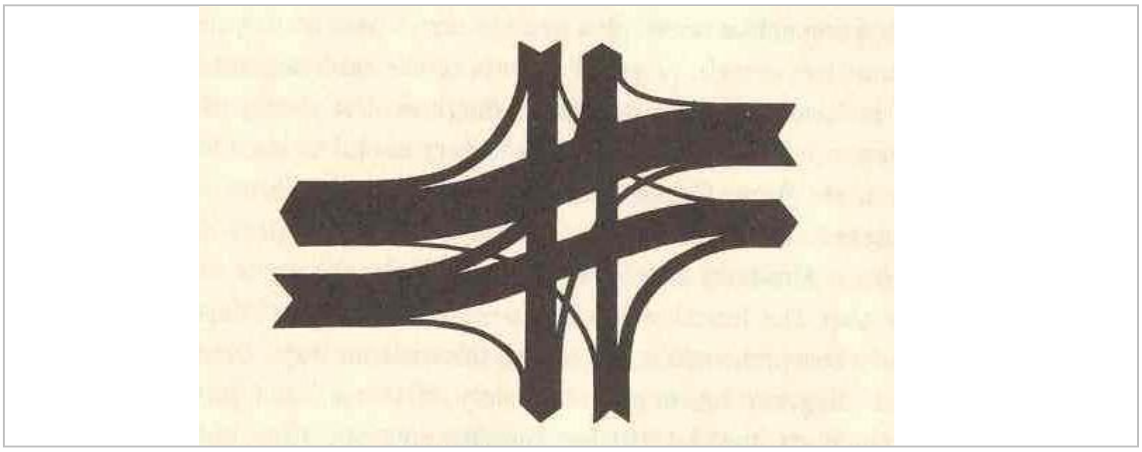


Figure 4. 8: Constructive diagram according to Christopher Alexander [1964]

The form diagrams may represent circulation patterns that are used as part of the ‘scheme design’ process. Their basic function is to link units and zones as well as to organize spaces [Arthur and Passini, 1992]. Also, they can be used to illustrate the system of human movements in an architectural environment.

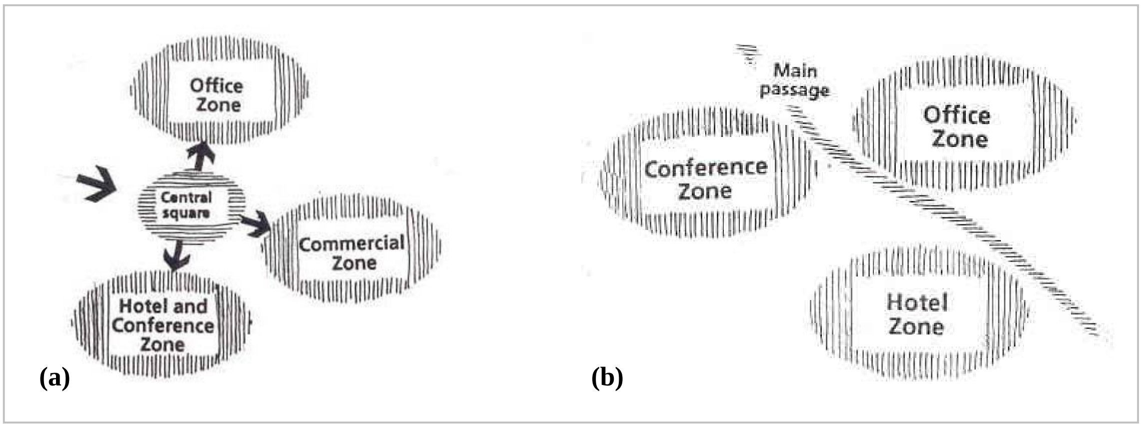


Figure 4. 9: Spatial planning on the basis of decision diagrams: first (a) and second (b) example [Arthur and Passini, 1992]

The relation between circulation diagrams and spatial organization is fundamental in layout design. Figure 4.9 shows two diagrams with different spatial and movement organization [Arthur and Passini, 1992]. In the first case the movement in different zones is achieved through the central zone, which distributes the movement in other zones as shown in figure 4.9a. Figure 4.9b shows organization of the zones across a main passage in linear form. Both form diagrams illustrate their importance in the ‘scheme design’ stage for the organization of the movement and the connection of functional areas.

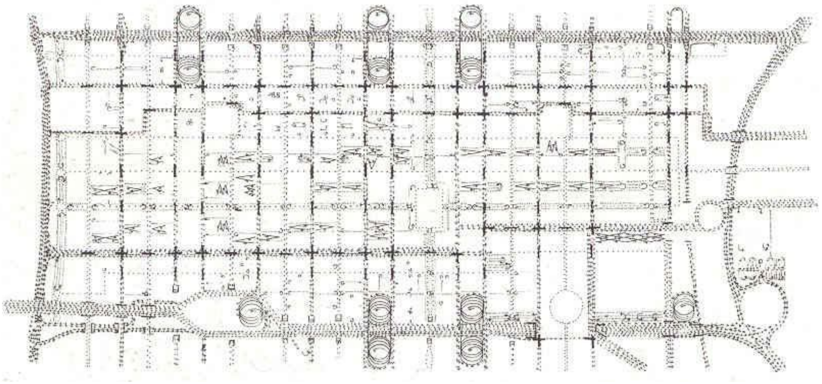


Figure 4. 10: Proposed movement pattern [Latour, 1991]

Figure 4.10 shows Louis Kahn’s design diagram as proposal for midtown Philadelphia. Symbols were used as a vocabulary to express new movement system. Different architectural entities were translated into symbols: arrow (->) indicates a garage, plus symbol (+) intersection, greater than symbol (>) parking, and so on. The pattern can be understood as movement system that can be read using a specific vocabulary of symbols [Latour, 1991].

Ben van Berkel and Caroline Boss [Feireiss, 1993] use form, requirement, or constructive diagrams to illustrate systems of human movement at a conceptual or at a realistic level. Their diagrams take ideas from mathematical concepts like the Möbius strip (discovered by the mathematician August Ferdinand Möbius), a surface with only one side and only one boundary component. The Möbius House is based on this conceptual diagram movement [9].

4.3.3 Circulation diagrams and physical models

Physical models can also be used to describe circulation diagrams. Pioneering work in this area was done by the German architect Frei Otto. Figure 4.11 shows three basic steps of generating ‘optimal path system’ using physical models [Otto and Rasch, 1995]. The system’s principle task was to connect targets (in this case houses) that were arranged in a circle. Using wool threads, the targets were connected by stretching the wool threads between the starting and the terminus points. Then, and after the threads have been loosen, the physical model was dipped into the water resulting the threads to

---

[9] The Mobius loop, the spatial quality of which means that it is present in both plan and section, translates into the interior into 24-hour cycle of sleeping, working and living. As the loop turns inside out the materialization follows these change-overs [Feireiss, 1993, p. 227].

stick together, starting to merge and creating an ‘optimal path system’ [Otto and Rasch, 1995] [Spuybroek, 2004].

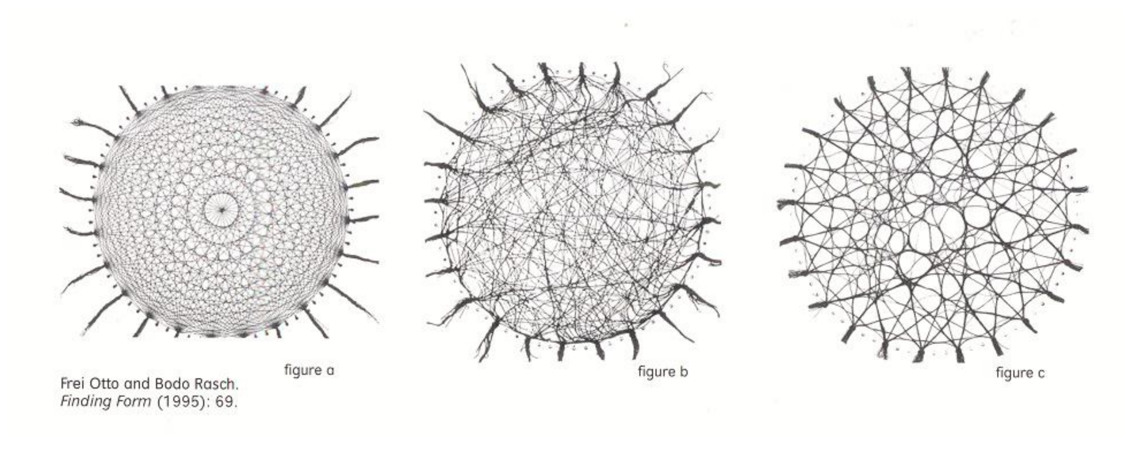


Figure 4. 11: Three steps of Otto’s procedure starting from ‘direct path system’ and ending by ‘optimal path system’ [Spuybroek, 2004]

The same approach has been applied in the research project named ‘Paris brain’ [Spuybroek, 2004] where Otto’s wool-thread techniques have been used for the form-finding, figure 4.12 [for further details, see Spuybroek, 2004].

Figure 4. 12: Machine procedure for ‘Paris brain’ [Spuybroek, 2004]

Manuel DeLanda [2004] in *Materiality: Anexact and Intense* analysed the work of Frei Otto and Lars Spuybroek describing their diagrams as systems of relations that are based on the relative than the absolute. DeLanda made connections between the diagrams and Deleuze’s ideas of ‘abstract machine’, ‘anexact yet rigorous’. The concept ‘progressive differentiation’, according to DeLanda, is a process in which an originally undifferentiated object progressively acquires more and more details. DeLanda wrote:

> ‘Since spontaneous progressive differentiation is a natural way to explore the abstract space of possible forms, it can also be used by architects and engineers to generate detailed design starting from a more or less bland initial state’

4.3.4 Christopher Alexander and computer-generated diagrams

Christopher Alexander's [1964] work in *Notes in the synthesis of Form* shows that the arbitrariness of form can be captured in the preliminary stage of design using mathematical and computational processes. This is possibly the first attempt at using computer-generated diagrams to connect different functional requirements and hence analyse the brief requirements of a design problem.

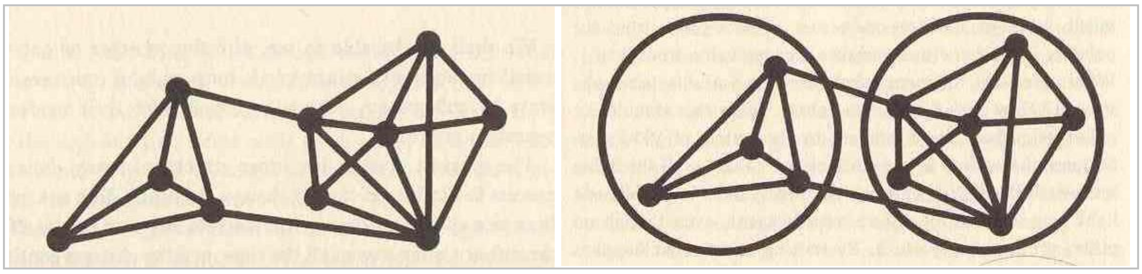


Figure 4. 13: Alexander's diagrams of functional connections [Alexander, 1964]

Alexander [1964] used an example from an Indian village to list different needs, assertions, rules, activities, etc., as discussed in section 3.2, *Analysis of the brief*. Algorithms based on rules of connectivity were used to connect different requirements forming groups of sub-systems. Figure 4.13 shows two diagrams where nodes represent requirements and lines represent the linkage between them. Clearly nodes that are grouped together have closer relations than nodes in a longer distance. Alexander [1964, p. 43] wrote about his process:

> 'We may therefore picture the process of form-making as the action of a series of subsystems, all interlinked, yet sufficiently free of one another to adjust independently in a feasible amount of time. It works because the cycles of correction and recorrection, which occur during adaptation, are restricted to one subsystem at a time'

According to Alexander, the generation of diagrams can be seen as an abstract representation of design problem at a fundamental level. He wrote:

> 'Because the diagrams are independent of one another, you can study them and improve them one at a time, so that their evolution can be gradual and cumulative […] because they are abstract and independent, you can use them to create not just one design, but an infinite variety of designs, all of them free combinations of the same set of patterns'

4.3.5 Ballantyne on Deleuze and Guattari

We have made frequent reference to Ballantyne's work *Deleuze and Guattari for Architects.* Concerning the connections between emergence, collective intelligence, artificial thinking, and networks, he wrote [2007, p. 31]:

> 'The sciences discuss immanence when they analyse 'emergence'. Emergent properties develop in a system when it is allowed to run, and they are interesting when they are unexpected properties that do not seem to have been programmed into the start conditions. Immanence and emergence are therefore different aspects of the same process, relating a set of generative properties to a set of products'

According to Ballantyne [2007, p. 31], the example of a swarm of slime mould shows that simple organisms without discernible brain, when they are acting as a group are able to fulfil task that need intelligence like to find the shortest route through a maze:

> 'Acting as a group, somehow it becomes possible for them to do it, and not because of any supernatural or psychic guidance that the slime mould […]'

Ballantyne makes connections with Johnson's [2001] work *Emergence* in which collective intelligence is about to 'think'. Ballantyne makes further connections with 'networks of interconnectivity' and the philosophy of Deleuze and Guattari [Ballantyne, 2007, p. 32]:

> 'The idea has moved from seeming very arcane, to being a very evident part of everyday life, as a certain group of our connections in the world become much more evident to us, and we construct our identities in part through those connections […] What distinguished Deleuze and Guattari'd account is the fact that it connects these large-scale networks with the networks of the body, and between bodies, so that we start to see things like temperament and identity as emergent properties that are products of the machine immanent in the initial condition'

4.4 Circulation design evaluation

When moving about a city or building people have to share circulation routes with other people. This leads to social interaction and communication. It may also lead to congestion if there are too many people. There is also the issue of way-finding, that is knowing which path to follow.

The evaluation of the performance of a proposed design in these respects is in the 'detailed design' part of the Design Framework (see figure 4.14).

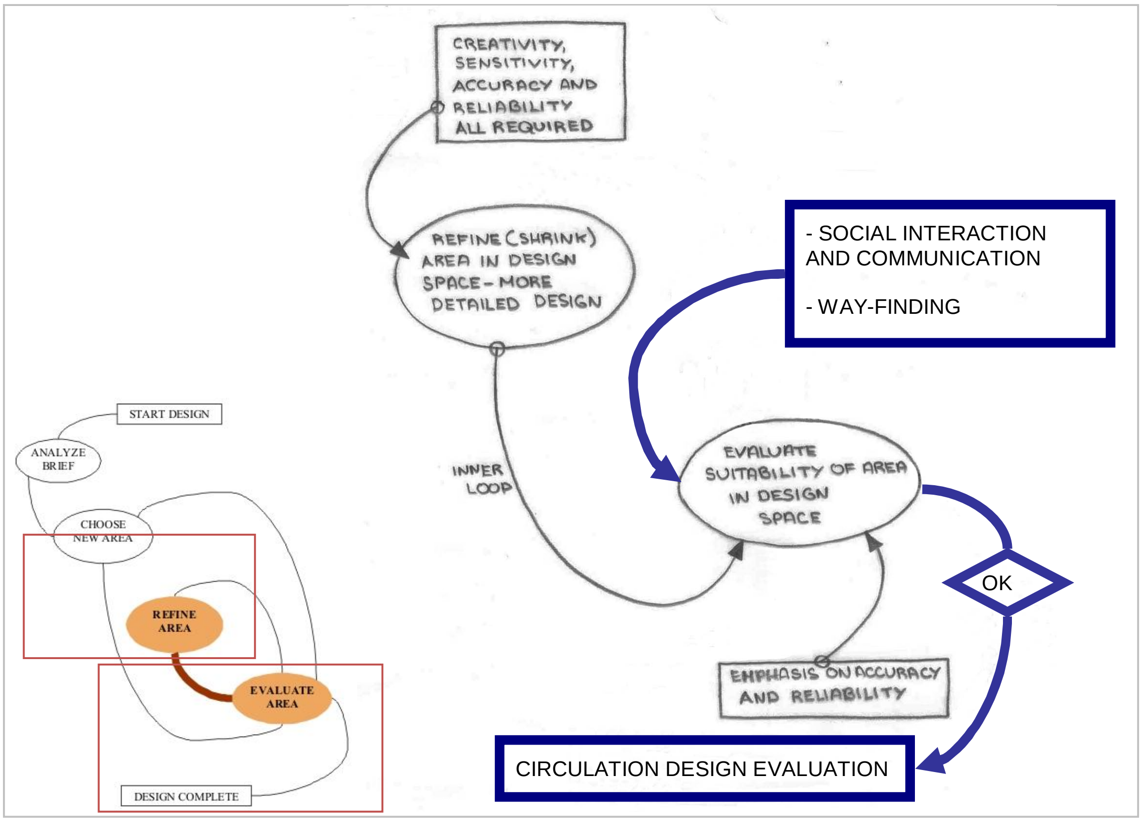


Figure 4. 14: Evaluation of circulation design emphasizing accuracy and reliability

4.4.1 Computer modelling of social interaction and communication

Social interaction and communication issues include:

- Pedestrian traffic and crowd control
- Congestion and evacuation
- Social and economic interactions

A number of related and sometimes interchangeable words and phrases are used to describe computer techniques for the modelling of pedestrian movement. They usually treat people as Cellular Automata, but it is possible to give people individual properties according to age, infirmity etc. These automata have desires to reach various different destinations are subject to social pressures and possibly physical forces from walls and other people. Emergent phenomena may occur, which if of benefit might be called Collective Inteligence. Hoogendooor et al., [2001] discuss the 'time-space behaviour' of individual pedestrians and Teknomo et al., [2000] discuss 'microscopic pedestrian modelling'.

Turner and Penn, [2002] describe agent (Cellular Automata) approaches as mesoscopic modelling. They use the phrase macroscopic modelling when a large number of people are treated as a continuum or continuous medium subject to differential flow equations similar to those of fluid dynamics [Turner and Penn, 2002]. Henderson [1974, cited by Teknomo et al., 2000] also examined pedestrian crowds using a fluid-dynamic model and Helbing et al. [1992] suggested a model based on Boltzmann-like gas kinetic approach. Real gases are composed of molecules or 'agents', however Avogadro's number (6.0 x 10^{26} molecules/kmol or 2.7 × 10^{25} molecules/m^3 at 0ºC and 1 atmosphere (Loschmidt's number)) precludes the use of considering every molecule in a computer simulation.

4.4.2 Social Forces

The Social Force technique is based on the idea of 'field theory'. The theory was introduced by Kurt Zadek Lewin (1890-1947), a German psychologist and one of the pioneers in social organizational and applied psychology. Lewin's [1951, p. 38] 'field theory' was one of the most influential developments in the social sciences, stating that the analysis of human behaviour should be based on a 'force field'. He wrote:

> 'There is an indication that certain types of questions can be answered by certain types of constructs. For instance, it seems that a 'prediction of the behaviour of an individual in a specific situation' has to be based on a 'force field' or a conceptually equivalent construct. If it is correct that no other conceptual type (such as power field, position, tension, force) suffices for such a prediction, important positive and negative methodological implications are obvious'

Such 'conceptual types' or constructs include terms like *Position, Locomotion, Cognitive structure, Force, Goal, Conflict, Fear, Power, Values*. One characteristic example is *Position*. It is described as 'spatial relation of region; for instance, the position of a region A can be characterized by its lying in B'. Also, *Force* is described as 'tendency to locomotion' and *Goal* as 'a force field of a special structure, namely a force field where all forces point towards the same region'.

These 'conceptual types' might influence the behaviour of an individual in a certain time: Lewin [1951, p. 19] wrote:

> 'According to the general field theory the actual behaviour is related to the resulting force acting on the person at that time. It is therefore always important to know which other forces might influence behaviour aside from those specifically established in the experiments'

This may seem a blinding glimpse of the obvious, but this might possibly not have been that case in 1951.

The application of computers to the Social Force Model originated in the work of Dirk Helbing and Péter Molnár at the Institute of Theoretical Physics at the University of Stuttgart. Helbing and Molnár [1994; 1995] gave a mathematical description of the 'field theory' by describing positions of pedestrians in space that can change continuously over time. The pedestrians are subjected by various forces, which represent different influences from other pedestrians or the environment. These influences may include a tendency to keep a distance from other pedestrians, obstacles, etc. and are described by repulsive forces. The tendency to be attracted by objects, etc. is described by attractive forces. The summation of all forces acting upon single pedestrian motivates its movement behaviour.

Following paragraphs describe briefly the formulation of the Social Force model as explained by Helbing et al. [2001] in his work *Self-organized pedestrian movement.* Different notations can be found in other papers by Helbing et al. The concepts will be explained fully in chapter 5 where the author will use a somewhat different notation (for example using a bold, non-italic font for vectors, rather than Helbing's bold, italic):

The position of pedestrian α is represented by a point specified by the position vector $\boldsymbol{r}_\alpha(t)$ and the position changes over time, t , thus speed is $\boldsymbol{\upsilon}_\alpha(t)$ and the equation of motion is:

$$\frac{\mathrm{d}\boldsymbol{r}_\alpha(t)}{\mathrm{d}t} = \boldsymbol{\upsilon}_\alpha(t)$$

The social force $\boldsymbol{f}_\alpha(t)$ represents the sum all the forces influencing pedestrian α and $\xi_\alpha(t)$ represents random behavioural variations, thus the acceleration or deceleration of pedestrian α can be written as:

$$\frac{\mathrm{d}\boldsymbol{\upsilon}_\alpha}{\mathrm{d}t} = \boldsymbol{f}_\alpha(t) + \xi_\alpha(t)$$

In Helbing's model the overall social force $\boldsymbol{f}_\alpha(t)$ is described mathematically as:

$$\boldsymbol{f}_\alpha(t) = \boldsymbol{f}_\alpha^0(\boldsymbol{\upsilon}_\alpha) + \boldsymbol{f}_{\alpha B}(\boldsymbol{r}_\alpha) + \sum_{\beta(\neq\alpha)} \boldsymbol{f}_{\alpha\beta}(\boldsymbol{r}_\alpha, \boldsymbol{\upsilon}_\alpha, \boldsymbol{r}_\beta, \boldsymbol{\upsilon}_\beta) + \sum_i \boldsymbol{f}_{\alpha i}(\boldsymbol{r}_\alpha, \boldsymbol{r}_i, t)$$

Where $\boldsymbol{f}_\alpha^0(\boldsymbol{\upsilon}_\alpha)$ is the acceleration force trying to keep the pedestrian α's speed constant, $\boldsymbol{f}_{\alpha B}(\boldsymbol{r}_\alpha)$ is the repulsive force from the boundary, B, $\boldsymbol{f}_{\alpha\beta}(\boldsymbol{r}_\alpha, \boldsymbol{\upsilon}_\alpha, \boldsymbol{r}_\beta, \boldsymbol{\upsilon}_\beta)$ are the repulsive interactions with other pedestrians β, and $\sum_i \boldsymbol{f}_{\alpha i}(\boldsymbol{r}_\alpha, \boldsymbol{r}_i, t)$ are the attraction effects drawing α towards the goal i.

The Social Force Model has been used extensively in the area of pedestrian dynamics for the investigation of self-organized phenomena including counter flow (pedestrians walking in different directions and creating lanes of uniform walking direction), bottlenecks, intersecting flows, lane formation in corridors, fire evacuation (agents trying to find their way towards nearest exists), and so on.

According to Helbing et al. [2005] any model can be evaluated depending upon its ability to describe self-organized phenomena and to provide accurate results. The Social Force Model has been widely used and has proved to be a realistic model for the modelling of human behaviour especially in cases of panic and fire escape situations where the evacuation of users needs to be investigated.

When pedestrians are walking in opposite directions and are passing through a narrow door at the same time. In such a case, oscillatory changes in the walking direction are observed that may influence the effective flow of pedestrians. Helbing et al. [2005] argued that:

> ‘a skilful flow optimization utilizing the self-organization phenomena emerging in pedestrian streams can increase the efficiency and safety of pedestrian facilities at comparable cost and even with less available space’

Different studies have examined the application of Social Force Models to improve pedestrian facilities in counter flows, bottlenecks, clogging in pushing crowds, intersections, egress routes of theatres, classrooms, lecture halls, queues at entrances, optimal way systems, and so on.

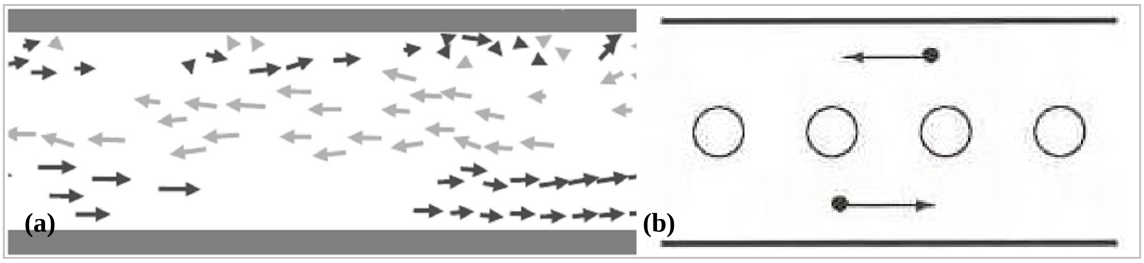


Figure 4. 15: a. In crowds of oppositely moving pedestrians, one can observe the formation of varying lanes consisting of pedestrians with the same desired direction of motion [Helbing et al., 2001, p. 369], b. Improved elements of pedestrian facilities: ways [Helbing et al., 2001, p. 373]

Figure 4.15a shows an example of a conventional corridor with a uniform walking direction and high pedestrian density. A simple solution suggests a series of obstacles in the middle of corridor that can guide lanes along specific routes and stop the obstruction (see figure 4.15b) [for further details, see Helbing et al., 2001].

Another example suggests improvements in corridor design of stadium exits. In this case, an emergency situation may cause blockage of corridors due to the frightened pedestrians pushing from behind. The situation can be improved by increasing the dimensions of the corridor, which will result the reduction of the waiting time. Also, a zig-zag shape of corridors may reduce the pedestrian pressure as shown in figure 4.16b [for details, see Helbing et al., 2005].

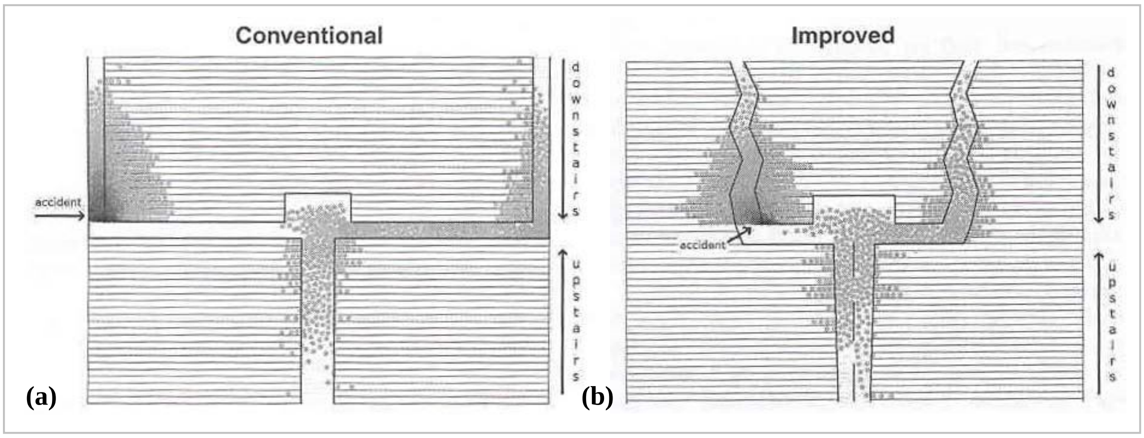


Figure 4. 16: Conventional (a) and improved (b) design of a stadium exit [Helbing et al., 2005, p. 18]

Okazaki and Matsushita [1993] used a concept similar to ‘social forces’ which they called the Magnetic Force Model. According to this approach, the equation of motion in a magnetic field is used to model pedestrian movement. The individual pedestrians move towards the destinations subjected by magnetic forces. Figure 4.17 shows the forces acting upon the pedestrian in the Magnetic Force Model.

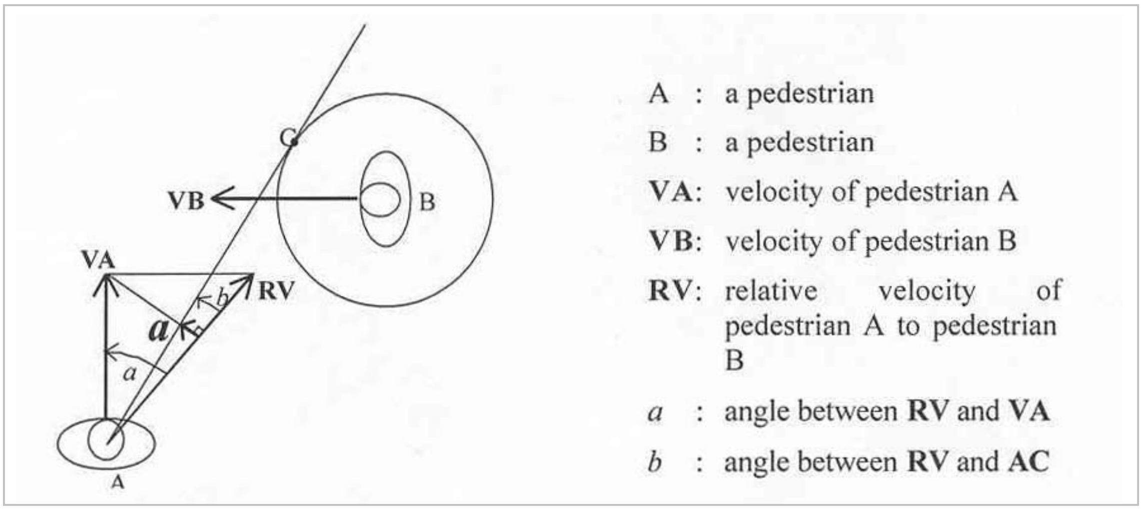


Figure 4. 17: Acceleration a acts on pedestrian A to avoid a collision with pedestrian B [Okazaki and Matsushita, 1993]

The Magnetic Force Model has been used to evaluate human movement behaviour in existing designs, to investigate evacuation phenomena in buildings, to examine the movement behaviour in queue space and so on.

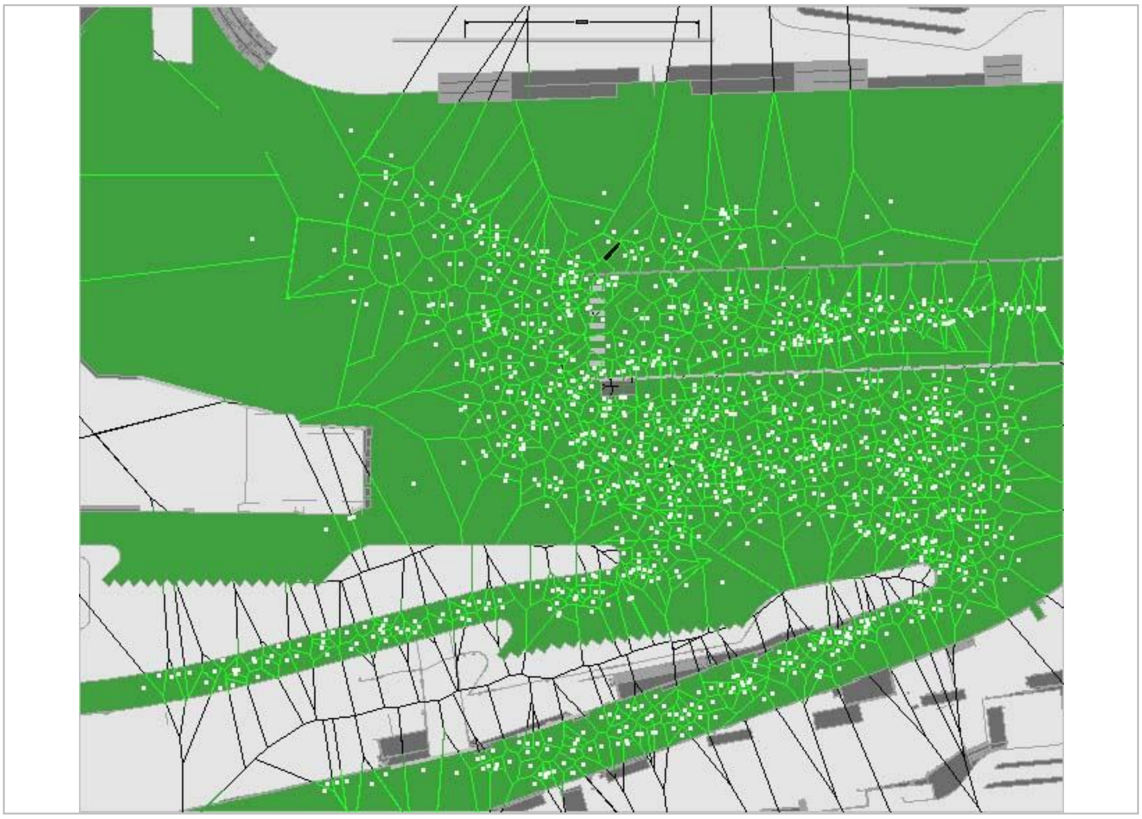


Figure 4. 18: Voronoi Analysis applied to crowd density analysis [http://www.crowddynamics.com/Voronoi/voronoi.htm sampled July 2008]

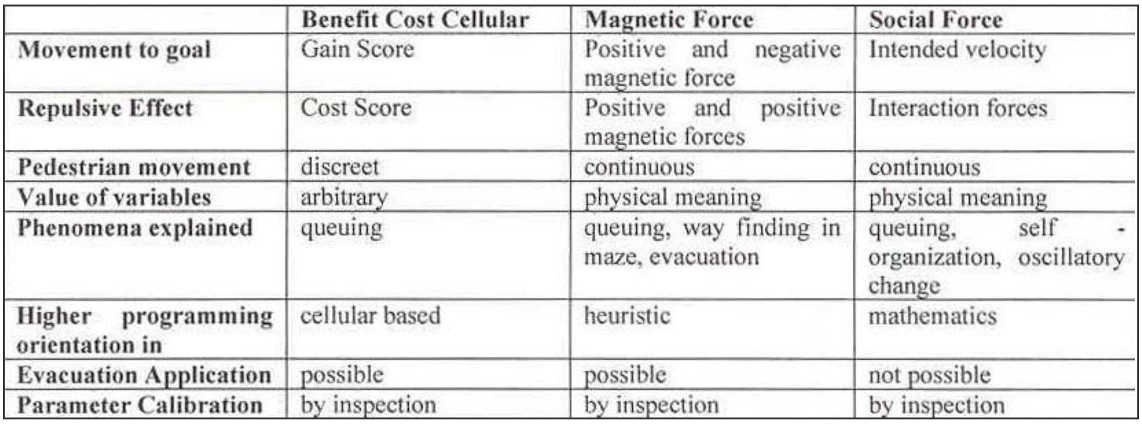
The social force model is related to the concept of ‘personal space’. Figure 4.18 is from the Crowd Dynamics Ltd. website [sampled July 2008]. It shows an interesting application of the Voronoi diagram in which each polygon defines the space containing points nearer to each individual than to any other.

4.4.3 Cellular Automata Models

Cellular Automata approaches are again based the assumption that different kinds of human behaviour are influenced by rational rules. According to this approach human-based activity may be regarded as cost-benefit behaviour [Hoogendoorn et al., 2001].

A Benefit Cost Cellular Model has been described by Teknomo et al. [2000]. In such models, pedestrians are simulated as particles in cells and their movement behaviour is influenced by maximum net benefit factors.

Table 4.1 shows a comparison by Teknomo et al. [2000] of the basic concepts of the Social Force, the Magnetic Force and the Cellular Automata.



	Benefit Cost Cellular	**Magnetic Force**	**Social Force**
Movement to goal	Gain Score	Positive and negative magnetic force	Intended velocity
Repulsive Effect	Cost Score	Positive and positive magnetic forces	Interaction forces
Pedestrian movement	discreet	continuous	continuous
Value of variables	arbitrary	physical meaning	physical meaning
Phenomena explained	queuing	queuing, way finding in maze, evacuation	queuing, self - organization, oscillatory change
Higher programming orientation in	cellular based	heuristic	mathematics
Evacuation Application	possible	possible	not possible
Parameter Calibration	by inspection	by inspection	by inspection

Table 4. 1: Comparison Microscopic Pedestrian Simulation Models [Teknomo et al., 2000]

All three techniques step through time, the main difference is that Cellular Automata Models are discrete in space whereas the other two approaches are continuous.

Teknomo et al. [2000] argued that none of these models have calibrated the parameters based on real pedestrian movement data, though later work done by Helbing et al. [2005] shows that the Social Force Model can be highly correlated with empirical observations. Work by Galea et al. [1996] showed these techniques can be used to predict behaviour during evacuation in the case of a fire.

4.4.4 Collective Intelligence Models

Social interaction and communication behaviour may be found in the simulation of collective animal behaviour (artificial life) using again ‘autonomous’ agents [Reynolds, 1987] [Terzopoulos, 1999].

Although Collective Intelligence and Social Force Models are found to be different in terms of the final simulation of tasks, similarities can be found since:

1. Models are based on similar ‘rules of navigation’ such as tendency between the agents to avoid each other applying rules of repulsion.
2. The process of interaction between agents in order to create global order is called self-organization
3. The internal organization of the system, an open system, increases automatically without being guided or managed by an outside source.
4. The self-organized system display emergent properties.

Craig W. Reynolds [1987] in *Flocks, herds and schools: A distributed behavioural model* introduced an artificial life program to simulate the flocking behaviour of birds. Each bird (boid) was treated as an individual, applying simple behavioural rules of interaction, and the result was an aggregate motion of flocks. The simulated flock was closely related to the behaviour of particle systems used for the simulation of a number of physical phenomena, for instance clouds, oceans waves, and so on [Reynolds, 1987].

The birds moved under the opposing forces of collision avoidance and the urge to join the flock. Briefly, three basic rules were applied [Reynolds, 1987]:

I. Collision avoidance: avoid collision with nearby flock mates
II. Velocity Matching: attempt to match velocity with nearby flock mates
III. Flock Centring: attempt to stay close to nearby flock mates.

The rules are illustrated graphically in figure 4.19 (a, b, c) together with the *attempt to maintain clear view* behavioural rule (d) [Flake, 1998].

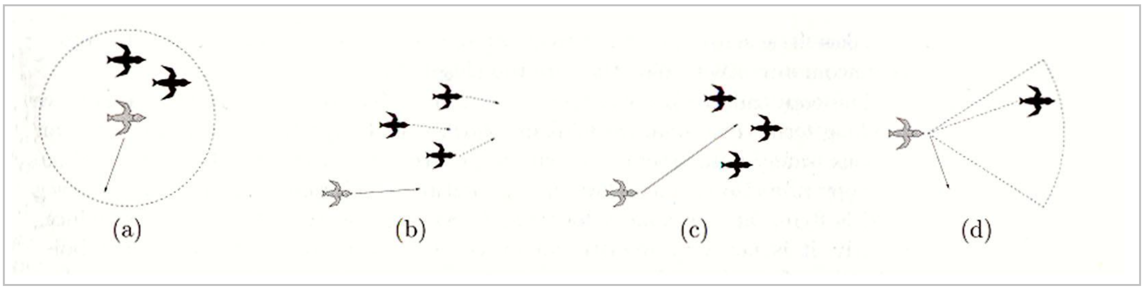


Figure 4. 19: Four boid rules: (a) avoid flying too close to others; (b) copy near neighbors; (c) move towards center of perceived neighbors; (d) attempt to maintain clear view [Flake, 1998]

According to Reynolds [1987] the *collision avoidance* and the *velocity matching* behavioural rules are complementary. Combination of the rules allows the birds to fly within crowded flock's interior without running into one another. The *flock centring* rule allows a bird to move near to the centre of flock. An example of birds in motion is shown in figure 4.20. It can be observed that from a disorganized initial position the birds gradually get together into a single flock.

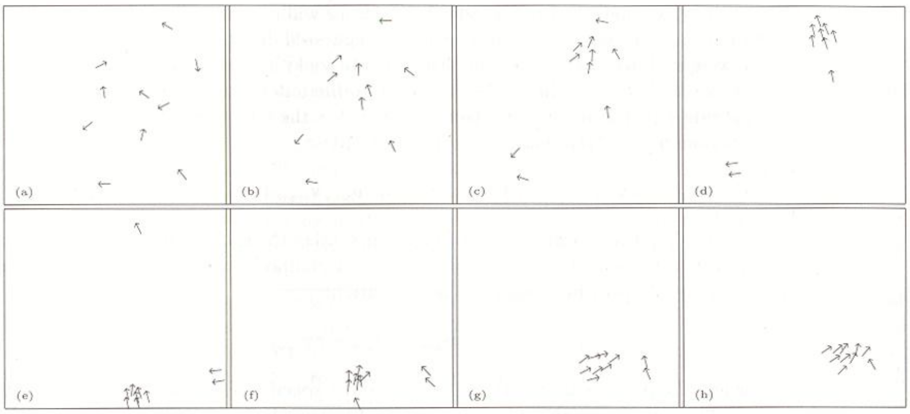


Figure 4. 20: A collection of boids self-organize to form a flock [Flake, 1998]

Resnick [1994] [Flake, 1998] developed a simulation system that was based on theoretical termites. The agents (termites) followed a number of simple rules, without any intelligence, but nevertheless produced interesting results. Each agent was given the following set of rules:

i. Wander around aimlessly, via a random walk, until the termite bumps into a wood chip

ii. If the termite is carrying a wood chip, it drops the chips and continues to wander

iii. If the termite is not carrying a wood chip, it picks up the one that it bumped into and continues wander

Figure 4.21 shows the results of this procedure.

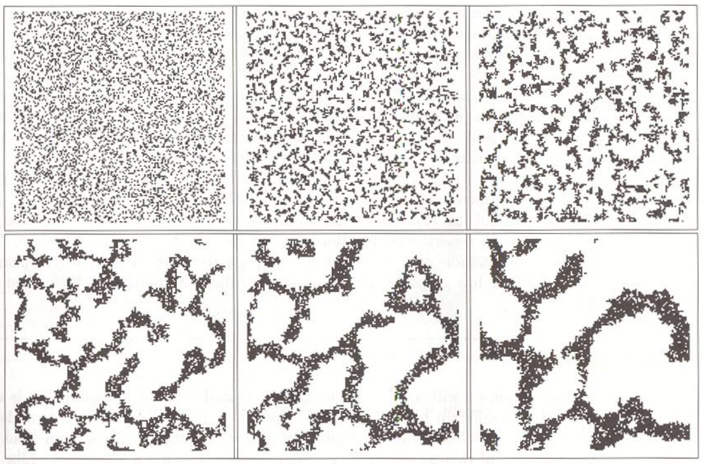


Figure 4. 21: Termites randomly placing wood chips according to simple rules produce order [Flake, 1998]

In all cases, the agents were acting independently without any command from a leader or without having a general plan of the goal that should be achieved. However, they were able to produce a global organization. An important aspect in this process was the nature of the rules that were applied. According to Flake [1998] small changes in the rules can produce very different results.

4.4.5 Summary of social interaction and communication models

The various approaches for modelling the behaviour of ‘autonomous’ agents in general and pedestrian behaviour in particular have shown the application of simple rules of behaviour generating complex outcomes. Similar approaches can be found in the area of fluid-dynamics and especially in the example of smooth particle hydrodynamics, which was discussed in section 3.7.2. Figure 4.22 is an attempt to locate these techniques in the Design Framework. Here they are shown in the ‘evaluation section’, although they might also occur as generative tools, searching out new locations in design space.

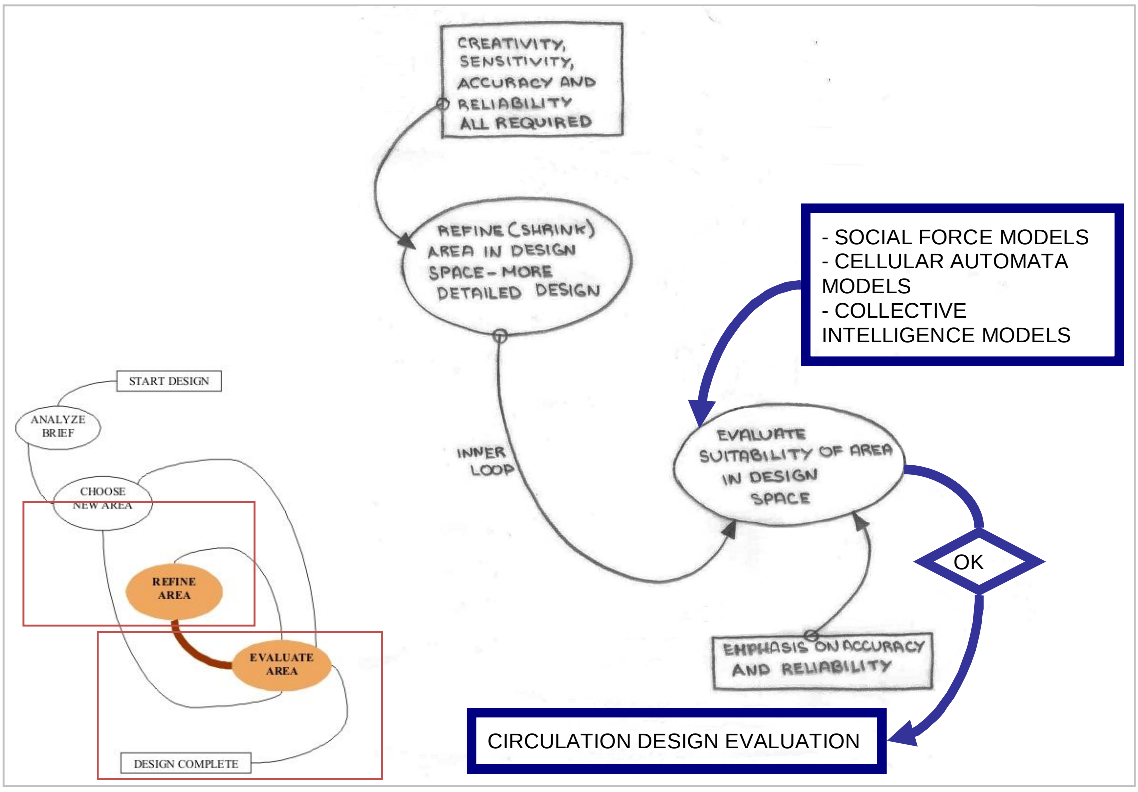


Figure 4. 22: Evaluate circulation design based on individual local interaction rules

4.4.6 Way-finding

Way-finding describes the process by which people establish which way to walk in order to get to their desired goal. The compound word ‘way-finding’ was introduced by Kevin Lynch [1960, p. 4] in his book *The Image of the City*, where he stated that:

> ‘In the process of way-finding, the strategic link is the environmental image, the generalized mental picture of the exterior environment that is held by every individual’

The hyphen is now often omitted and the word ‘wayfinding’ used.

Lynch was writing about a city, but clearly the same concepts apply to large building complexes. He listed 10 clues to be used in way-finding:

1. Singularity
2. Form simplicity
3. Continuity
4. Dominance

5. Clarity of joint
6. Directional Differentiation
7. Virtual scope
8. Motion awareness
9. Time series
10. Names and meaning

The University of East London Map Problem

The problem of orientation in wayfinding can be illustrated by the example of the University of East London campus:

Figure 4.23 shows the Campus Map from the University of East London (UEL) website. The Campus is served by Cyprus Station on the Docklands Light Railway (DLR). The DLR runs east – west and Cyprus Station is the circle at the bottom of figure 4.23.

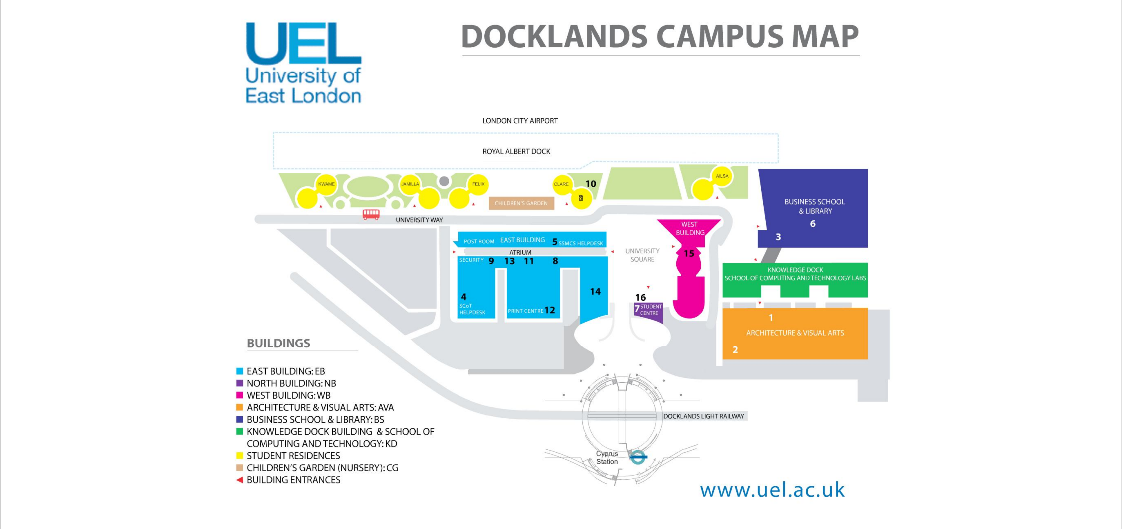


Figure 4. 23: University of East London [www.uel.ac.uk]

Clearly upon arriving at Cyprus Station one has to head north to get to the Campus. The visitor knows for certain which is north since they know which way they were travelling on the DLR and there are visual clues from the River Thames which is south of the DLR and can be seen at numerous points on the journey east from central London, although not from Cyprus Station itself.

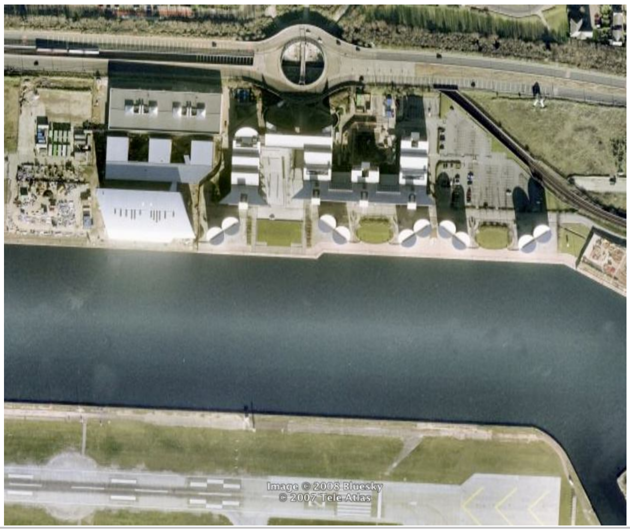


Figure 4. 24: University of East London [Google Earth, 2008]

However, upon arriving at Cyprus Station and heading north, one finds that the UEL is not there and after frantic searching for signs one eventually realises that one has to travel south – see the satellite photograph in figure 4.24. This is a tremendous shock, one feels disorientated and has to rotate ones mental map through 180º.

In fact the Campus can only be approached from the north as it is bordered on the south by the Royal Albert Dock with the London City Airport on the other side of the dock. Thus the reason that the map designers put south at the top is that it is intended for a person walking south. There is possibly some logic in this; at least for the sort of person who turns maps around to put the top is in the direction they are moving. Clearly the problem can be partly solved by putting a north sign on the map (pointing downwards).

This rather trivial example shows that in some respects knowledge of one's orientation is more important than knowing one's location.

Hospitals

These problems can occur even with the healthiest, most mentally alert person. But for the elderly and physically or mentally disabled it can be so much worse. Consultancy firms such as Carpman Grant Associates specialise in advising architects and building owners on this issue. As Carpman Grant Associates say on their website, [www.wayfinding.com] 'Or think about an elderly patient who takes pride in her punctuality, already ten minutes late, as she slowly moves her walker down one corridor after another in a mazelike hospital, trying to find the Imaging Department for the first time.'

A design driver

Initially one might think that the wayfinding problem can be solved after a building design is complete by the provision of suitable signs, maps etc. But the more one thinks about it the more one realises that it is a central if not the central problem of building design, particularly for buildings visited by the general public.

4.4.7 The internet and satellite navigation systems

The Internet, Satellite Navigation Systems, Geographical Information Systems, etc. are systems that offer the possibility to the users, car drivers or pedestrians to find their way towards their 'virtual' or actual destinations. The following paragraphs discuss the use of such systems and make connections with the pedestrian way-finding process.

4.4.7.1 The Internet

The Internet is a publicly accessible system of interconnected computer networks that can pass information between any two computers that are connected to it. Computers are geographically concentrated according to population and economic conditions as shown in the image in figure 4.25.

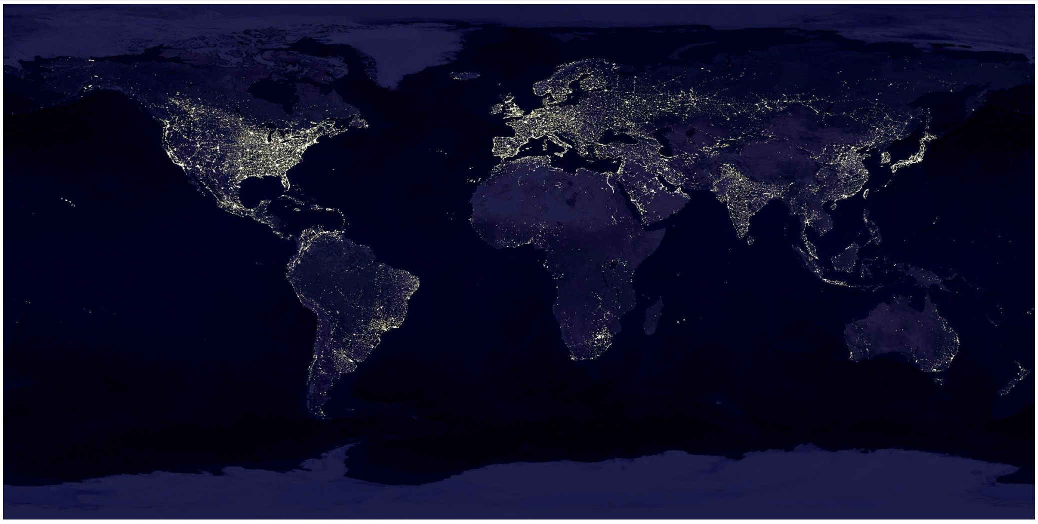


Figure 4. 25: The Earth at night. Image: NASA [http://apod.nasa.gov/apod/ap001127.html]

The traffic on the internet is primarily between these concentrations as shown in figure 4. 26.

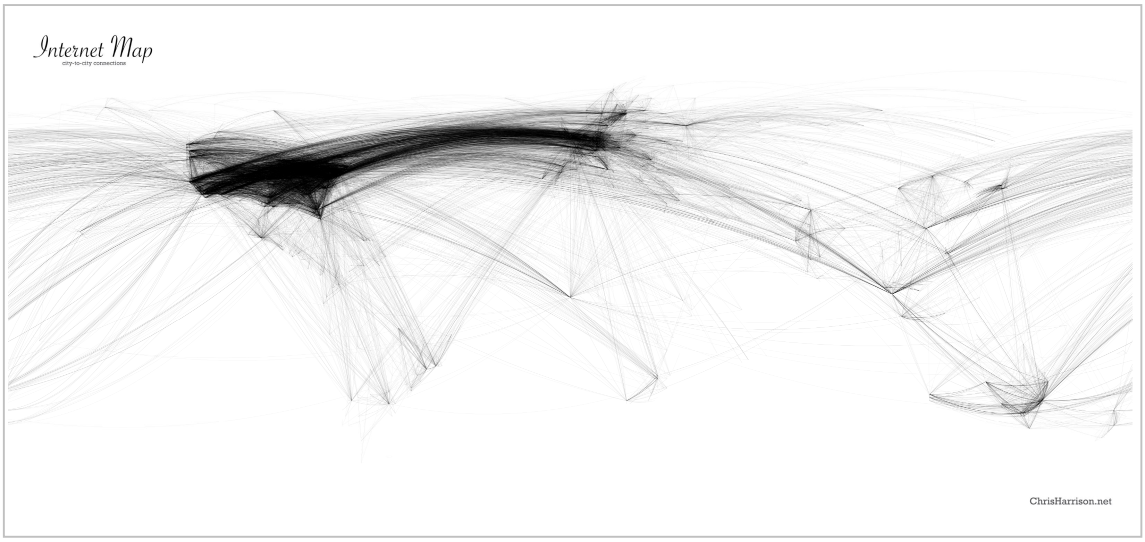


Figure 4. 26: Internet traffic. Image: ChrisHarrison.net

The information is passed from node to node along wires or optical fibres, by microwave radio relay and via satellite. The switching of signals at the nodes is done by computers that are called routers.

Routers use sophisticated software to decide where to send the digital information using the internet protocol (IP). They use packet switching, which means that the connection is only made when information is being sent. The opposite of packet switching is circuit switching in which a connection is maintained even when no data is being sent, for example on a conventional telephone line when both parties pause speaking.

The Internet is not controlled by individual company or government, but various organizations meet to agree technical standards. Domain names are assigned by the Internet Corporation for Assigned names and Numbers (ICANN), a United Stated of America non-government organization. There is a controversy over this fact.

The World Wide Web is a system of interlinked, hypertext documents accessed via the Internet. It was created in 1989 by Tim John Berners-Lee, working at CERN (Organisation Européenne pour la Recherche Nucléaire) in Geneva. The word ‘hypertext’ was introduced in 1965 by Ted Nelson ‘to mean a body of written or pictorial material interconnected in such a complex way that it could not conveniently be presented or represented on paper’.

4.4.7.2 Satellite Navigation Systems

The Internet is primarily about topology, in how the nodes of the network are connected to each other. Geometry is relevant only in planning the links of the networks, crossing cities, oceans or mountain ranges.

Satellite navigation systems are primarily about determining geometry, where you are. Global Navigation Satellite System (GNSS) is the generic term and Global Positioning System (GPS) (officially NAVSTAR GPS) was developed by the United States Department of Defense. In 1983 the GPS system became free for civilian.

The GPS receiver needs a time signal from 4 satellites. It then calculates its position in space and time (x, y, z, t) by comparing these signs that differ due to that fact that they have traveled different distances at the speed of light. The GPS receiver needs information about the position of the satellites which is sent at with the time signal.

4.4.7.3 Geographic Information Systems

In order to guide a pedestrian or car driver the GPS receiver also needs information about the position of seas, mountains, towns and roads, that is a Geographic Information System (GIS). This information is provided by companies such as the US based Navteq Corporation or the Netherlands based Tele Atlas. The GIS information can either be stored locally, perhaps on a DVD, or sent via the Internet or mobile telephone technology.

4.4.7.4 Relevance to pedestrian circulation

The Internet is a virtual analogue of a journey undertaken by a person. The person would:

i. Walk to the bus stop,
ii. Take the bus to a railway station,
iii. Catch a train to the airport,
iv. Fly to another airport,
v. Catch a train,
vi. Catch a bus,
vii. Walk to the hotel.

At each end of the journey the transport is ‘local’ in the middle it is ‘long distance’. Before undertaking the journey the person would study maps and timetables and at each transport intersection or node they would follow signs telling them how to board the vehicle that will take them to the next node. If we were to imagine that each person had a label around their neck giving their final destination and they made no plans prior to leaving home, then the equivalent of the Internet would be that at each node the ‘router’ would read the person's destination and instruct them to take an appropriate route, and if that route was at full capacity or broken the router would suggest an alternative.

Maps and our memories are the equivalent of a Geographic Information System, but we do not carry around with us the non-digital equivalent of a Satellite Navigation System, although traditional navigation instruments, the sextant, chronometer and altimeter could achieve a similar objective. However we do have an optical system (eyes and brain) that is superior to any computer vision.

Discussion of these digital technologies is relevant to our analysis of pedestrian circulation for two reasons. Firstly the digital world provides us with an analogy of the real world, enabling us to better understand the real world thorough thinking about the virtual. Secondly digital technologies are increasingly influencing our behaviour. The introduction of the mobile phone means that we can be very vague about defining exactly when and where we will meet our friends and colleagues. We call each other to compare information about where we are so that modifications can be made of our plans.

Complex organization like Universities no longer need central information points for visitors since visitors now rely on maps and other information they have downloaded from the Internet before leaving home, or, increasingly that they download to their mobile telephones.

Given the sophistication of technology, it is surprising that we still have to stare at Departure screens to see which Gate to go to and when to catch our plane, rather than have that information sent automatically to our mobile telephones. Each new technology at first seems a waste of time and money and then quickly becomes indispensable.

It is essential that circulation design is considered at the conceptual stage of the design process and the role of architect is to find solutions that take into account ease of way-finding. However, research into way-finding has been mainly on the evaluation of a design rather than on processes to generate a new design.

4.4.8 Way-finding techniques

The research that has been done in the area of pedestrian dynamics has largely ignored way-finding problems. Helbing et al. [2005] stated:

> ‘for reliable simulations of pedestrians crowds we do not need to know whether a single pedestrian, say, turns to the right at the next intersection […] This can be either empirically measured or calculated by means of a route choice model’

Obviously, the examination of the route choice behaviour was beyond the scope of pedestrian dynamics research. However, pedestrian movement in space is influenced not only by the social interaction and communication between pedestrians but also by the decision to follow the route towards a particular destination.

Shopping in a supermarket or visiting an art gallery or roaming aimlessly in buildings may affect the behaviour of pedestrians. Also, the destination selection is not always predetermined but can be influenced by other factors such as time available, personal choice, curiosity, interesting views, configuration of the environment, etc.

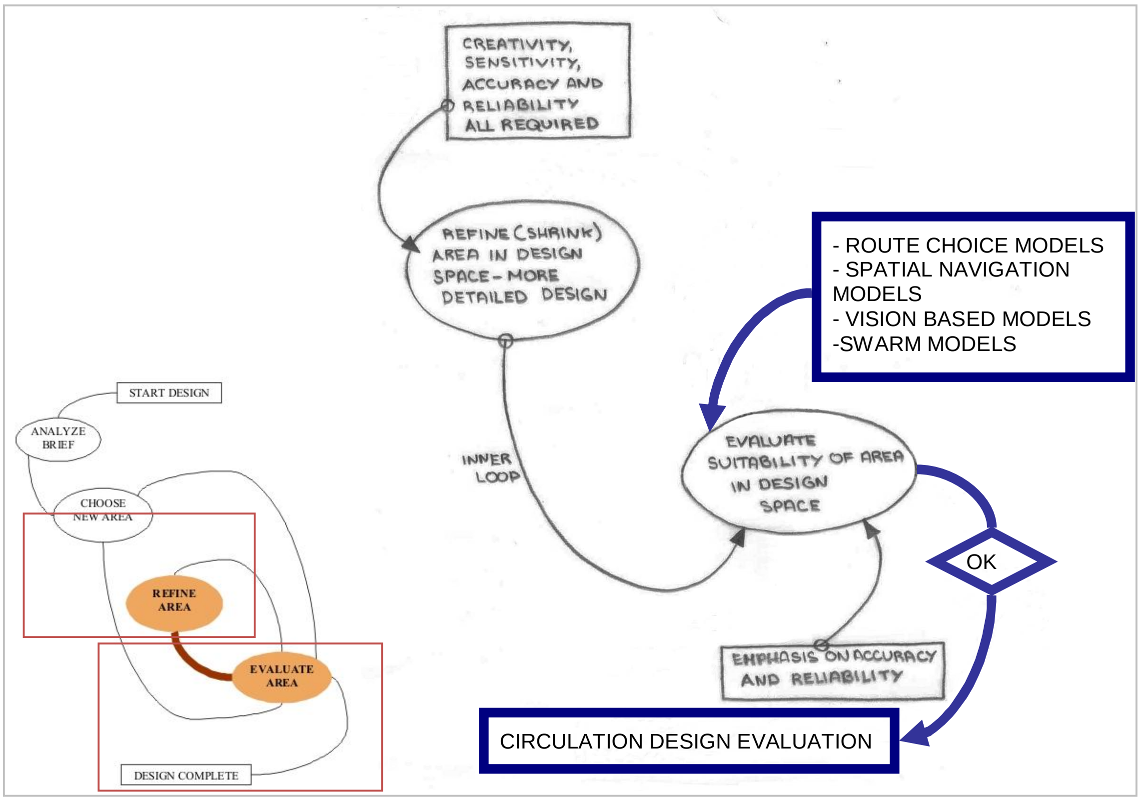


Figure 4. 27: Evaluation of circulation design based on way finding techniques

The complexity of human movement does not allow for the simulation of all possible aspects of behaviour. However, different modelling techniques have been applied to investigate way-finding. It has been found that way-finding, in contrast with social interaction and communication, needs the application of more complicated methods than the simple local rules between ‘automata’. A global knowledge of the destination goal might be necessary in order to simulate such behaviour. Current research into way-finding is divided into three categories according to the techniques applied:

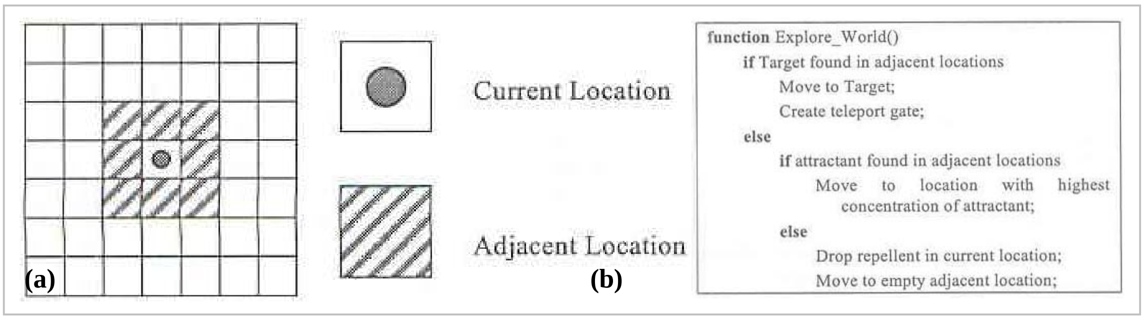
1. Collective intelligence models – in complex systems
2. Route choice models – in pedestrian modelling based on optimization
3. Space syntax models – in pedestrian modelling based on spatial configuration

Again such techniques have been mainly used to evaluate circulation design and movement behaviour in existing building environments (see figure 4.27).

4.4.8.1 Collective intelligence models

Collective intelligence techniques have been examined in section 4.4.4 dealing with the social interaction and communication behaviour. However, swarm algorithms have been also used as way-finding tool to aid users in a virtual environment In this case, pedestrians are replaced by virtual creatures and their movement is based on simple rules derived from ant behaviour [Yoon and Maher, 2005].

Specifically, the swarm algorithm has been created around the concept of the foraging ant behaviour where the aim was to find the shortest path to a food source and then bring the food back to the nest. The movement of a creature was based on a current location grid and on adjacent locations that were the eight surrounding grids. The creature had sensors that sense food, pheromones, obstacles, and other creatures, and effectors that allowed them to deposit a pheromone. Figure 4.28a shows the creature’s current location and the adjacent locations.



```
function Explore_World()
    if Target found in adjacent locations
        Move to Target;
        Create teleport gate;
    else
        if attractant found in adjacent locations
            Move to location with highest
            concentration of attractant;
        else
            Drop repellent in current location;
            Move to empty adjacent location;
```

Figure 4. 28: a. Sensing adjacent locations, b. Algorithm 2: Explore world algorithm [Yoon and Maher, 2005]

Figure 4.28b describes the sequence of steps that the virtual creature followed to reach food and return back to the nest. In any step the creatures checked if the target (food) was located in the next adjacent grid. If the target was found, then the creature moved to the target. Otherwise, it sensed for pheromones. If a pheromone was found, the creature was attracted and then moved to the adjacent grid, and if not, it dropped a repellent and moved to a new adjacent grid.

The importance in this technique is that the virtual creatures could find a target without any outside control. Also, the swarm algorithm could be applied in a dynamically changing environment [Yoon and Maher, 2005].

4.4.8.2 Route choice models

In pedestrian modelling, way-finding techniques might combine local rules of optimization and global knowledge of the destination goal.

Route choice models have been used to identify the optimized pedestrian shopping behaviour in shopping areas [Kurose et al., 2001; Borger and Timmermans, 1986]. Also, the technique has been applied to the investigation of pedestrians' shortest routes in areas like shopping malls, airports, train stations, etc. using minimizing cost functions [Hoogendoorn et al., 2001].

Hoogendoorn et al. [2001] simulated human behaviour in airport terminals by the application of origin (O) and terminal destination (D) areas where pedestrians might enter and leave facilities respectively, and a trip-generation level where pedestrian trips were generated. The trip generation model was the relation between origin-destination points (OD) and was influenced by a number of parameters like activity choice behaviour, trip-chain and leg performance. These parameters represented a number of predetermined assumptions that were taken into account for the modelling of behaviour.

The origin (O) and the terminal destination (D) areas were stable. The number of a pedestrian's choices in between was infinite and the problem was basically an optimal control problem where each pedestrian was trying to find the path that will optimize the trip. In terms of the rules applied, the approach was developed based on an optimal walking direction at each location in the considered area, and was determined by solving a partial differential equation derived from dynamic programming [Hoogendoorn et al., 2001].

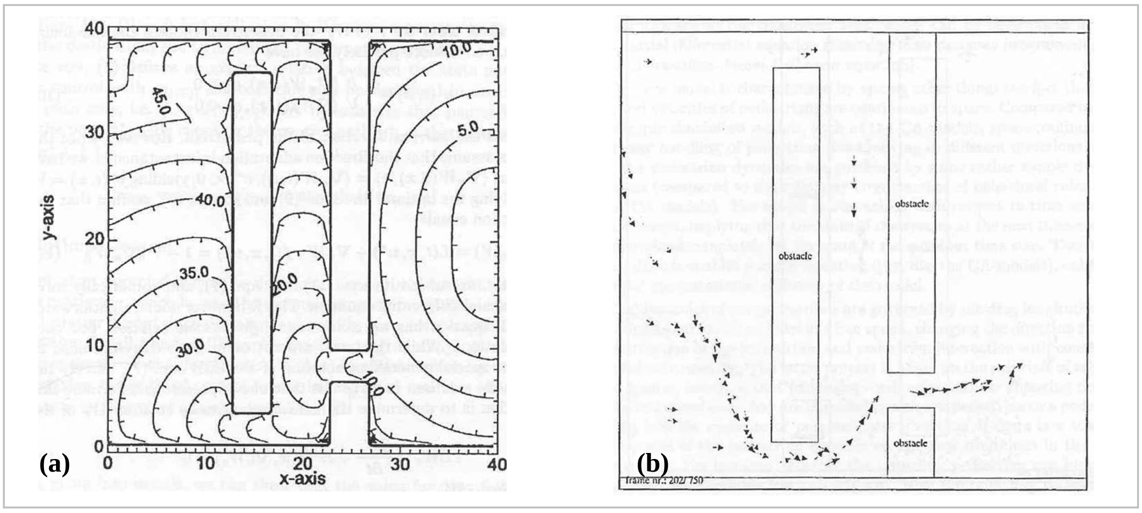


Figure 4. 29: a. Example value function W(t,x) assuming that V=1.5 m/s and destination at x1=40 and x2 somewhere between 17.5m and 22.5. The numbers indicate the minimal travel time (in seconds) to the destination. The lines perpendicular to the iso-cost curves define the optimal directions of the pedestrian, b. Snapshot from pedestrian simulation in case pedestrian choose the route optimizing the expected travel time [Hoogendoorn et al., 2001]

Hoogendoorn’s [2001] model was found to be highly correlated with empirical observation of route choice behaviour. Also, the model produced other characteristics of the pedestrian flows, such as self-organization of dynamic lanes and clusters. Even if the destinations were predetermined, the route selection movement was chaotic and was depended on the initial conditions and the microscopic behaviour. However, the model was mainly used for the analysis and the observation of existing small architectural areas using highly sophisticated algorithms.

Empirical studies, again based on route-choice mechanisms, especially the investigation of pedestrian shopping behaviour in shopping malls [Golledge, 1995; Garling and Garling, 1998], have shown that pedestrian path choice is not always the optimized route but has many fluctuations due to psychological or other parameters.

According to Turner and Penn [2002], members of the ‘space syntax’ group, pedestrians do not take always the shortest path when they move in the building environment.

4.4.8.3 Space Syntax models

The Space Syntax group was created by Bill Hillier and Julienne Hanson at The Bartlett School of Architecture, University College London, and is dedicated to studying techniques for the spatial analysis of buildings and cities [http://www.spacesyntax.com/]. Space Syntax is distinct from Shape Grammar (discussed in section 3.4), which creates or analyzes architectural plans using shape rules. In contrast, Space Syntax examines architectural spaces by decomposing them into fundamental components and then analyzing them as network of relations (views, paths, etc).

i. Space navigation and configuration models

The Space Syntax research group has examined way-finding behaviour based on a combination of local rules of navigation and global knowledge of the environment. Peponis et al. [1990] used Space Sytax to perform empirical way-finding studies in Homey Hospital and suggested a model based on the three parameters shown in figure 4.30.

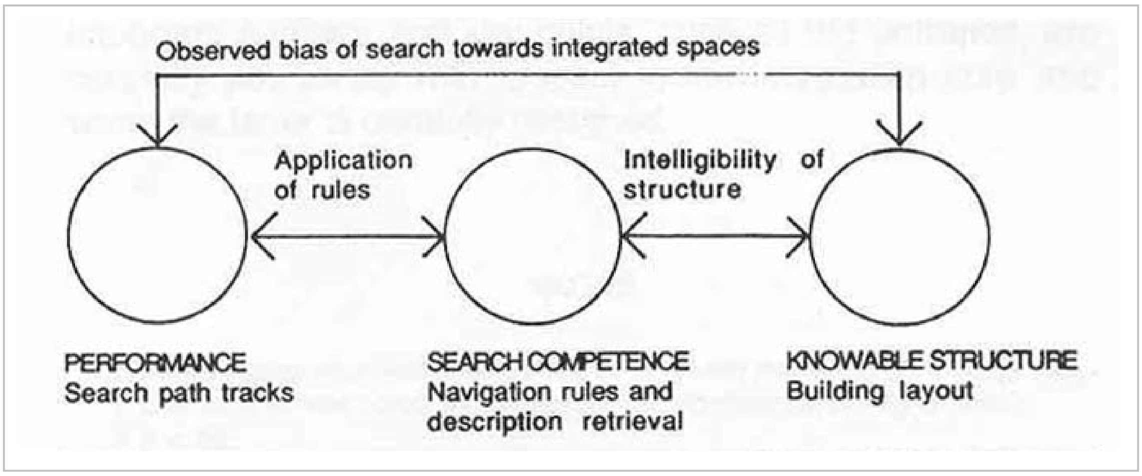


Figure 4. 30: A Preliminary Model of Wayfindig [Peponis et al., 1990]

First there is the abstract and knowable structure of building layout. This represents the relationship of each space to the global structure of a building. Second is the movement performance of users based on search path tracks, which are structured in accordance with integrated patterns. Third are the navigation rules that are mediated between the two first parts.

According to Peponis et al. [1990], the navigation rules are used when the pedestrians search the buildings for the first time without previous knowledge of the space. The pedestrians tend unconsciously to carry out ‘parsimonious rules’ for spatial navigation, which are the simplest rules in order to find their way in unknown environment.

The rules are listed as: 1. Avoid unnecessary backtracking, 2. If all else is equal; continue along the same line, and 3. Divert from the line of movement when a new view allows you to see more space and activity or provides a longer view and lets you see further ahead.

The importance of the overall structure of buildings in the way-finding has been also noted by Hillier et al. [1993]. They suggested that pedestrian movement is influenced by the configuration of the environment and especially the configuration of the urban grids, which can be seen as generators of human movement [Hillier et al., 1993].

Similar conclusions have been drawn in research by Penn and Dalton [1994] on modelling natural pedestrian movement. Their research has shown that reliable simulations can be achieved taking into account the configuration of the environment. The experiments were done using individuals called ‘rats’ and applying random rules of next destination selection. They found that the ‘rats’, when subjected only to simple movement behavioural rules, often took long and inefficient paths. As they put it:

> ‘Simple rules seemed liable to give rise to failure to reach goal. Intuitively, it seems likely that people use many different rules, replacing failing rules with other rules as needed’

They concluded that a real person may need a form of ‘global knowledge’ of the built environment to reach destinations. People know generally their direction and goal, thus people movement rules need to cover clear direction towards their final goals. In conclusion, they stressed the importance of the global knowledge and the spatial configuration in order to achieve the ‘goal-directed movement performance’.

ii. Vision based models

Turner and Penn [2002] introduced a vision based approach to the modelling of agent movement using visibility graph analysis. The movement behaviour rules were conceptualized from Gibson’s [1986] theory of visual perception and movement was afforded by the available walkable surface. The agent movement rules presupposed the choice of a location to walk towards through a stochastic process. The approach has been called EVA (exosomatic[10] visual architecture) because the movement was afforded by an exosomatic architecture that was a visibility graph out of the agent’s body. The visibility graph was constructed using a two-dimensional grid over a plan layout and it calculated which points can ‘see’ which other points. Each point had a set of visible locations that was stored, and it was used to calculate the approximate viewable area. Figure 4.31a shows the visible location on the visible graph grid.

---

[10] Goonatilake’s [1991] concept of ‘exosomatic’ information, that is information stored in a durable form external to the body, is discussed in Bates [2006].

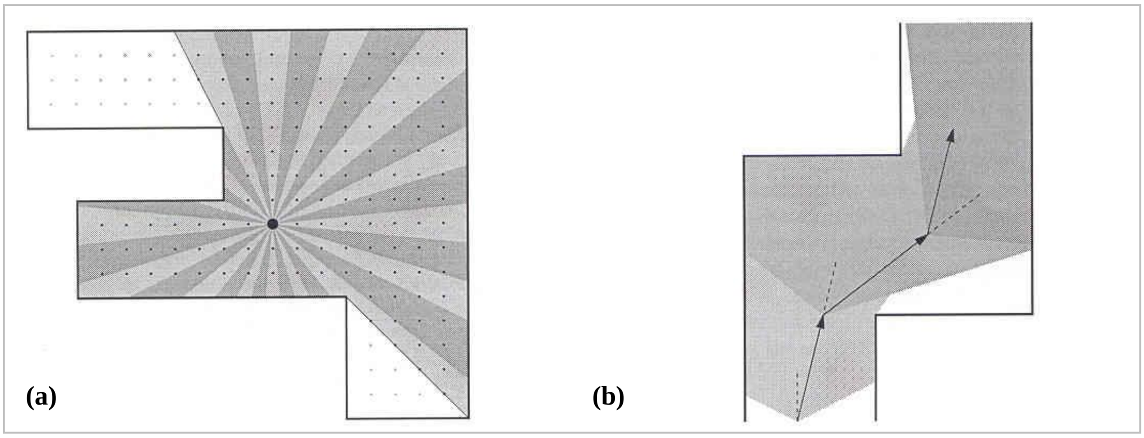


Figure 4. 31: a. Visible location on the visibility graph grid are split between 32 angular bins, b. Theory: selecting destinations from the visibility graph through a stochastic process should draw the agent through a configuration [Turner and Penn, 2002]

The basic rules were described as: a. Pick a visibly graph vertex from the field of view by selecting any vertex from F with equal probability. b. Take, on average *n* steps towards that vertex, based on the Poisson distribution. The space and especially the visibility graph were the guiding mechanisms that directed the movement of agents. Thus, any movement behaviour was influenced by the spatial configuration. The application of the model to Tate Britain using a large number of agents has shown that the rules applied may capture pedestrian movement similar to that found in actual buildings [Turner and Penn, 2002].

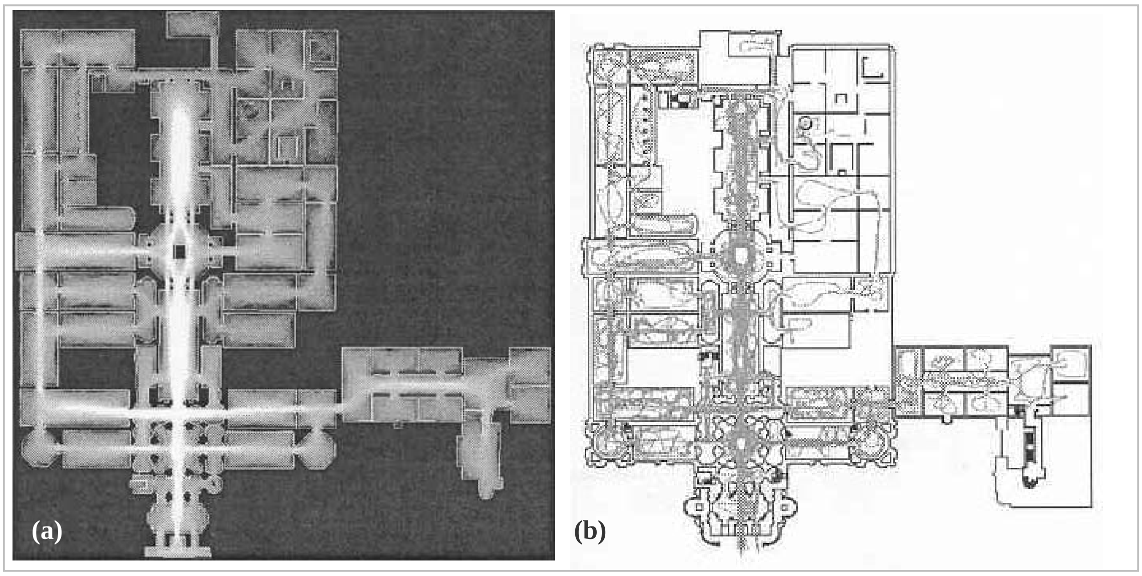


Figure 4. 32: a. Trails left by agents walking through Tate Britain Gallery, Millbank. As each agent steps on a grid square it increments a counter. Black areas have low counts and white areas have high counts, b. Actual movement traces for 19 people followed for the first ten minutes of their visit to the gallery (reproduced by Hillier et al., 1996, page 15) [Turner and Penn, 2002]

4.5 Pedestrian modelling in design

Human movement behavior is influenced by a large number of factors that may be impossible to simulate or investigate fully. Techniques including Social Force, Cellular Automata, Route choice, and Vision based modeling can only be used to study simple social interaction and way-finding. The selection of rules is fundamental to the process as they control the way the pedestrian agents navigate the architectural environment.

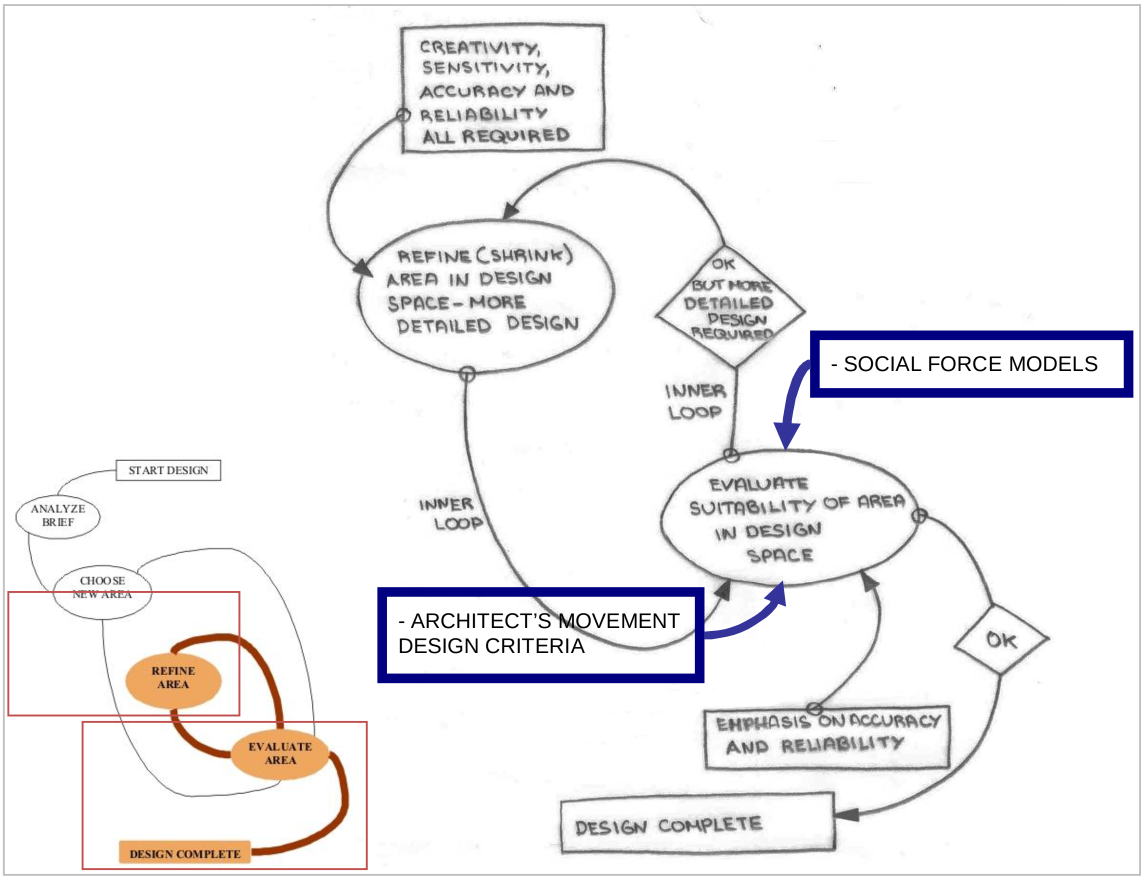


Figure 4. 33: Design pedestrian facilities using pedestrian simulation and architect's movement design criteria

These techniques are largely used to evaluate existing buildings or proposed designs, rather than as generative tools in which the design emerges from the process. Thus in our Design Framework such applications can be represented as part of the feedback loop mechanism refining the design area in design space as shown in figure 4.33.

However generative design processes have been applied for the evolutionary optimization of bottlenecks showing that method can be applied not only to improve existing designs but also to suggest new design solutions [Helbing et al., 2001].

Turner and his co-workers used techniques based on visual perception in parallel with generative design approaches to solve spatial configuration problems [Turner et al., 2004]. Agents were programmed to move initially in a completely open space and wander around. Then gradually a random configuration of walls was applied allowing the agents to move in different rooms as shown in figure 4.34.

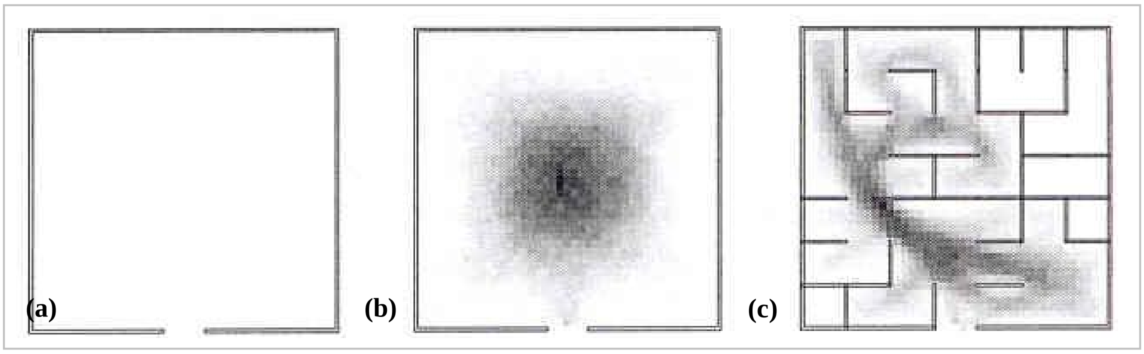


Figure 4. 34: a. The environmental context, a space notionally about 28mx28m with an entrance, b. Agent trails from a ten-minute exploration task. The areas are darker where more agents have trodden, c. A randomly generated configuration, again with ten-minute agent trails [Turner et al., 2004]

The building environment was given the task of dispersing the movement of agents as much as possible by rearranging the position of walls in each cycle. A simple evolutionary algorithm was used to calculate the penalty function that was the summation of all differences between the average number of agents who were passing through all the rooms and the number of agents who passed each room in particular. The system was changed using a standard genetic algorithm and the population consisted of hundreds environments that were generated automatically [Turner et al., 2004].

Another example was the research has been done by Helbing et al. [1997a; 1997b; 2001] on trail formation and specifically, the development of trail systems in deformable grounds. Helbing et al. [2001] tried to find answers concerning how trails can be formed by pedestrians and how urban planners can design public way systems so the walkers can use them.

Based on empirical observations and by extending the Social Force Model, Helbing et al. [2001] introduced an *active walker model*. This was a pedestrian simulation model where the walkers were subjected to the fluctuations of their environment. The pedestrians had rules to change their environment, which again influenced their movement and behaviour. It was basically an indirect interaction approach between the pedestrians or active walkers and the environment, which, according to Helbing et al., ‘may lead to the self-organization of large-scale spatial structures’.

Figure 4.35 shows the generation of trail systems. Figure 4.35a illustrates the initial trail pattern where the pedestrians start in a random position in time, and they form all possible connection movements towards the four possible fixed destinations. In the beginning, the pedestrians take the direct route to their destinations. Then, gradually the pedestrians use existing path lines because they are more comfortable and attractive. The attractive paths become important and finally the unused ones are gradually destroyed. This has the result of the formation of paths that reduce the overall length of the trail system (see figure 4.35b) [for further details, see Helbing et al., 1997].

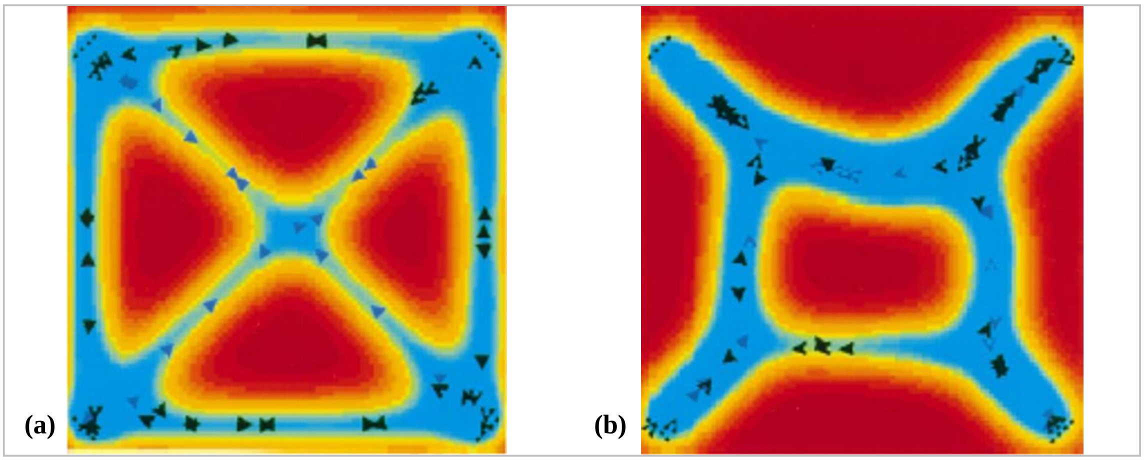


Figure 4. 35: Starting with a plain ground, the structure of the trail system changes considerably during the simulation. Initially, pedestrians use the direct ways (left). Because frequently used trails become more comfortable, a bundling of trails sets in which reduces the overall length of the trail system (right) [Helbing et al., 1997, p. 48]

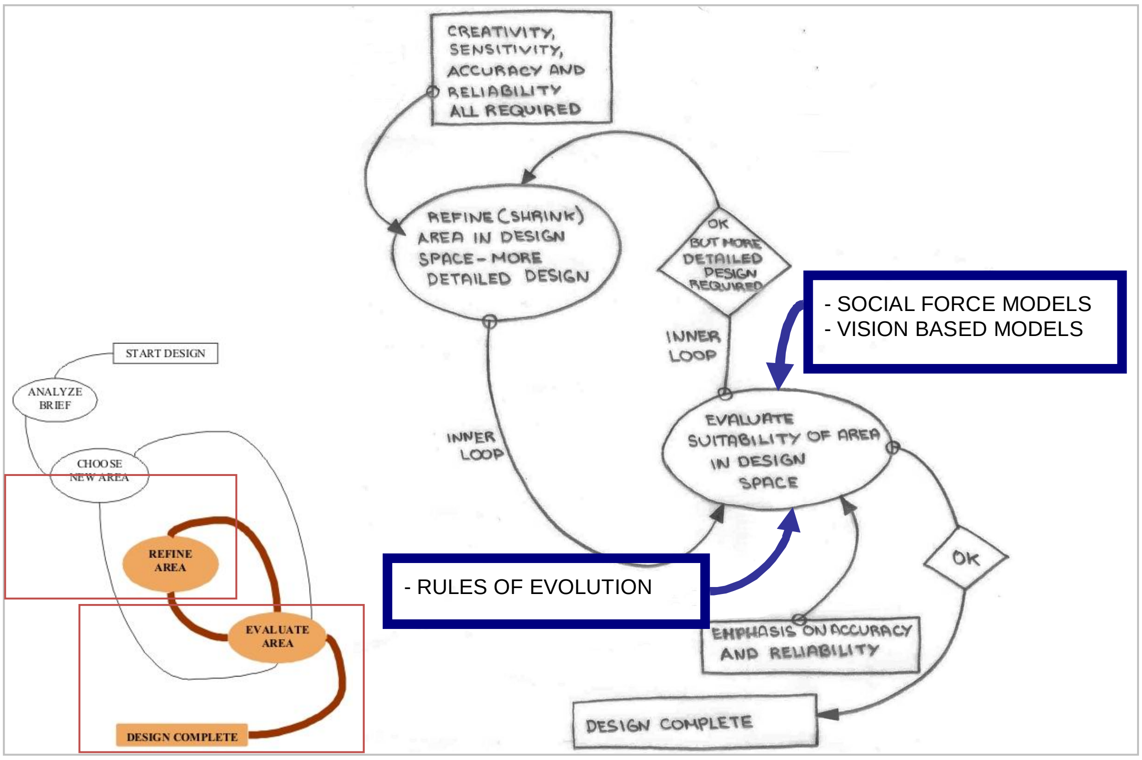


Figure 4. 36: Pedestrian simulation and optimization in design framework

Pedestrian simulations and the evolutionary optimization techniques are applied in parallel in the ‘evaluation’ stage of design in order to refine the detailed design as shown in figure 4.36. The solution navigates the design space based on both, the pedestrian movement and the evolutionary optimization rules.

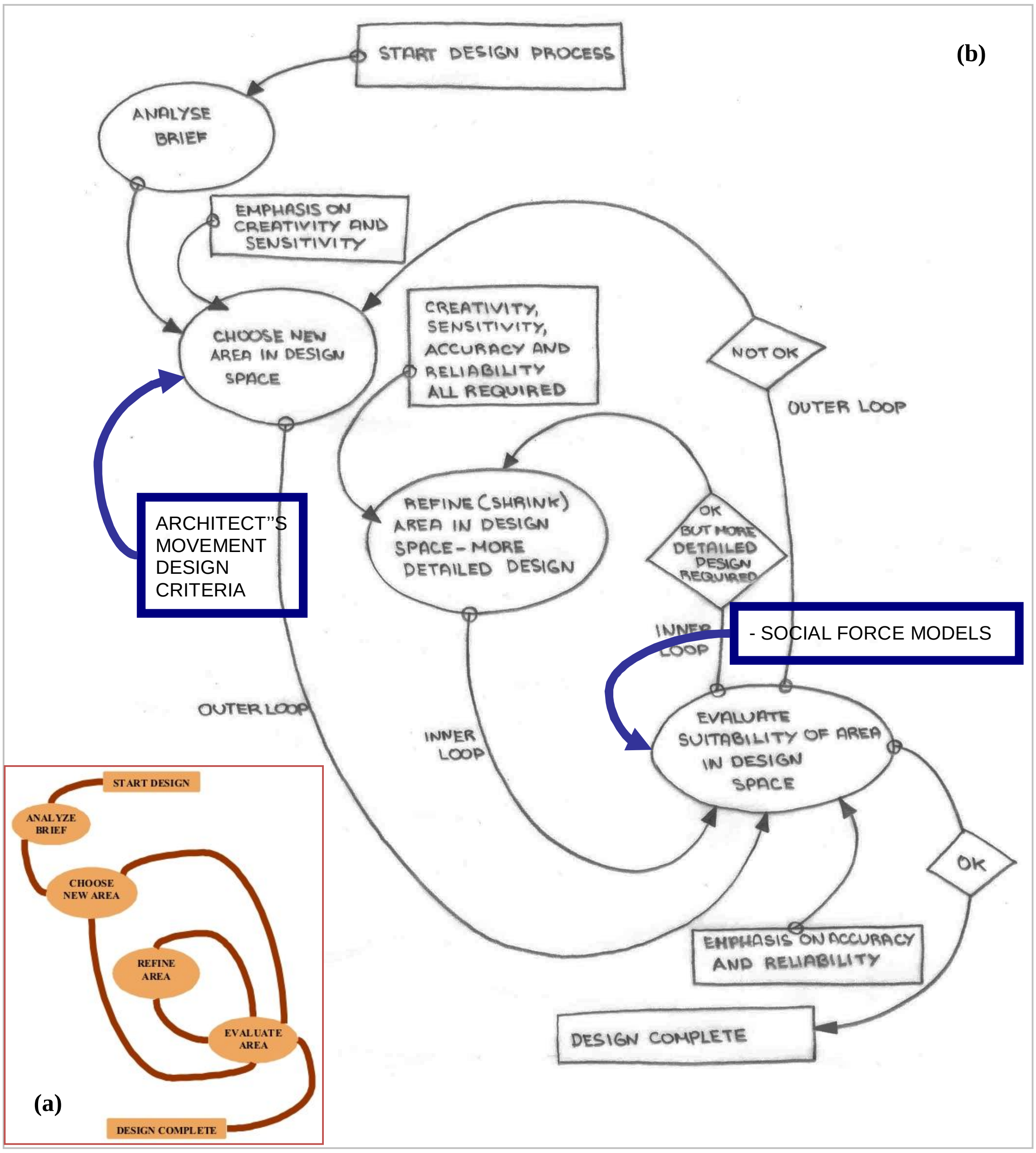


Figure 4. 37: a. Ideal connections in the Design Framework, b. Suggested Design Framework that connects ‘outer’ loop of scheme design, and ‘inner’ loop of detailed design with evaluation part

In any architectural project where circulation is a central design driver, the movement criteria that need to be satisfied are detailed and complex and architects cannot fully describe or sufficiently evaluate their project only by conventional design approaches. Computers can help in the generation and evaluation of circulation design, but just as in structural or other areas of design, computer analysis and optimisation do not remove the need for the architect or engineer to have a simple, clear conceptual model.

The architects can navigate through solutions in design space or in the design framework (figure 4.37) using ‘rules of navigation’, which are analogous to pedestrian movement rules.

Chapter 5: Mathematical model and computer code

5.1 Basic concepts

In order to establish a language of discussion, first we need to introduce some basic concepts that will be used in our research. Such concepts are velocity, mass, density, etc. Similar concepts can be found in areas like fluid dynamics where fluids are represented either as particles or as continua. Thus, analogies between the flow of fluids and people will be drawn.

In the flow of fluids we can have the following equation that represents the mass flow rate through a given tube:

$$Mass\ flowrate = Density \times Cross\text{-}section\ area \times Velocity$$
$$\text{kg/second} = \text{kg/m}^3 \times \text{m}^2 \times \text{m/second}$$

Figure 5.1 shows this relationship in diagrammatic form.

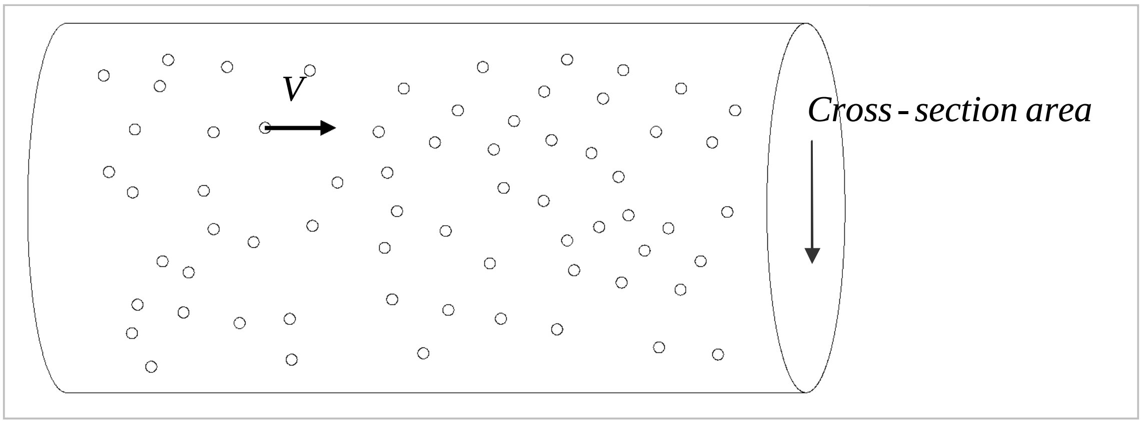


Figure 5. 1: Flow of fluids

In the flow of people a similar concepts can be applied. We assume that a number of people occupy a certain given area with $W_{\text{path width}}$ as shown in figure 5.2.

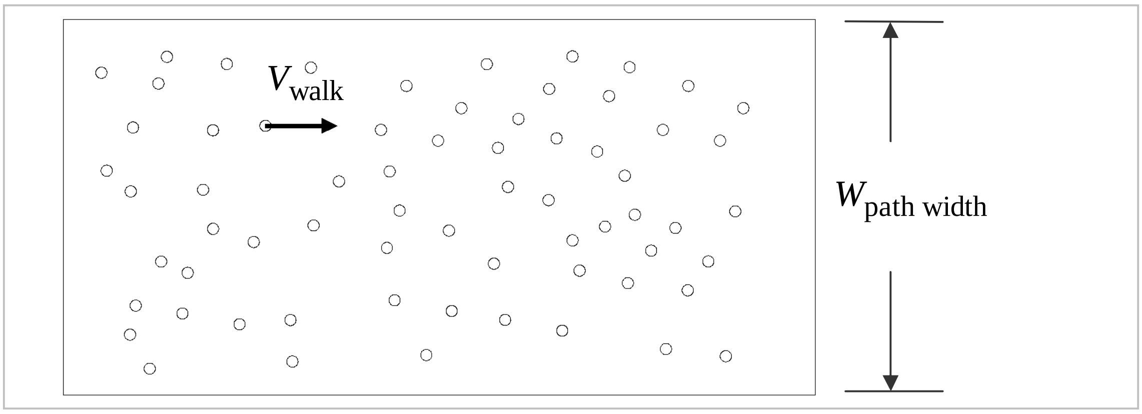


Figure 5. 2: Flow of people

The people flow rate is given by the following equation:

$$People\ flow\ rate = Density \times Path\ width \times Velocity$$

$$\text{person/sec} = \text{person/m}^2 \times \text{m} \times \text{m/s}$$

where the density is the number of people per unit area which is $\frac{1}{\text{area per person}}$.

And hence the equation can be written as follows:

$$N_{\text{people flow rate}} = \frac{V_{\text{walk}}\, W_{\text{path width}}}{A_{\text{person}}}$$

where N the people flow rate, V the velocity, W the path width, and A the person area.

In the following example we investigate how the path width can be calculated using our equation.

If $N_{\text{people flow rate}} = 10 \,{}^{\text{people}}\!/_{\text{sec}}$

$V_{\text{walk}} = 2 \,{}^{\text{m}}\!/_{\text{sec}}$

$A_{\text{person}} = 1\text{m}^2$

Then we can get $W_{\text{path width}} = 5\text{m}$.

In the case of steady flow, the total number of people passing a point during a given time interval is simply the flow rate times the time interval. If the flow is unsteady, the total number of people passing is the area under the flow rate against time curve as shown in figure 5.3.

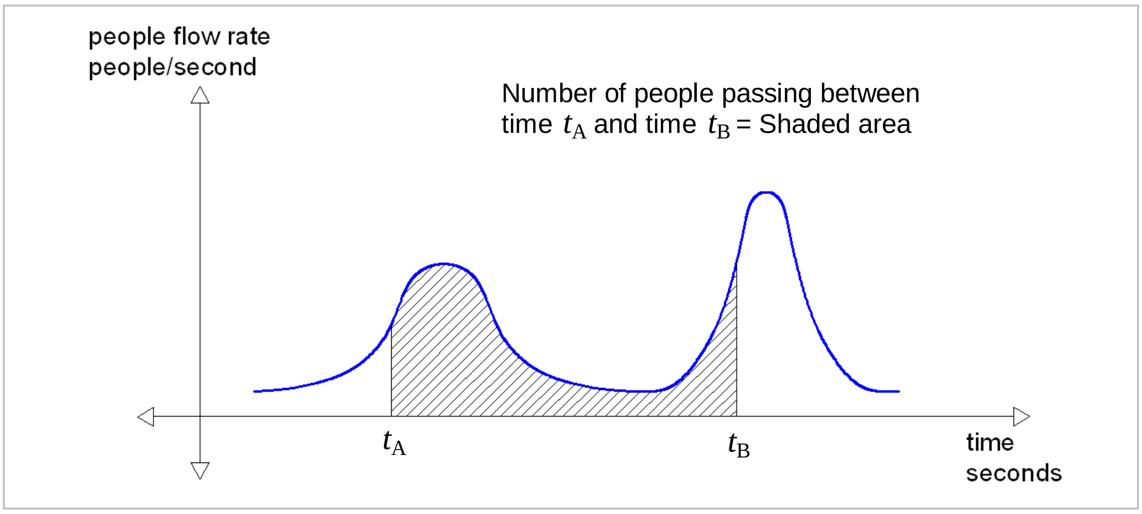


Figure 5. 3: People flow rate in unsteady situation

This idea can be used to investigate the relation between people flow rate and time in various areas of buildings where people flow is central aspect of the design investigation.

Following examples show how such concepts can be applied and discussed.

It is very difficult to change the density of water and water is therefore usually assumed to be *incompressible*. Air is more compressible, but pressures associated with even the strongest winds only produce density changes of the order of 1%. Thus when air flows along a duct, if the cross-sectional area is reduced, the velocity must increase since the density remains constant.

However when people walk along a corridor, they do not speed up when the corridor narrows, and instead their density increases, that is the area per person decreases. If the people flow rate is too high a queue will form at the start of the constriction, only allowing people through at a certain rate.

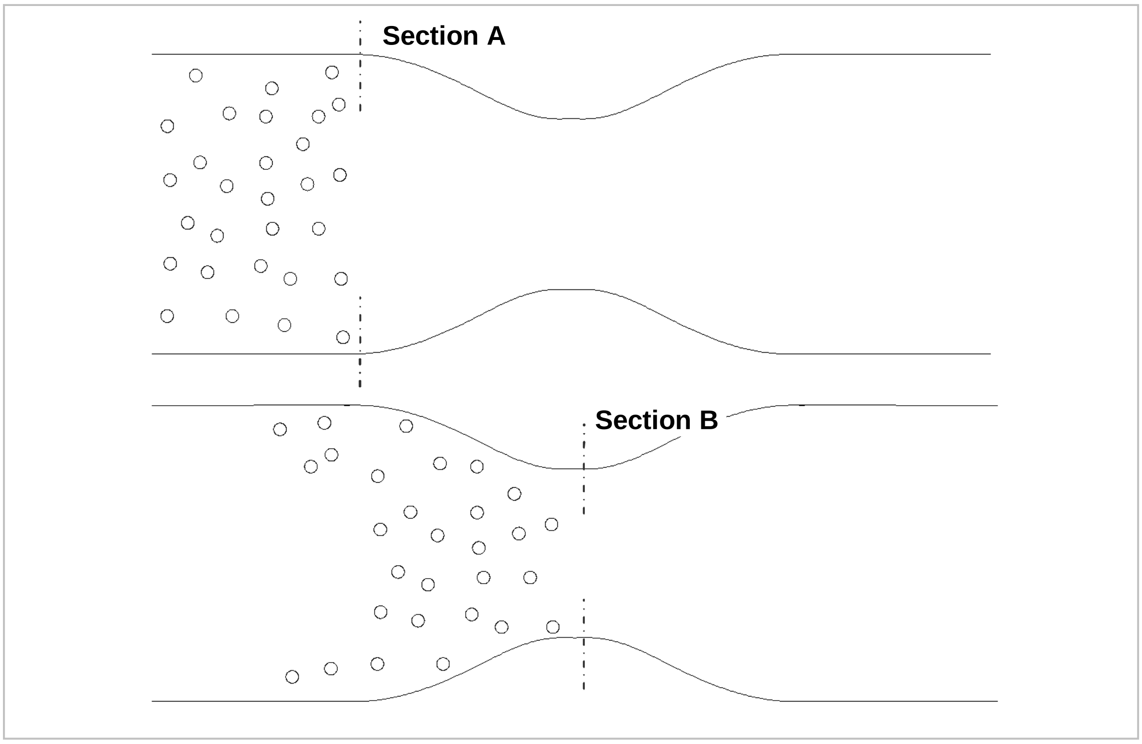


Figure 5. 4: People flow through corridor

The upper drawing in figure 5.4 shows a slug[11] of people arriving at constriction in a corridor. As they squeeze through the constriction the *peak* flow rate is decreased as shown in lower drawing in figure 5.4. However the *total* number of people passing each

---

[11] The Oxford English Dictionary [OED Online, July 2008] gives one of the definitions of ‘slug’ as ‘A compact mass of liquid regarded as retaining its identity as it travels’. The concept is used in medicine and pipeline design.

point is the same so that the area under the two curves is the same as shown in figure 5.5.

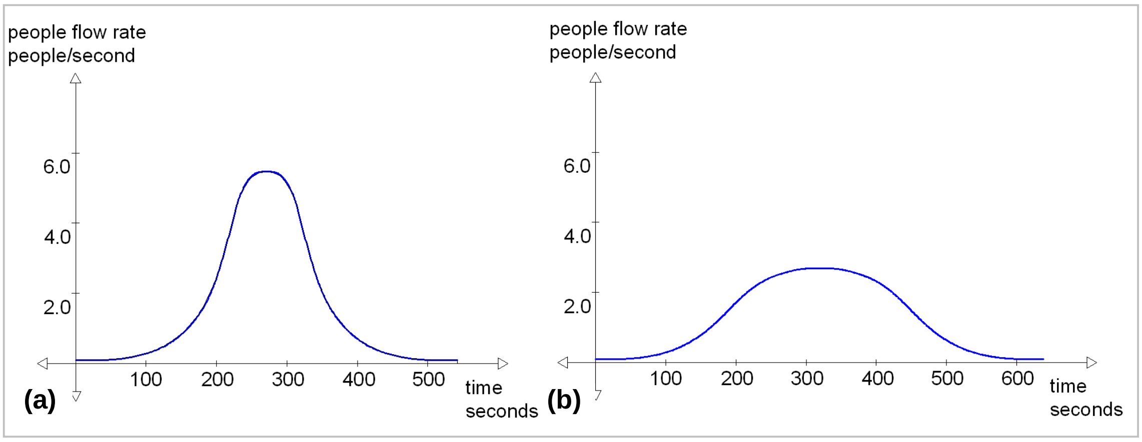


Figure 5. 5: People flow rate – time relationship in corridor, a. Section A, b. Section B

In the second example we assume that people are in the arrival hall area of a terminal building and they wait to pass through the immigration/passport control point as shown in figure 5.6.

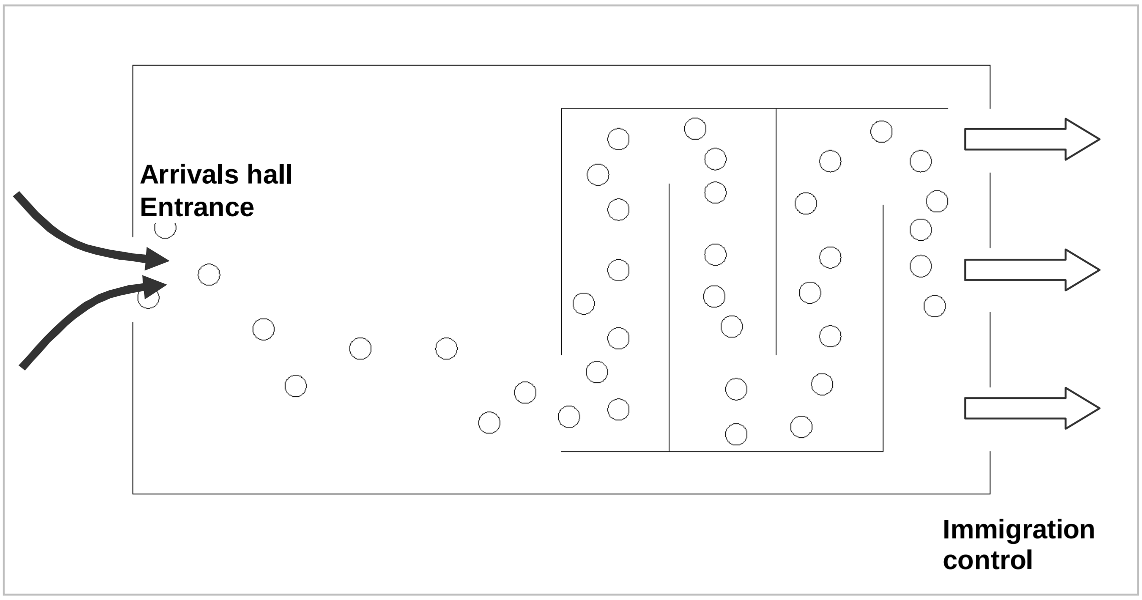


Figure 5. 6: People flow in arrival hall

An ideal situation would be the continual flow of people from the arrivals entrance to the control points in a smooth flow as shown in figure 5.7a.

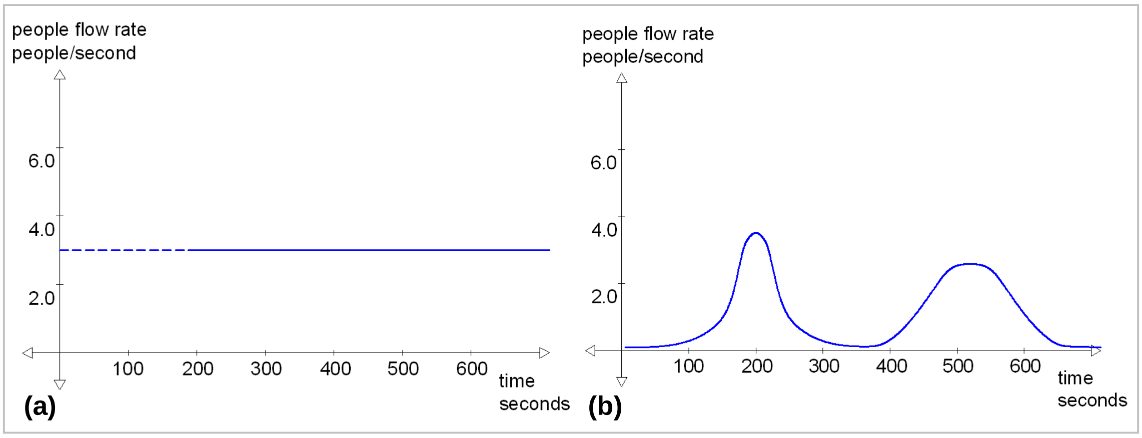


Figure 5. 7: People flow rate – time relationship in immigration/passport control point

However, in airport terminals due to the arrival of individual aircraft, the people flow rates fluctuates and can be represented with peak points as shown in figure 5.7b. However the flow rate through the departures security check is much smoother.

5.2 The Virtual Force model

Newton’s laws of motion

The Virtual Force model is based upon Newton’s laws of motion as formulated in the *Philosophiae Naturalis Principia Mathematica* (1687):

> **Newton's first law: law of inertia**
>
> Lex I: *Corpus omne perseverare in statu suo quiescendi vel movendi uniformiter in directum, nisi quatenus a viribus impressis cogitur statum illum mutare.*
>
> Every body perseveres in its state of being at rest or of moving uniformly straight forward, except insofar as it is compelled to change its state by force impressed.
>
> **Newton second law: law of rate of change of momentum**
>
> Lex II: *Mutationem motus proportionalem esse vi motrici impressae, et fieri secundum lineam rectam qua vis illa imprimitur.*
>
> The rate of change of momentum of a body is proportional to the resultant force acting on the body and is in the same direction.

Newton's third law: law of reciprocal actions

> Lex III: *Actioni contrariam semper et æqualem esse reactionem: sive corporum duorum actiones in se mutuo semper esse æquales et in partes contrarias dirigi.*
>
> All forces occur in pairs, and these two forces are equal in magnitude and opposite in direction.

Newton's laws apply to all bodies, non-living and living. The momentum of a body is the product of its mass and its velocity,

$$\mathbf{p} = m\mathbf{v}$$

Here the momentum, $\mathbf{p}$, and the velocity, $\mathbf{v}$, are *vectors*, that is they have both direction and magnitude. In printed work vectors are usually represented by bold, non-italic characters. The mass, m, is a *scalar*, that is it has a magnitude only. In printed work scalars are usually represented by italic, non-bold characters. Hand written work usually involves putting bars or arrows over vectors due to the difficulty of distinguishing bold and italic characters. Some printed work does the same, but if anything, it makes it more difficult to read and is not commonly done.

The easiest way to represent a vector is in terms of its Cartesian components:

$$\mathbf{v} = v_x\mathbf{i} + v_y\mathbf{j} + v_z\mathbf{k}.$$

v_x, v_y and v_z are the components of $\mathbf{v}$ in the x, y and z directions. $\mathbf{i}$, $\mathbf{j}$ and $\mathbf{k}$ are unit vectors, that is vectors with unit magnitude in the x, y and z directions. Even though v_x, v_y and v_z are italic, non-bold, they are not scalars[12], they are the components of a vector and change if we change the orientation of the axes.

The rate of change of momentum is equal to

$$\begin{aligned}\frac{\text{the change in momentum}}{\text{the change in time}} &= \frac{\delta\mathbf{p}}{\delta t} \text{ as } \delta t \text{ tends to } 0\\ &= \frac{d\mathbf{p}}{dt}\end{aligned}$$

This is using the terminology of the differential calculus, discovered independently by Newton and Leibniz. There will be very few references to calculus in this dissertation.

---

[12] A scalar is a quantity whose value at a point in space and time does not depend upon the orientation of the coordinate system. Clearly v_x, v_y and v_z depend upon the choice of coordinate system.

In certain problems, such as throwing a ball, the mass does not remain constant, the single mass (person + ball) splits into two (when an astronaut throws a ball while floating in space, the momentum of the astronaut and of the ball change, but in opposite directions, so that the total momentum remains constant).

However if the mass does remain constant we have

$$\frac{d\mathbf{p}}{dt} = m\mathbf{a}$$

where the acceleration vector, $\mathbf{a}$**,** is equal to the rate of change of velocity,

$$\mathbf{a} = a_x\mathbf{i} + a_y\mathbf{j} + a_z\mathbf{k} = \frac{d\mathbf{v}}{dt}$$

$$a_x = \frac{dv_x}{dt}$$

$$a_y = \frac{dv_y}{dt} \quad .$$

$$a_z = \frac{dv_z}{dt}$$

Thus, if the mass is constant, the second law states that the force, $\mathbf{f}$, is proportional to $m\mathbf{a}$. If the constant of proportionality is taken as 1.0 we have

$$\mathbf{f} = m\mathbf{a}$$

which leads to the definition of the Newton as the force required to accelerate one kilogram mass at one metre per second per second and the poundal as the force required to accelerate as one pound mass at one foot per second per second. It also leads to the definition of the slug as the mass, which when acted upon by a pound force, accelerates at one foot per second per second.

A position vector is a vector describing the position of a point. It is a vector from the origin to the point and its components are simply the x, y and z coordinates of the point:

$$\mathbf{r} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}.$$

Velocity is the rate of change of position:

$$\begin{aligned}\mathbf{v} &= \frac{d\mathbf{r}}{dt}\\ v_x &= \frac{dx}{dt}\\ v_y &= \frac{dy}{dt}\\ v_z &= \frac{dz}{dt}\end{aligned}.$$

Pedestrian movement

People move about in two dimensions, except when swimming underwater. They may move up or down hills or stairs, but they are essentially constrained to move on a flat or curved two dimensional surface ‘embedded’ in three dimensional space. Peoples’ movements are subjected to Newton’s laws and the forces acting on their bodies are *real* forces. These forces not only act on bodies as a whole, but on parts of bodies, arms, legs etc. leading to the science of biomechanics. Conservation of angular momentum allows skaters and dancers to spin.

The Virtual Force model imagines the existence of an imaginary or *virtual* force which then causes the person to exert a *real* on the ground, so changing their velocity and momentum. These virtual forces represent the conscious and unconscious desires of the individual as filtered by social norms and taboos. There is no need for Newton 3rd law to apply. If person A is attracted to person B, there is (unfortunately) no reason why person B should be attracted towards A. Of course the 3rd law does apply to the real forces, person A pushes against the ground as they accelerate towards B, so accelerating the Earth in the opposite direction.

Thus an individual can be acted upon by a combination of real and virtual forces, the only difference being that the virtual force requires the individual to translate the virtual force into its real equivalent. The term ‘virtual’ force can be taken to include ‘social’ force as a special case. Thus a person might be drawn to the smell of fresh bread emanating from a bakery, which is not really a ‘social’ force.

Walking over a hill

Imagine a person walking over a hill towards a fixed destination. Further imagine that the hill is grassed and that there are no paths on which the person is constrained to move. If the person follows a straight line on plan they have to climb to the top of the hill. On the other hand if they follow a contour line they will have to walk a lot further. Thus, looked at in plan, or on a map, the person is subjected to two forces, a force pointing downhill trying to stop them climbing too high and a force drawing towards their destination. Thus the chosen path is a compromise between too much climbing and walking too far. Different people would choose different paths based on their age and fitness and whether or not they were in a hurry. There would be other forces, if they found themselves moving too fast they would want to slow down and so on. The level of sophistication of the model is arbitrary. One might, for example say that it is easier to walk alternatively up and down in a zig-zag pattern, than to follow a contour line which requires one leg to be bent.

Mass

The effect of a force on a body depends upon its mass. If we expect a certain force to affect some people more than others, then we can either adjust the value of the force or the mass. Since we do not require that Newton's 3rd law applies, we do not have to apply equal and opposite forces on individuals, so we can adjust the forces at will. Thus the mass that we use in Newton's 2nd law can either be the person's real mass or some virtual mass.

Impulse

An impulse is an infinitely large force exerted over an infinitely short time, such that the change of momentum of the mass that receives the impulse is finite. Examples include the impact of billiard balls. The force is not infinite and the time is not infinitely short, but the time of contact is very small compared with the time it takes for a ball to roll even 1mm. Clearly impulses can be incorporated into a virtual force model.

Large numbers of people

It is easier to model the behaviour of a group of people than that of an individual. This particularly applies when the group of people are in a building or public place in which certain sorts of activities and behaviour patterns are to be expected. Then one can concentrate on the virtual forces which one might expect to dominate, finding a nice spot to lie in the sun in a park (or in the shade on a hot day) or getting onto an aeroplane at an airport.

Notation

Notation is not particularly important, especially since the calculations will almost certainly be done in a computer in which words, rather than symbols are used. This is because computer programs do not have bold, italic or Greek letters available to them. Nor do they have superscripts, subscripts, overbars etc. They do have arrays and the components of velocity of the i^{th} person might be written as Vely[i][0], Vely[i][1], Vely[i][2] in which the 0, 1 and 2 correspond to the x, y and z direction. What is clear is that we can devise any system of rules or algorithms with which to calculate the real and virtual forces applied to any individual, such as person i. It is convenient to give people numbers rather than names, a fact which reinforces the unfortunate fact that people are not really treated as individuals. These algorithms can include whatever data we care to chose including the locations and velocities of other individuals, the location of walls, shops, toilets, paths etc. etc. Different people can be given different properties, age, height etc. – in an art gallery tall people would experience a social force drawing them towards the back of a group of people viewing a painting.

Time stepping

Computers cannot handle continues change since that implies an infinite amount of information. Thus the location of an individual can only be recorded at discrete times,

$$...t-3\delta t, t-2\delta t, t-\delta t, t, t+\delta t, t+2\delta t, t+3\delta t...$$

A simulation is started in some configuration at some arbitrary time, often taken as time, $t=0$. It then steps through time, increasing t by δt for each step. The value of δt is a compromise, the larger the value the faster the simulation, but the lower the accuracy. The fastest sprinters travel at 10m/sec and a walker might move at 1m/sec (3.6km/h). Thus for people motion simulation one might expect δt to be of the order of 0.1 seconds, corresponding to a walker moving approximately 100mm.

At time t Newton's 2nd law is $\mathbf{f}_t = m\mathbf{a}_t$ and the differential equations are replaced by the corresponding difference equations:

$$\frac{d\mathbf{v}}{dt} = \mathbf{a} \qquad \mathbf{v}_{t+\frac{\delta t}{2}} = \mathbf{v}_{t-\frac{\delta t}{2}} + \mathbf{a}_t \delta t$$

$$\frac{d\mathbf{r}}{dt} = \mathbf{v} \qquad \mathbf{r}_{t+\delta t} = \mathbf{r}_t + \mathbf{v}_{t+\frac{\delta t}{2}} \delta t$$

These difference equations have been used for many years in different fields. In molecular dynamics it is known as the Verlet Algorithm. Molecular dynamics simulates the motion of molecules in the same way that the motion of people is modelled in techniques such as the social force model.

These equations can be expressed in component form as

$$\frac{dv_x}{dt} = a_x \qquad v_{x_{t+\frac{\delta t}{2}}} = v_{x_{t-\frac{\delta t}{2}}} + a_x \delta t$$

$$\frac{dv_y}{dt} = a_y \qquad v_{y_{t+\frac{\delta t}{2}}} = v_{y_{t-\frac{\delta t}{2}}} + a_y \delta t$$

$$\frac{dx}{dt} = v_x \qquad x_{t+\delta t} = x_t + v_{x_{t+\frac{\delta t}{2}}} \delta t$$

$$\frac{dy}{dt} = v_y \qquad y_{t+\delta t} = y_t + v_{y_{t+\frac{\delta t}{2}}} \delta t$$

Resolution of forces

The components of a force are related to its direction and magnitude by

$$f_x = |\mathbf{f}| \cos\theta$$
$$f_y = |\mathbf{f}| \sin\theta$$

where $|\mathbf{f}|$ is the magnitude of $\mathbf{f}$ and θ is the angle between the force and the x axis as shown in figure 5.8.

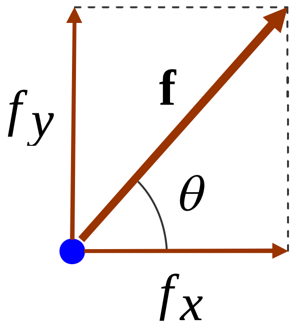


Figure 5. 8: Resolution of force

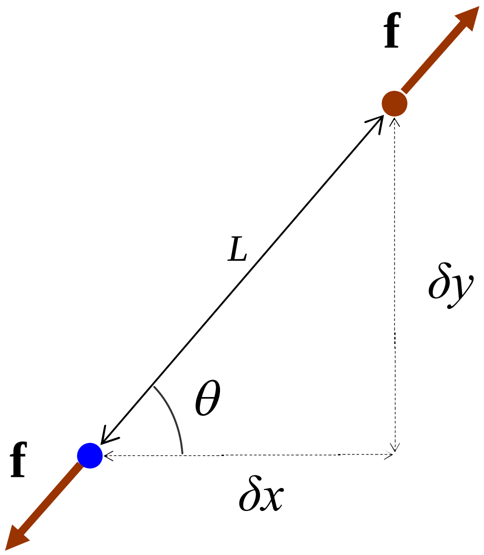


Figure 5. 9: Force acting upon two people

If the force acts along the line joining two people as shown in figure 5.9, then

$$\delta x = L\cos\theta$$
$$\delta y = L\sin\theta$$
$$L = \sqrt{\delta x^2 + \delta y^2}$$

so that

$$f_x = |\mathbf{f}|\frac{\delta x}{L}$$
$$f_y = |\mathbf{f}|\frac{\delta y}{L}$$

Force – distance relationships

The behaviour of people subjected to various forces is described using the diagrams of force – distance relationships. If two strangers approach too close to each other then they will experience a virtual social repulsive force as shown in figure 5.10. The graph combines a real contact force that becomes very large when two strangers come close and a virtual social force that 'dies off' as strangers go away.

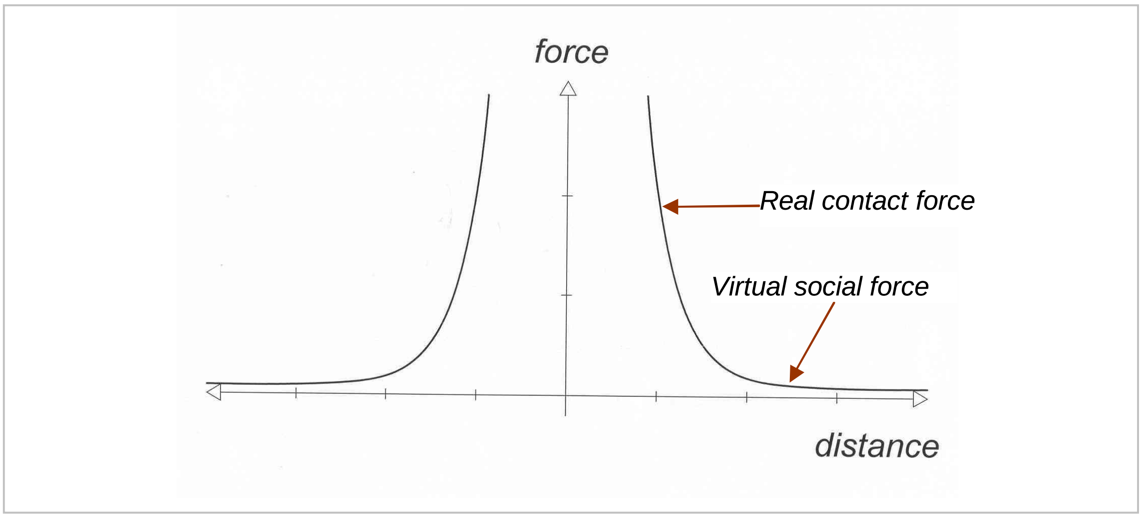


Figure 5. 10: Virtual social repulsive force

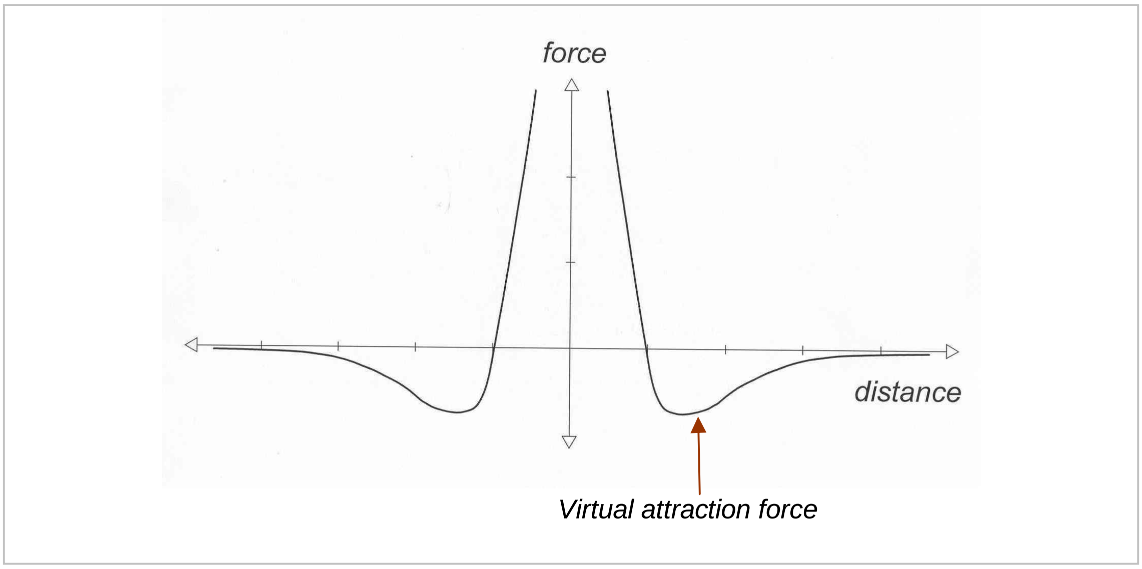


Figure 5. 11: Virtual attraction force

In the case of family groups, a virtual attraction force (negative repulsive force) might occur as shown in figure 5.11.

5.3 Virtual forces and patterns in space – time

The Virtual Force model simulates the behaviour of people in real time and in parallel. The position of individuals changes in each time step due to the parallel interaction between them.

Figure 5.12 shows two examples of a single time step of two individuals that are moving in different directions in a 2D environment.

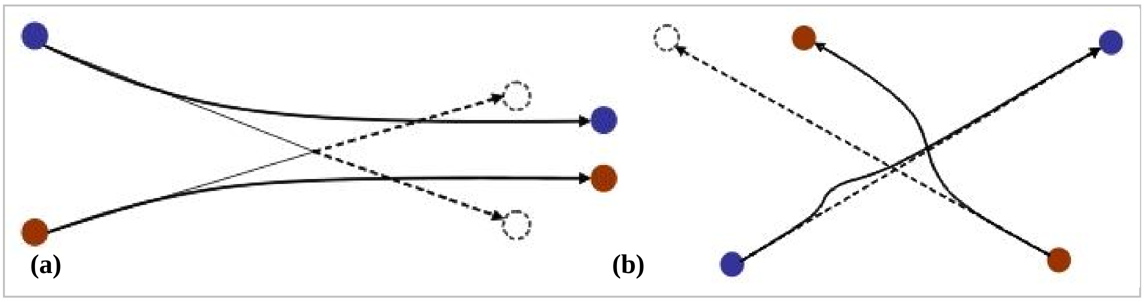


Figure 5. 12: a. Individual behavior before and after collision in horizontal axis, b. Individual movement behavior before and after collision in vertical axis

If a virtual social repulsive force is applied, their direction of movement will be changed. The direction of movement will be influenced by force-distance relationship, their first position and direction of movement. Figure 5.12a shows a possible pattern generated by simple symmetric movement from the left to the right. Figure 5.12b shows the pattern generated by an asymmetric movement from bottom to top.

In real cases, the behavior of people performing collision avoidance is different. They do not act mechanically but according to complex rules which involve predicting their own and other peoples' future positions filtered through social norms. Thus if one thinks that another person will have moved before you get to a particular point, no avoiding action is required. However, in the Virtual Force model (or at least its simple implementation), people react dumbly to the current situation with no attempt to predict the future. As hunting animals people are very adept at these temporospatial calculations.

If the behavior of many individuals is examined, possible patterns can be generated as shown in figure 5.13. The images illustrate two groups (red and blue) that cross and move in different directions. Each individual is subjected to virtual social repulsive force and the direction of movement is similar to the behavior examined in figure 5.12.

The patterns can be illustrated in space-time, a 3D environment where z direction represents time. Such space-time models may be useful for the discussion of the behaviour of individuals in time as a mechanism for simulation and design.

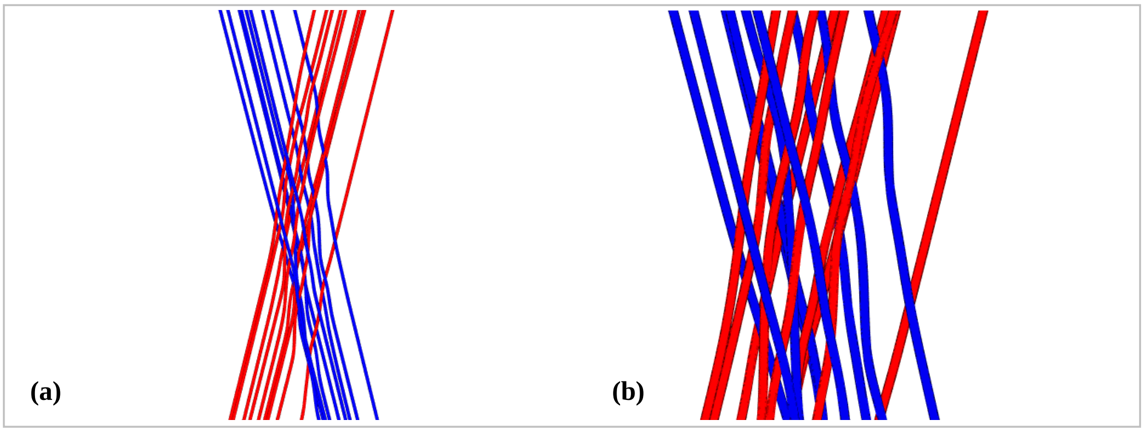


Figure 5. 13: 'Weaving' individual movement behavior in space-time

Individuals' paths 'weave' as shown in figure 5.13. Clearly some weaving patterns are better than others and one may hope for some beneficial pattern to emerge out of the computer modeling or the real people flow situation.

5.4 Modelling platform and programming language

The Virtual Force model was coded (i.e. programmed) mainly using the C++ language [Hubbard, 2000] with OpenGL [Woo et al., 1999]. C++ is a general purpose language widely used in many different fields and one of the most popular programming languages since the early 1990's. It can support and compile variety of programming (object-oriented, generic programming, etc.), other cross-platform computer programs and tools, and it is widely taught in the academic environment.

OpenGL is a multi-platform language (library of graphical functions) developed by Silicon Graphics Inc. It has been used for the creation of programs for virtual reality, computer games, etc. and, in general, for writing applications using 2D and 3D graphics. It has been selected for this research because it is a very powerful graphical interface that provides high quality real-time simulation. In addition, it can be used practically in any computer platform (UNIX, Windows, Macintosh etc).

Recently, a number of other programming languages have been introduced that combine a scripting (i.e. coding) environment with graphical interface. Processing is a new language that is used for design and visualization. It was created by Ben Fry and Casey Reas and 'It evolved from ideas explored in the Aesthetics and Computation Group at the MIT Media Lab'. Its purpose is to teach the basics of computer programming in a visual context and to be used by students, artists, architects, etc. [http://processing.org/]. The programs written with Processing can be rendered using modes like Java 2D drawings libraries, custom 3D engine (P3D) and OpenGL using JOGL interface. Other software packages such as GenerativeComponents by Microstation (Bentley) provides scripting environment in parallel with graphical interface however it has been found very slow to be used as simulator of people motion. Some of the simulations described in this dissertation were programmed in Processing.

Natural languages contain hundreds of thousands of words and have complex grammatical structures. But computer languages have very few words, perhaps 10 or so, and almost no grammar. Computers are very good at repetitive tasks, particularly numerical calculations and processing text, treating the words solely as collections of letters with no meaning. The sophisticated actions of computer programs are produced by possibly thousands of lines similar to those below written in C++. Underneath the program is the output it prints to the computer screen. It lists a group of people and their ages. Then it sorts them alphabetically and prints the list again. Finally it sorts them by age and prints the list for a third time[13].

Program

```
#include <iostream>
#include <fstream>
#include <cmath>
#include <cstdlib>
using namespace std;
```

---

[13] Note that the computer does not print the ø in Søren, instead it prints S\277ren.

```
#define LastPerson 7
#define MaxCharactersPlus1 50
int  Age[LastPerson+1];
char Name[LastPerson+1][MaxCharactersPlus1];
int Swap(int);
void PrintThePeople(void);
int main(void)
{
     strcpy(Name[0],"Ludwig   ");Age[0]=62;strcpy(Name[1],"Bertrand ");Age[1]=97;
     strcpy(Name[2],"Søren    ");Age[2]=42;strcpy(Name[3],"Martin   ");Age[3]=86;
     strcpy(Name[4],"Jean-Paul");Age[4]=74;strcpy(Name[5],"Immanuel ");Age[5]=79;
     strcpy(Name[6],"Friedrich");Age[6]=55;strcpy(Name[7],"Blaise   ");Age[7]=39;
     int TotalAge=0;
     for(int Person=0;Person<=LastPerson;Person++)TotalAge+=Age[Person];
     cout<<"\nSum of the ages of all the people = "<<TotalAge<<"\n";
     PrintThePeople();
     for(int go=0;go<=LastPerson-1;go++)
     {
          for(int Person=0;Person<=LastPerson-1;Person++)
          {
               if(strcmp(Name[Person],Name[Person+1])>0)Swap(Person);
          }
     }
     PrintThePeople();
     for(int go=0;go<=LastPerson-1;go++)
     {
          for(int Person=0;Person<=LastPerson-1;Person++)
          {
               if(Age[Person]>Age[Person+1])Swap(Person);
          }
     }
     PrintThePeople();
     return 0;
}
int Swap(int ThisPerson)
{
     char TemporaryName[MaxCharactersPlus1];
     strcpy(TemporaryName,Name[ThisPerson]);
     strcpy(Name[ThisPerson],Name[ThisPerson+1]);
     strcpy(Name[ThisPerson+1],TemporaryName);
     int TemporaryAge=Age[ThisPerson];
     Age[ThisPerson]=Age[ThisPerson+1];
     Age[ThisPerson+1]=TemporaryAge;
}
void PrintThePeople(void)
{
     cout<<"\n";
     for(int
Person=0;Person<=LastPerson;Person++)cout<<Name[Person]<<"\t"<<Age[Person]<<"\n";
}
```

Output from running program

```
Sum of the ages of all the people = 534

Ludwig          62
Bertrand        97
S\277ren        42
Martin          86
Jean-Paul       74
Immanuel        79
Friedrich       55
Blaise          39

Bertrand        97
Blaise          39
Friedrich       55
Immanuel        79
Jean-Paul       74
Ludwig          62
Martin          86
```

```
S\277ren        42

Blaise          39
S\277ren        42
Friedrich       55
Ludwig          62
Jean-Paul       74
Immanuel        79
Martin          86
Bertrand        97
```

The line

```
for(int Person=0;Person<=LastPerson;Person++)TotalAge+=Age[Person];
```

adds up the ages of all the people. The ‘++’ (from which the name C++ is derived) means that ‘Person’ is increased by 1 each time. += means that Age[Person] is added to TotalAge.

The line

```
if(Age[Person]>Age[Person+1])Swap(Person);
```

goes to the function Swap() if the Person is older than Person+1. Swap() interchanges the two people in the list.

The lines

```
          for(int Person=0;Person<=LastPerson-1;Person++)
          {
               if(Age[Person]>Age[Person+1])Swap(Person);
          }
```

goes through the entire list, swapping if appropriate.

The lines

```
     for(int go=0;go<=LastPerson-1;go++)
     {
          for(int Person=0;Person<=LastPerson-1;Person++)
          {
               if(Age[Person]>Age[Person+1])Swap(Person);
          }
     }
```

Goes through the entire list a number of times, sufficient to move the oldest person from the top to the bottom of the list (or the youngest from the bottom to the top) if necessary.

In C++ and most other modern programming languages (including Processing) blocks of program are contained within curly brackets, {….} which can be nested. The for() statement and the if() statement, together with calculation statements like y=sin(x); are essentially the complete palette available to the programmer.

In C++ counting in for() loops is normally started from zero. This might seem odd at first, but a metre rule has 101 centimetre divisions on it: 0, 1, 2,……, 99, 100.

The programs described in this dissertation only involve calculations involving numbers, there is no need to process text. Text is included in the program above to show how text can be treated in a similar way to numbers.

OpenGL commands include lines like:

```
glColor4f(0.0,1.0,0.0,0.5);
glPointSize(2.0);
glBegin(GL_POINTS);
glVertex3dv(CoordinatesToPlot);
glEnd();
```

which draws a dot of diameter 2 pixels at the point specified by the coordinates in the array CoordinatesToPlot. glColor4f() defines the red, green and blue values together with the transparency of the dot.

There are also OpenGL commands for finding out the position of the mouse and when it is clicked.

5.5 General flowchart of the model

The general flowchart of the computer code shown in figure 5.14 remains the same whether using C++, Processing or any other language.

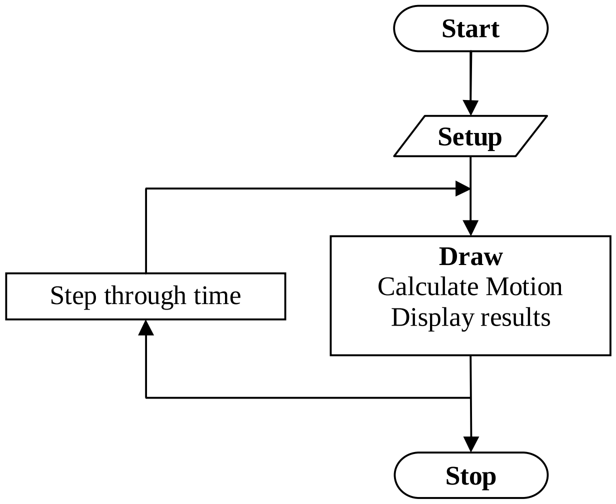


Figure 5. 14: General flowchart of the computer code

Flowchart consists of two main parts:

i. **Setup** where all the input information like individual initial positions, velocity, and time of appearance are defined. Other inputs are geometrical information including boundaries of the system and position of various components such as walls or columns.

ii. **Draw** where the program calculates the motion and displays the results. Virtual or social forces have to be calculated between each pair of people and each person and each obstacle, and added to find the total force on each individual. This involves $\frac{n(n-1)}{2}$ interactions between people where n is the number of people (each of the n people has $(n-1)$ interactions, but we do not need to consider A's interaction with B separately from B's with A). If n is large it makes sense to impose a hidden grid over the space and only consider people in the same or adjacent grids.

Figure 5.15 shows the organisational diagram of the computer code including the various effects that are generated using virtual forces. Figure 5.15 includes the 'Sign effect' which will be introduced in section 5.6.3.1.

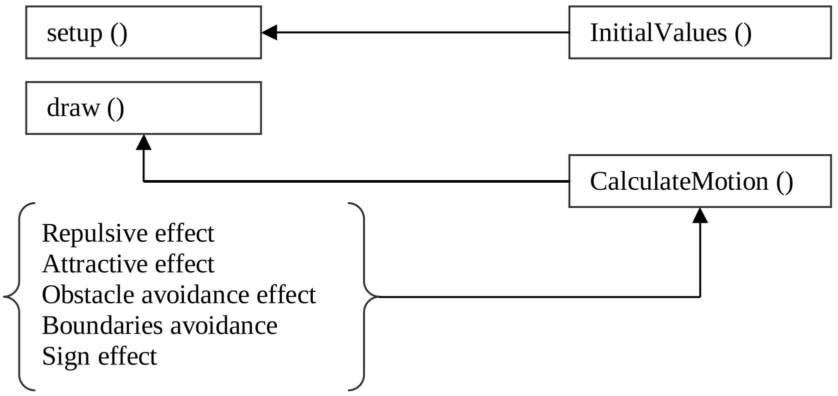


Figure 5. 15: Organizational diagram of computer code

Examples of two simple programs written in C++ and Processing are given in Appendix B, which show the basic structure of the algorithm and the commands used for OpenGL graphics. The program simulates a large number of randomly distributed individuals in the virtual environment.

5.6 Numerical implementation

Following examples will demonstrate various numerical implementations of the Virtual Force model. In these examples different forces might affect people movement behaviour since people are influenced by various interactions with other people and their environment.

5.6.1 Encoding simple movement

The investigation into people behaviour will start by examining the simple movement. Each person is given the same initial velocity. Velocity is a vector so that means that each person starts with the same speed (the magnitude of velocity) and in the same direction. The people are programmed to appear randomly and move from the left to the right boundary. If the position of a person is beyond the right boundary, then the person starts again from the left boundary. The program runs for ever until is terminated by the user. The part of program that generates the simple movement is scripted in *CalculateMotion()* function and it is shown as below:

```
for(Individual=0;Individual<=LastIndividual;Individual++)
{
   Vely[0][Individual]=1.5;Vely[1][Individual]=0.0;
   for(xyz=0;xyz<=1;xyz++)
   {
      Coord[Time+1][xyz][Individual]=Coord[Time][xyz][Individual]+Vely[xyz][Individual]*deltat;
      if(Coord[Time+1][xyz][Individual]>+Boundary[xyz])
      {
         Coord[Time+1][xyz][Individual]=-Boundary[xyz];
      }
   }
}
```

where the velocity in the x direction is constant and the velocity in the y direction is zero. Thus the people movement is from the left to the right boundary as shown in figure 5.16a.

Because in general what happens in the x direction the same as what happens in the y, the loop `for(xyz=0;xyz<=1;xyz++){     }` is used so that the same instructions are followed in both directions. In 3 dimensions one would use `for(xyz=0;xyz<=2;xyz++){     }`.

The if statement:

```
   if(Coord[Time+1][xyz][Individual]>+Boundary[xyz])
      {
         Coord[Time+1][xyz][Individual]=-Boundary[xyz];
      }
```

has the effect of returning people to the left when they disappear to the right.

The line of code:

```
   Coord[Time+1][xyz][Individual]=Coord[Time][xyz][Individual]+Vely[xyz][Individual]*deltat;
```

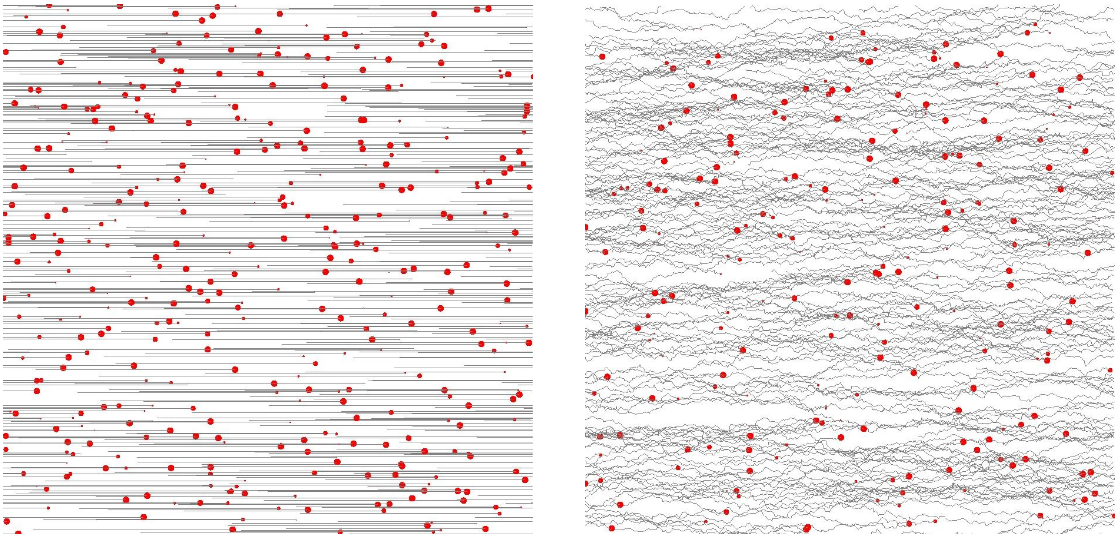
could have been written

```
   Coord[xyz][Individual]=Coord[xyz][Individual]+Vely[xyz][Individual]*deltat;
```

or

```
Coord[xyz][Individual]+= Vely[xyz][Individual]*deltat;
```

in which case the position of a person is simply updated. The reason that the positions of people are remembered is to be able to draw their tracks and hence give a semblance of motion to a static drawing.



(a) People movement with constant velocity

(b) People random movement in y direction

Figure 5. 16: Encoding pedestrian movement

If the velocity is given by a random number generator, then the people motion is random as shown in figure 5.16b. In this example, the velocity in the x direction is constant and the velocity in y direction is random. As a result, random fluctuation of the movement appears in y direction and while the individuals move from the left to the right boundary.

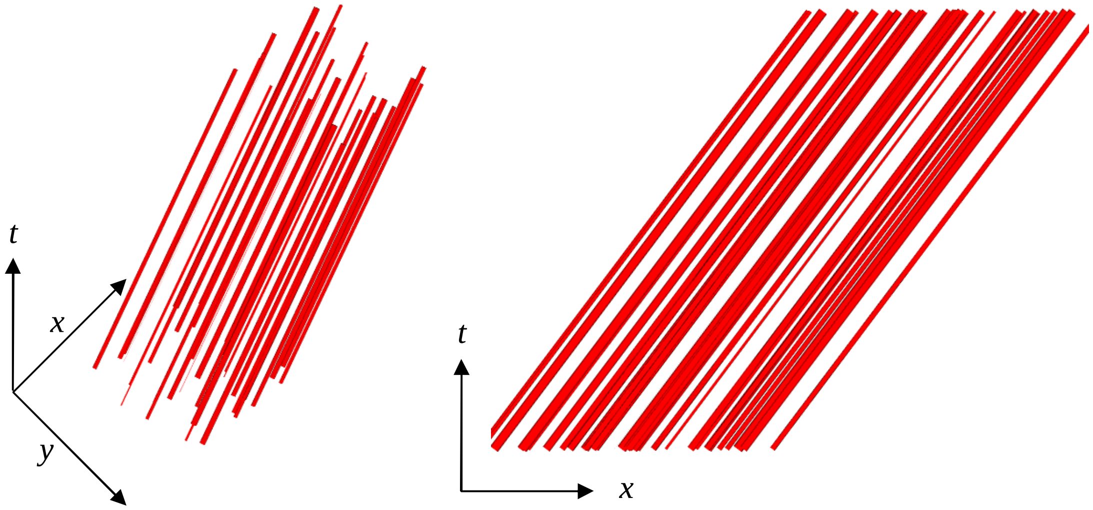


Figure 5. 17: Simple movement behavior in space-time environment

Figure 5.17 shows the simple movement behaviour illustrated in the space-time environment where the vertical direction represents time. The diagrams show people moving in x direction with constant velocity producing straight lines in space-time.

5.6.2 People interaction behaviour

The predictability of people behaviour will be radically changed if the people are subjected to forces from other people or objects while their motion. In the following we will discuss a number of effects and phenomena.

5.6.2.1 Repulsive effect and stripe formation

People movement is influenced by a large number of parameters. People tend to keep a distance from others and they try to avoid obstacles or boundaries.

The repulsive effect includes the repulsive social force and the repulsive real force when people actually touch. The repulsive force dies away with distance and any number of different mathematical functions might be used to describe this. Perhaps the simplest is to assume that the force is proportional to the exponential function $e^{-\frac{L^2}{a^2_{\text{interperson repel}}}}$ where L is the distance between individuals and $a_{\text{interperson repel}}$ is constant length which determines how fast the force dies away as shown in figure 5.18. The shape is the bell-shaped curve familiar from the normal or Gaussian distribution used in statistics and probability theory. Note that there is nothing ‘magical’ about the normal distribution, there is no theoretic reason why interpersonal forces should be exactly this shape, there are many ‘bell-shaped’ curves, $\dfrac{1}{1+\dfrac{L^2}{a^2_{\text{interperson repel}}}}$, $\dfrac{1}{\cosh\left(\dfrac{L}{a_{\text{interperson repel}}}\right)}$ as well as polynomials and the spline curves used for CAD. Numerical ‘experiments’ can be done with different relationships with the aim of producing lifelike behaviour. This is approach is adopted in the animation and computer game world. Whether it is appropriate for architecture is more questionable, but it would seem to be more feasible to propose a model and compare its results with real situations than to measure social force in a laboratory.

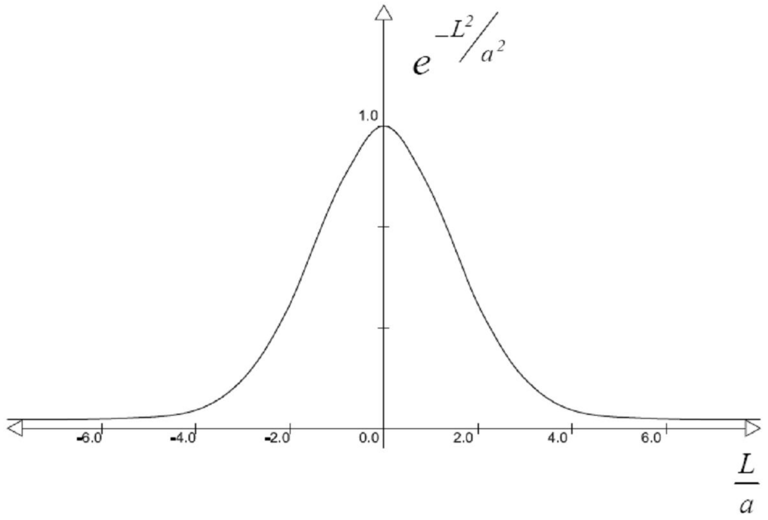


Figure 5. 18: Normal distribution

The position of a person at each time step is calculated as the sum of the velocity that drives the person towards destination plus the velocity due to the virtual social repulsive force. The simple movement that is influenced by repulsive effect has been written in *CalculateMotion()* function as follows:

```
for(Individual=0;Individual<=LastIndividual-1;Individual++)
{
   for(otherIndividual=Individual+1;otherIndividual<=LastIndividual;otherIndividual++)
   {
      for(xyz=0;xyz<=1;xyz++)
      DeltaCoord[xyz]=Coord[0][xyz][Individual]-Coord[0][xyz][otherIndividual];
      SeparationSq=DeltaCoord[0]*DeltaCoord[0]+DeltaCoord[1]*DeltaCoord[1];
      ForceMagnitude=RepulsionOverMassFactor*exp(- SeparationSq/RepulsionDecayLengthSq);
      for(xyz=0;xyz<=1;xyz++)
      {
         RepulsiveForce[xyz][        Individual]+=ForceMagnitude*DeltaCoord[xyz]/sqrt(SeparationSq);
         RepulsiveForce[xyz][otherIndividual]-=ForceMagnitude*DeltaCoord[xyz]/sqrt(SeparationSq);
      }
   }
}
for(Individual=0;Individual<=LastIndividual;Individual++)
{
   for(xyz=0;xyz<=1;xyz++)
   {
      for(TimeBeforePresent=AbsoluteLastTime;TimeBeforePresent>=1;TimeBeforePresent-=1)
      {
         Coord[TimeBeforePresent][xyz][Individual]=Coord[TimeBeforePresent-1][xyz][Individual];
      }
      RepulsiveVely[xyz][Individual]=deltat*RepulsiveForce[xyz][Individual]/mass;
      Coord[0][xyz][Individual]=Coord[0][xyz][Individual]+deltat*Vely[xyz][Individual]+
                                    deltat*RepulsiveVely[xyz][Individual];
      if(Coord[0][xyz][Individual]>+Boundary[xyz])
      {
         Coord[0][xyz][Individual]=-Boundary[xyz];
      }
      if(Coord[0][xyz][Individual]<-Boundary[xyz])
      {
         Coord[0][xyz][Individual]=+Boundary[xyz];
```

```
        }
    }
}
```

In the line

```
RepulsiveForce[xyz][       Individual]+=ForceMagnitude*DeltaCoord[xyz]/sqrt(SeparationSq);
```

the quantity `DeltaCoord[xyz]/sqrt(SeparationSq)` is used to resolve the force into the direction xyz = 0 for the *x* direction and xyz = 1 for the *y* direction.

Figure 5.19 shows patterns generated by two intersecting groups of people who are moving in different directions. The individuals are initially distributed randomly covering the whole area of the 2D virtual environment. Their movement behaviour is mainly to move toward the given destination and to avoid collision with other people.

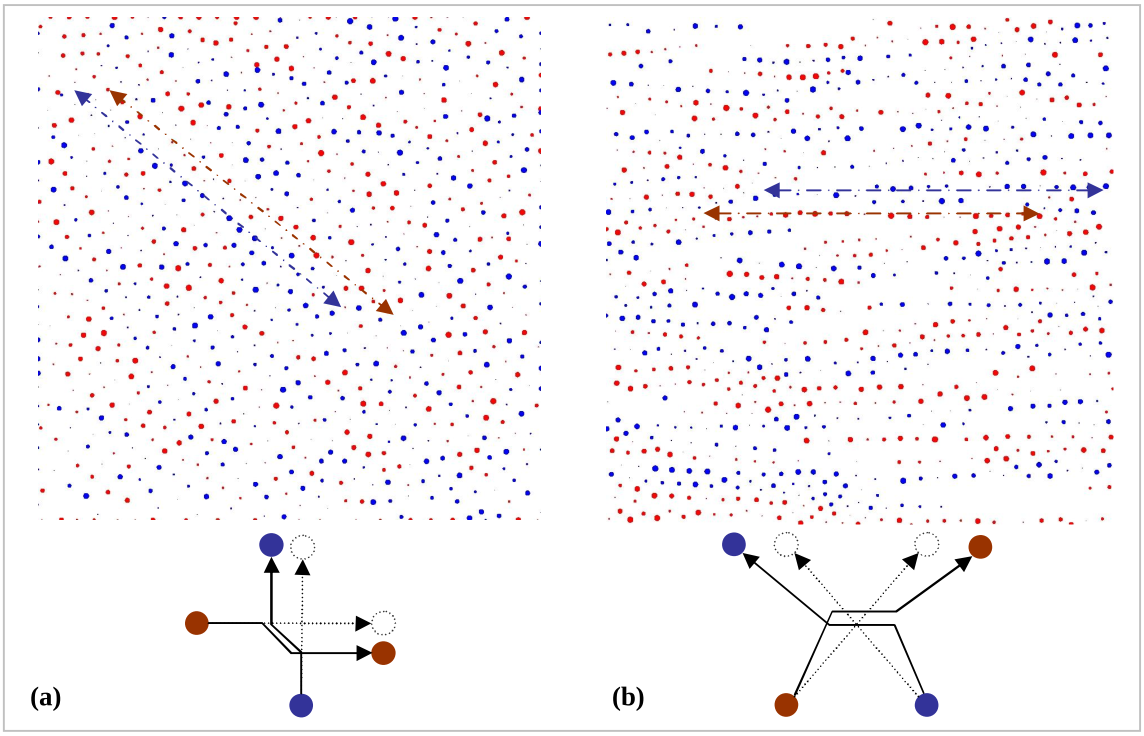


Figure 5. 19: Pattern generated by intersecting groups of people in different directions

In figure 5.19a the blue people are trying to move upwards and the red people to the right creating emergent diagonal stripes. In figure 5.19b the blue people are trying to move from bottom right to top left and the red from bottom left to top right. This time the stripes are horizontal.

Figure 5.20 is taken from Helbing et al. [2005] in which they used the Social Force model to investigate two intersecting pedestrian streams. Again a banding effect or stripe formation can be observed [for details, see Helbing et al., 2005].

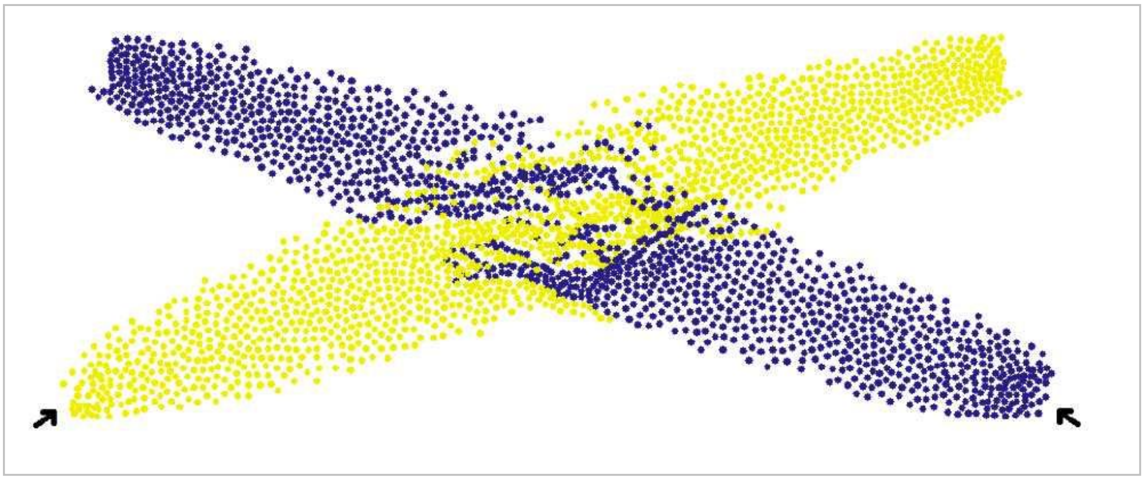


Figure 5. 20: Representative simulation results of two intersecting pedestrian streams using the social force model [Helbing et al., 2005, p. 12]

Stripe formation is a self-organized phenomenon, as it has not been programmed from the beginning but emerges out spontaneously from the process itself. Empirical data show that stripes are formed with real pedestrians. When walking through a crowd it is natural to follow somebody going in the same direction.

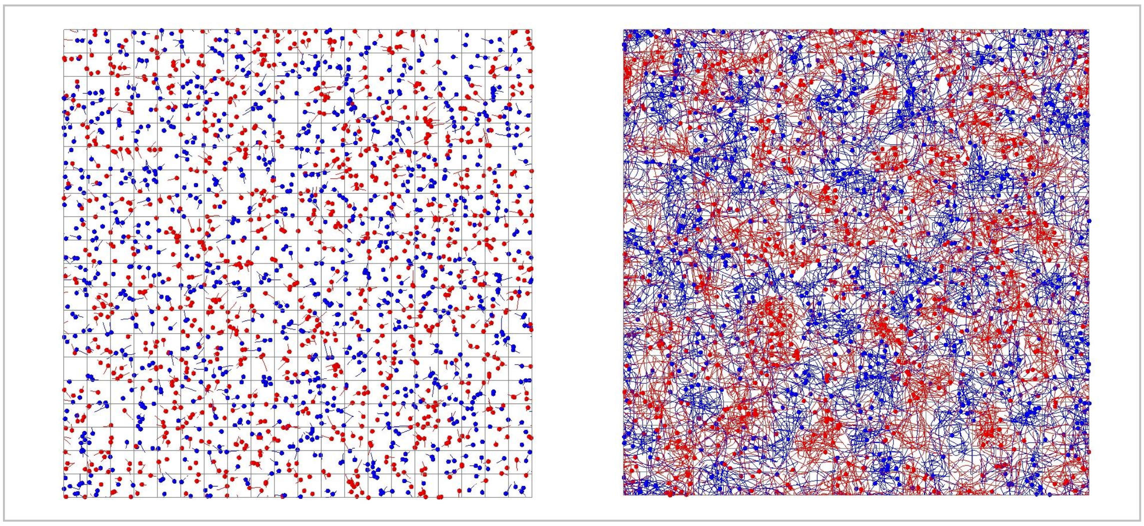


Figure 5. 21: Emergent pattern generated by social repulsive forces

An attempt to investigate different design possibilities given by the interaction between individuals is shown in figure 5.21. In this case the red individuals feel a greater repulsive force from the blue individuals than they do from their fellow reds and *vice versa*. As might be expected the initial random pattern in the left hand diagram is gives way to the clumping shown in the right hand diagram.

In the social force model it is assumed that the social repulsive is sufficient to stop people touching. Another possibility is to assume that people act like billiard balls and bounce off each other when they actually touch. In reality both effects occur.

In the analysis of colliding objects it is assumed that when objects bounce off each other their relative radial velocity after impact is ξ times the approach radial velocity where ξ is the coefficient of restitution. If ξ equal to 1.0 the encounter is purely elastic and kinetic energy is conserved. If ξ is less than one kinetic energy is lost and people settle down.

Figure 5.22 shown banding or stripe formation using this rule. Having been formed, the stripes do not dissipate as they would with the social force model where people would move away from each other after crossing the intersecting group.

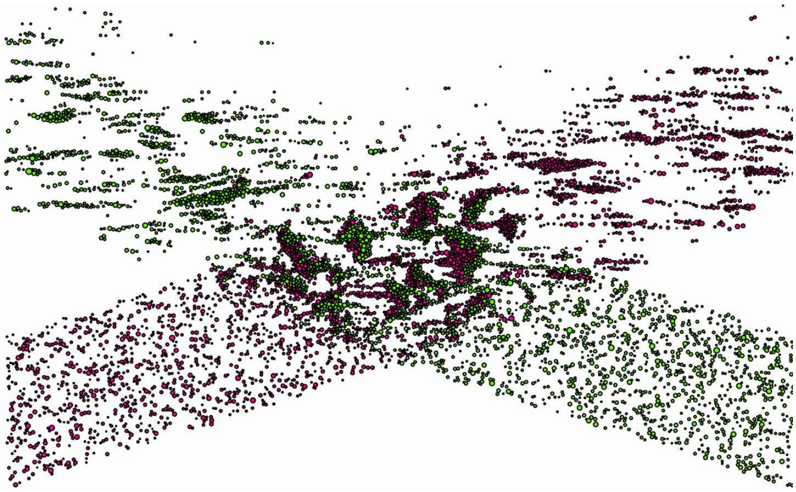


Figure 5. 22: Simulation results of two intersecting people groups applying banding effect [Williams, 2006]

The behaviour of two individuals who bounce when they touch can be described by simple local rules. The banding phenomenon at a global level is not the result of any overall intelligence but it is generated due to the random variations.

(a) Small drag pulling **(b) Large drag pulling**

Figure 5. 23: Banding patterns generated by changing parameters

As well as the bouncing, the people in figure 5.23 are dragged along by a wish to move towards their destination. This drag force is proportional to the difference between their desired velocity and their actual velocity at a given instant of time. A small value of the drag is shown in figure 5.23a and a larger value is shown in figure 5.23b. The larger the desire or the drag the more people will push other people out of their way, generating stronger bands.

5.6.2.2 Obstacle avoidance

People are subjected to virtual repulsive forces not only by other people (social repulsive force) but also by inanimate objects such as structural elements (columns) and furniture.

The obstacle avoidance effect is modelled based on the same principles that have been applied in the repulsive effect between people. However, the rules are different, in walking past a column, one does not worry about invading the column's personal space and also the column is unlikely to undergo some unexpected movement. On the other hand the column is not going to 'give way' and there is no alternative but to walk round it as shown in figure 5.24.

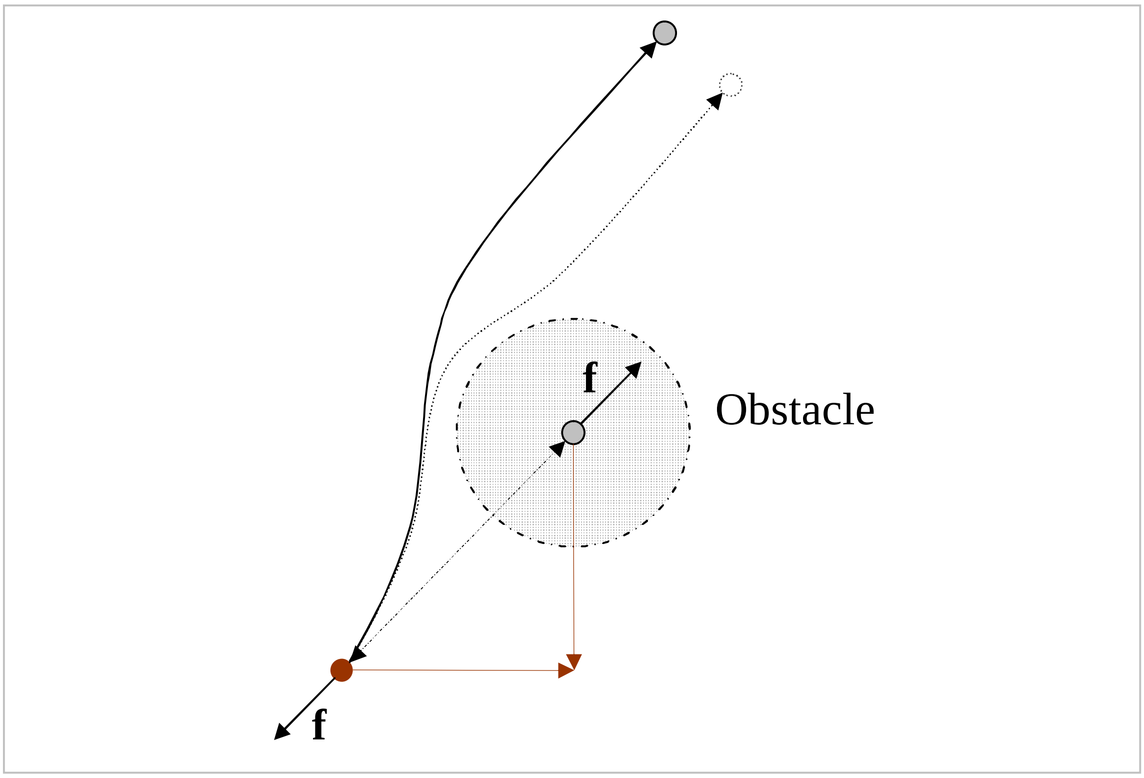


Figure 5. 24: Virtual obstacle repulsive force

After having passed the obstacle, one has the choice of either continuing parallel to one's initial course or returning near to the line of one's initial path (the dotted line in figure 5.24). This latter behaviour is more complex, it requires a repulsive force, then an attractive force and finally a repulsive force.

The simplest technique is to again assume that the force is proportional to the exponential function $e^{-L^2/a^2_{\text{obstacle repel}}}$ where L is the distance between the individual and the obstacle and $a_{\text{obstacle repel}}$ is constant length which may differ from $a_{\text{interperson repel}}$.

Obstacle avoidance is simpler than person avoidance in that the obstacle does not move. This makes it obvious that if one were walking directly towards a column, one would never get round it because all the force would do is try and send you back in the direction from which you have come. In fact what you want is a sideways force to send you one way or the other. In practice people do not get stuck in this way in simulations because there is always a slight asymmetry causing one to go one way or the other.

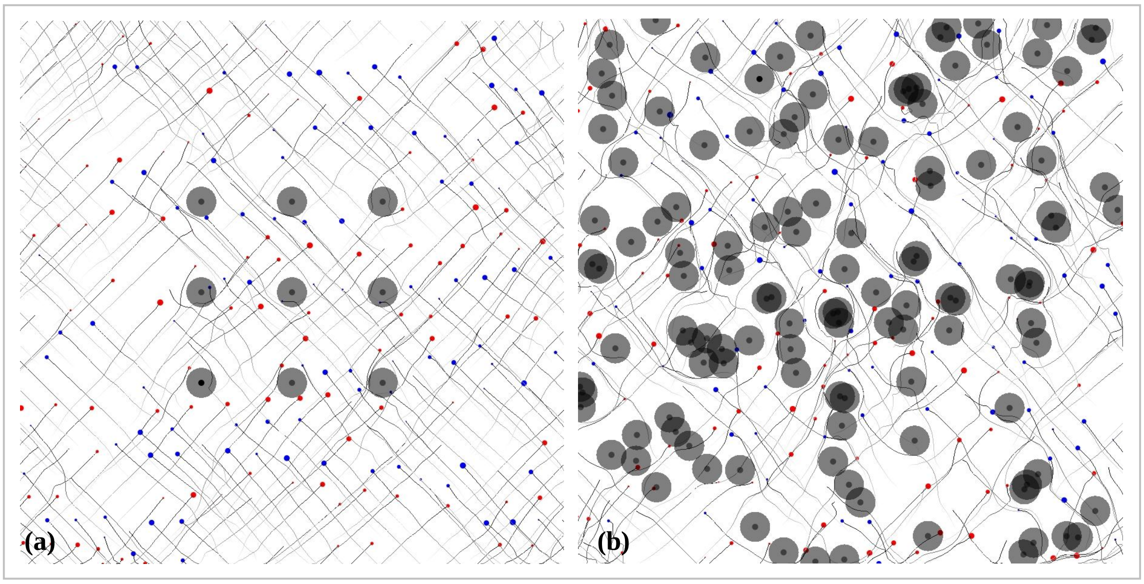


Figure 5. 25: Patterns produced by obstacle avoidance effect

Figure 5.25 shows two emergent patterns produced by the application of the obstacle avoidance effect. In this case, the obstacles are represented as static individuals. Basically, three types of individuals are created, the two intersecting red and blue groups of individuals that represent people and the static individuals that represent the obstacles.

Figure 5.25a shows the obstacles arranged in a grid and figure 5.25b shows the obstacles positioned using a random number generator.

5.6.2.3 Boundaries avoidance

The boundaries can be any large object, in the interior or exterior of the building environment. Boundaries can be distinguished into different types. One possible type is window boundaries, which can motivate either attraction or repulsion behaviour.

The current section examines the boundary avoidance phenomenon that may occur due to the repulsion between people and boundaries. If the distance between a person and any point on the boundary is less than a given length then virtual repulsive force occurs which can be calculated using different approaches:

i. By using the same type of individual for the people and for the points on the boundary. In this case, the parameters that influence their repulsive force like repulsive distance and force magnitude will be the same as well.

ii. By creating a new type of individual for the points on the boundary and by calculating a new virtual repulsive force. Such approach is closer to a real situation since the reaction of a person in case of boundary avoidance is different than his or her reaction in the case of people avoidance.

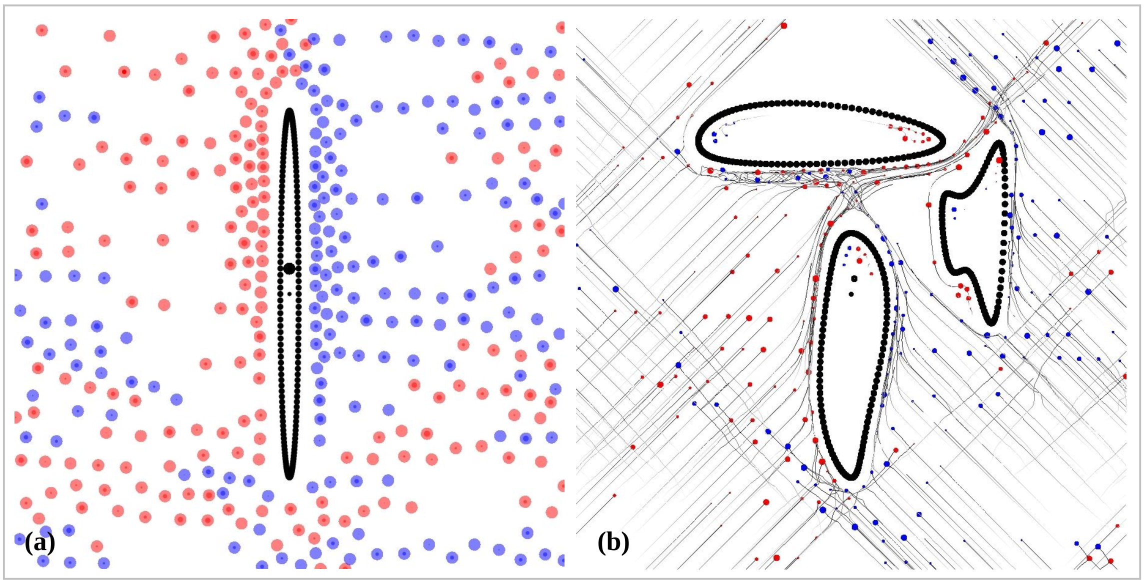


Figure 5. 26: Patterns produced by the boundaries avoidance effect

Figure 5.26 shows two patterns produced by the boundary avoidance effect. In this case, a simple model of intersecting groups of people (red and blue) is examined in relation to their behaviour in the presence of boundaries of different shapes.

Virtual boundaries can be geometrically defined in different ways:

i. By defining various shapes (circles, ellipses, etc) and by replacing their points with static individuals.

ii. By defining a system of lines in .txt files. Then, lines can be divided into a number of points, which can be the positions of static individuals. This method can be applied for the analysis of existing buildings.

iii. By positioning group of obstacles in real time using the computer mouse. This approach is easier since it allows the direct poisoning and shaping of boundaries while the program runs.

Any modification of the parameters that influence boundary avoidance behaviour may produce different results. Thus, the user's interaction with the program is very important. The users can apply their own rules and parameters, and investigate different design possibilities.

5.6.2.4 Attractive effect

People movement is also influenced by a virtual attractive force. The attractive effect can be applied in different cases.

i. First, it can be used to investigate the behaviour of people who move together or keep a certain distance from others without losing contact.

ii. Second, it can be used in cases where people are attracted by different elements in the environment, for instance information points, chairs, windows, and so on.

Virtual attractive force is similar to virtual repulsive force as explained in section 5.2, although it has negative value. Again, the force magnitude is influenced by the relationship between force-distance.

In this case, it is assumed that members of a group of people like families and friends are attracted because they tend to walk close together. In such a case, people are attracted at a certain distance, though if this distance is less than a given length, they repel each other. This is because individuals tend to be close together but they also try to keep a distance from each other and they cannot physically occupy the same space. If the distance between the individuals is increased, the attractive force becomes small and the attraction occurs only between the people who are close together. In this way, small groups of individuals that move together are generated.

The virtual force magnitude is assumed to be equal to

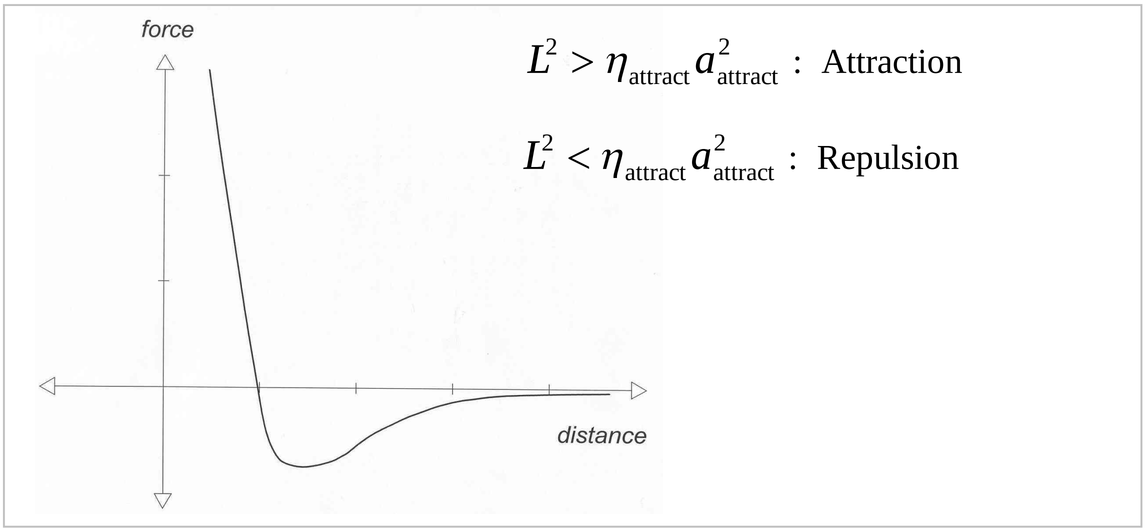
$$C_{\text{attract}}\left(1-\frac{\eta_{\text{attract}}\,a_{\text{attract}}^{2}}{L^{2}}\right)e^{-\frac{L^{2}}{a_{\text{attract}}^{2}}}$$

where C_{attract} is a constant with the units of (virtual) force L is the distance between individuals, $a_{\text{interperson attract}}$ is a constant length are η_{attract} is a constant non-dimensional number. The values of C_{attract}, a_{attract} and η_{attract} apply between two particular individuals. Different constant values may apply between different individuals.

If $L^2 > \eta_{\text{attract}} a^2_{\text{attract}}$ then the individual will attract each other.

If $L^2 < \eta_{\text{attract}} a^2_{\text{attract}}$ then the individuals will repel each other.

This force - distance relationship is shown in figure 5.27.



$L^2 > \eta_{\text{attract}} a^2_{\text{attract}}$: Attraction

$L^2 < \eta_{\text{attract}} a^2_{\text{attract}}$: Repulsion

Figure 5. 27: Force distribution in relation to the distance L. The crossing of the line by axis $x = 0$ shows the distance L between two individuals where their behavior changes from repulsion to attraction

Again there is nothing ‘magical’ about this relationship and other relationships could be tried. The dying off of the attraction with distance is not because the individuals no longer want to be together, but because each no longer knows where the other is.

Figure 5.28 shows two patterns produced by the attractive effect. In the first simulation (see figure 5.28a), the red and blue groups are distributed randomly but the red group all have the same initial velocity and the blue group all have the same initial velocity, different from that of the red group. The virtual attractive force is applied between any individual and it does not make distinction between individuals from the two groups.

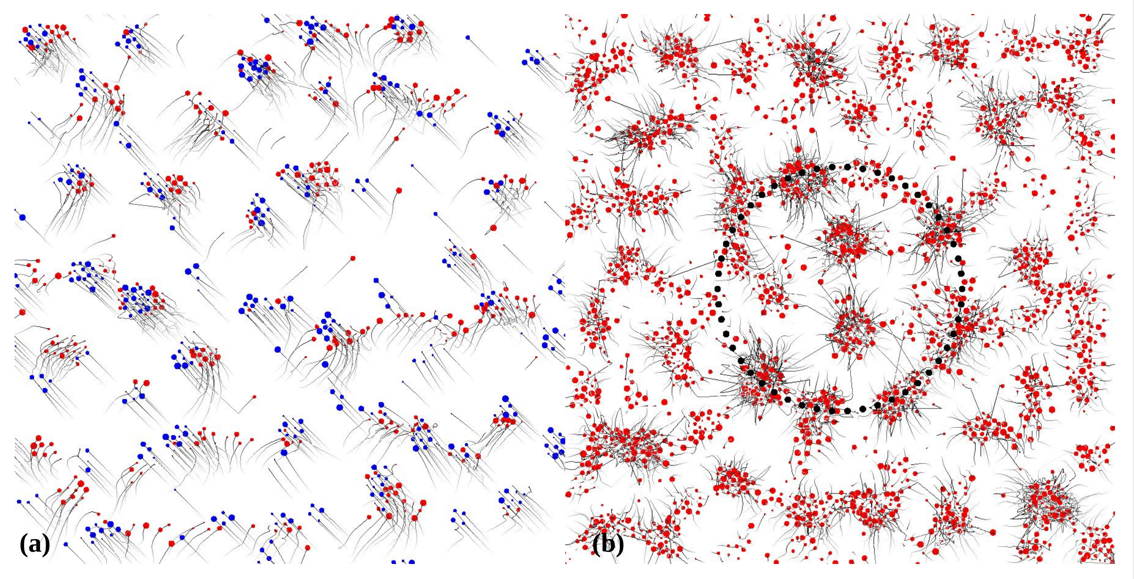


Figure 5. 28: Patterns produced by virtual attractive force

Figure 5.28b shows a pattern produced by a different initial data configuration. In this case, red individuals are distributed randomly in the virtual environment with zero initial velocity. As one might expect, the individuals form clumps although some individuals are left unattached. The fixed black dots also attract the individuals. They could, for example, represent art works in a gallery.

Modification of the values C_{attract}, a_{attract} and η_{attract} can produce structured patterns such as those in figure 5.29. In this case just one clump appears, and the repulsion between individuals produces a roughly constant spacing.

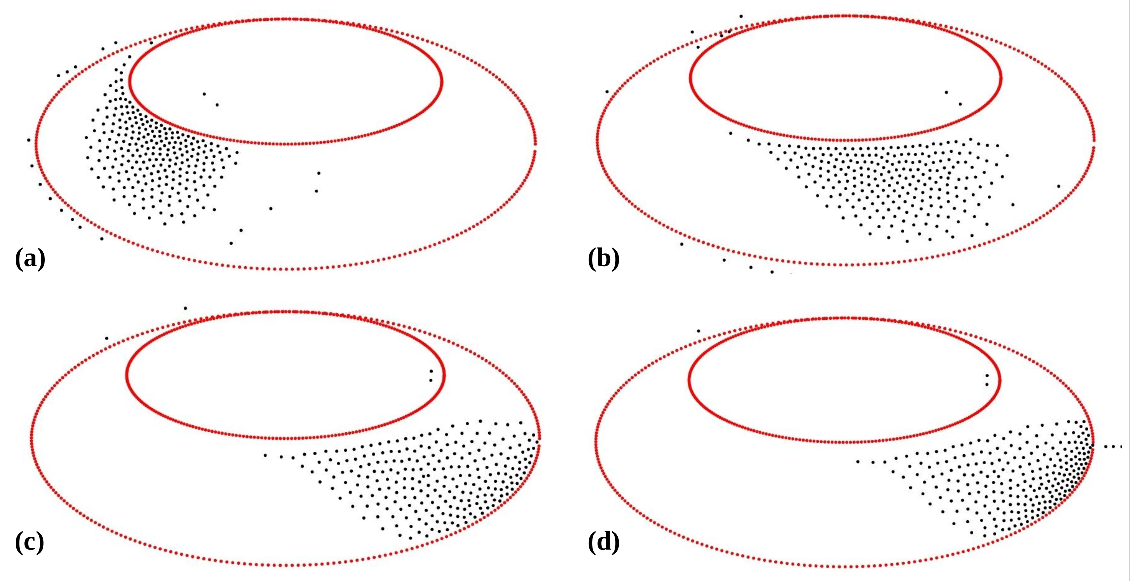


Figure 5. 29: Clump of individuals generated by attractive affect

5.6.2.5 String affect

The string effect is another numerical implementation of individual behaviour that is based on virtual attractive forces. This technique was developed to simulate Frei Otto's [1995] physical models for the form-finding of optimal path systems in which wool threads were dipped into water and pulled together by surface tension (see section 4.3.3).

The wool threads are represented by strings that are initially arranged as random straight lines. Each string consists of a number of individuals that are distributed at a uniform spacing as shown in figure 5.30.

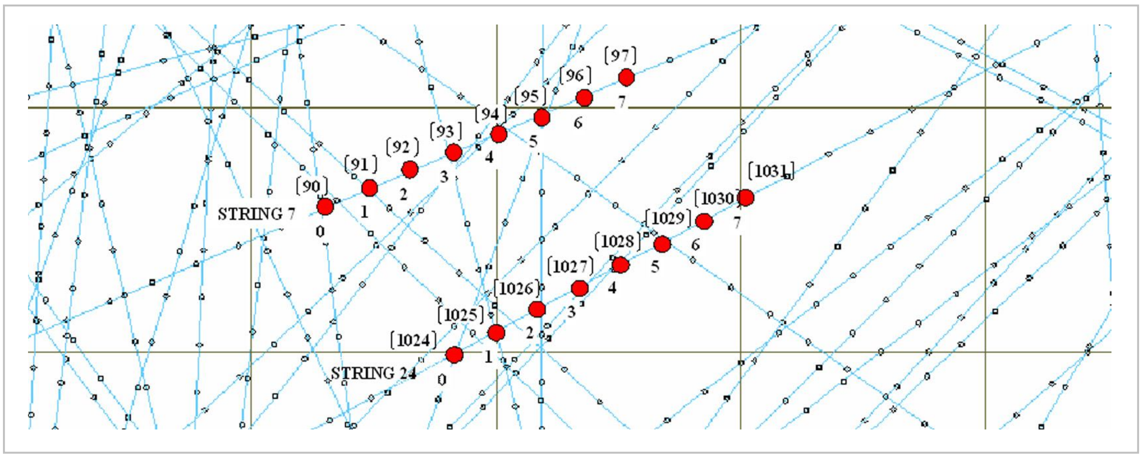


Figure 5. 30: System of strings (blue) and individuals (red)

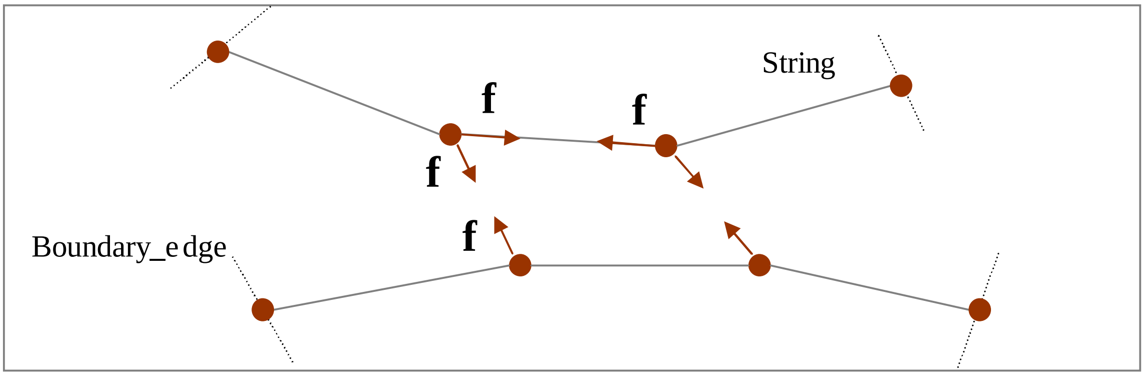


Figure 5. 31: Virtual attractive forces

Figure 5.31 shows the virtual attractive forces that motivate individual behavior. Individuals on the boundaries remain fixed. The individuals on the same string attract each other due to the tension in the string. The individuals from different strings attract each other by surface tension if their distance is less than a given length R. The computer code with analytical explanations is given in Appendix C.

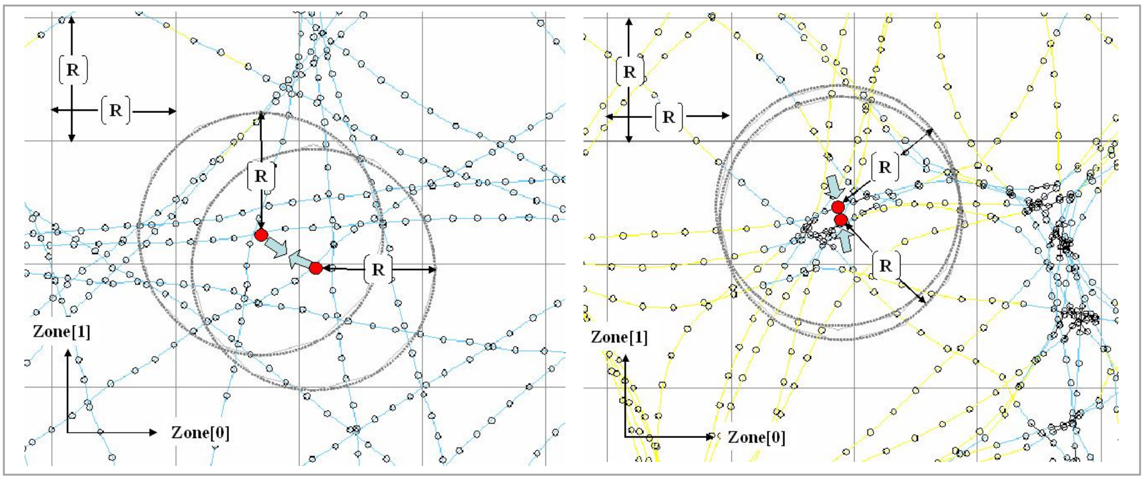


Figure 5. 32: Individuals attract each other if their distance is less than a given length

Figure 5.32 shows two time steps of the attraction behaviour. Gradually the attraction between individuals results in the generation of a network of single strings coalesced strings, holes, and empty areas.

Figure 5.33 shows four steps of form-generation. The string ends are fixed to the outer circle but the strings are repelled by the inner circle which might represent some fixed obstacles.

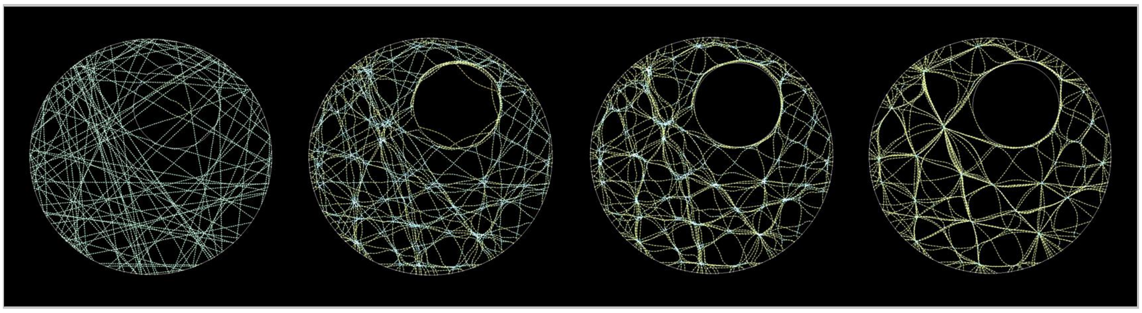


Figure 5. 33: Four basic steps of computer-generated path systems

5.6.3 The sign effect

Thus far, it has been assumed that individuals or groups of people were moving in simple predetermined directions, either in an open space or restrained by walls. Research has been done in the area of pedestrian dynamics such as simulation of emergency evacuation does not consider navigation of individuals as a special problem.

Computer simulation of vision and way-finding is much more difficult. Vision involves ‘seeing’ the object with an eye or a camera and then interpreting what the object is. The interpretation is very difficult involving Signal Processing, Artificial Intelligence, and mathematics in an attempt to simulate neurobiology. Similarly for a computer program to have any understanding of different people's needs in terms of way-finding is almost impossible.

We have already discussed the work of the Space Syntax group in this area (section 4.4.8.3). Here we will introduce a simple concept the *sign effect* which can be used in conjunction with any agent based or 'social force' model.

Signs are necessary in public buildings when it is not obvious which way to go to get to one's desired objective. In a sense the need for a sign is an admission of failure in that other more subtle architectural devices should draw the visitor towards his or her destination. However in a complex building such as a hospital, there is no alternative but to use signs to help people navigate. Airport and ferry terminal buildings are simpler than hospitals, with a well defined sequence of movement:

Entrance check-in, security, departure lounge, passport control, gate, aeroplane

or

Aeroplane passport control, baggage claim, customs, leaving.

Thus far we have had 'agents' or 'people' acted upon by other 'people' as well as other agents representing boundaries or art works. Here we will introduce a new type of agent or entity, the *sign*. Thus far our agents have not been orientated, except by their direction of travel. However a sign will usually be stationary and point in a particular direction.

We will use the word sign to also include any visual clue that directs people towards a particular goal. Thus, if at a ferry terminal the departing passengers can see the ships, they will be drawn towards them and signs are only necessary to make sure they get on the right one.

5.6.3.1 Sign effect definition

The sign effect geometry is based on the fact that if one sees a sign in the distance pointing to the right towards one's desired goal, one has a tendency to two things:

i. Move towards the right
ii. Move towards the sign

The main function of sign is to allow the individuals to be attracted towards an 'active line', and then directed by this active line to the next sign. The direction of the 'active line' is defined by an angle theta and in turn the angle specifies the direction of individual movement.

In order to explain the geometrical relation between an individual and a sign, an analytical model is introduced. The model describes the position of an individual (red circle) in relation to the position of a sign (grey circle) in space. Figure 5.34 shows the individual-sign geometry:

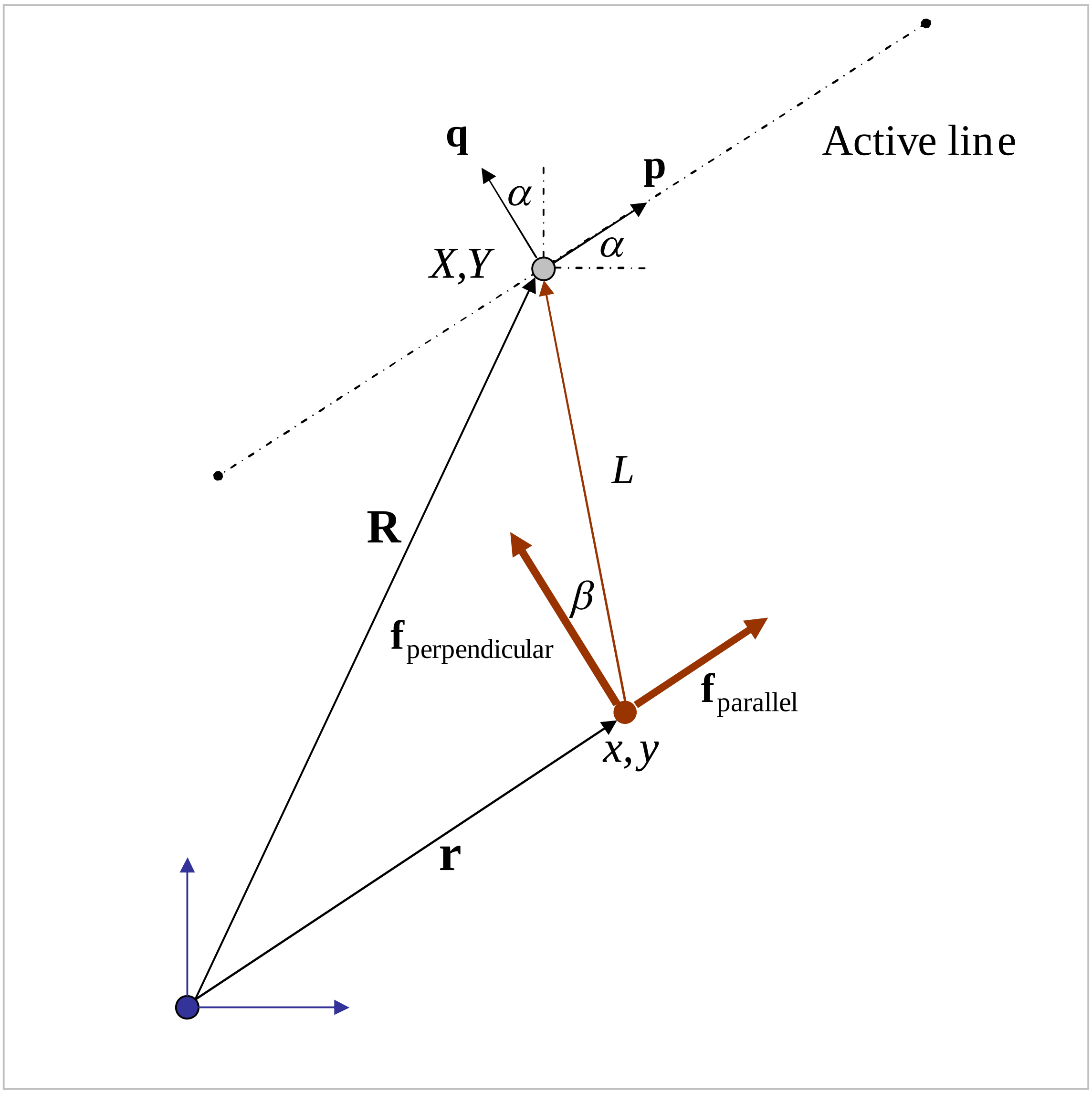


Figure 5. 34: Individual-sign geometry. The red circle represents the individual and the grey circle the sign

The individual and the sign position are defined by the vectors $\mathbf{r}$ and $\mathbf{R}$ respectively. The vector from $\mathbf{r}$ to $\mathbf{R}$ is described as follows:

$$\mathbf{R} - \mathbf{r} = (X - x)\mathbf{i} + (Y - y)\mathbf{j}$$

The sign is pointing in the direction of the unit vector $\mathbf{p}$ or the 'active line' of sign. Perpendicular to the vector $\mathbf{p}$ is the vector $\mathbf{q}$. Both vectors are unit vectors and their Cartesian form is given as follows:

$$\mathbf{p} = p_x\mathbf{i} + p_y\mathbf{j} = \cos\alpha\mathbf{i} + \sin\alpha\mathbf{j}$$

and

$$\mathbf{q} = q_x\mathbf{i} + q_y\mathbf{j} = -p_y\mathbf{i} + p_x\mathbf{j} = -\sin\alpha\mathbf{i} + \cos\alpha\mathbf{j}$$

The scalar product $\mathbf{q}\bullet(\mathbf{R}-\mathbf{r})$ is the projector of vector $\mathbf{R}-\mathbf{r}$ onto the unit vector $\mathbf{q}$ and is equal to:

$$q_x(X-x) + q_y(Y-y) = L\cos\beta$$

where L is the distance of the person from the sign, $L = |\mathbf{R}-\mathbf{r}| = \sqrt{(X-x)^2 + (Y-y)^2}$.

The forces on the person are now written

$$\mathbf{f}_{\text{parallel}} = C_{\text{parallel}} e^{-\frac{L^2}{a_{\text{sign}}^2}} \mathbf{p}$$

and

$$\mathbf{f}_{\text{perpendicular}} = C_{\text{perpendicular}}\, \mathbf{q}\bullet(\mathbf{R}-\mathbf{r})\, e^{-\frac{L^2}{a_{\text{sign}}^2}} \mathbf{q}$$

in which C_{parallel} and $C_{\text{perpendicular}}$ are constants with the units of virtual force which may vary from person to person – only some people will be motivated to follow the sign. a_{sign} is a constant with the unit of length, which again may vary from person to person and sign to sign. Again the use of the bell-shaped curve, $e^{-\frac{L^2}{a_{\text{sign}}^2}}$ is arbitrary. The use of the scalar product $\mathbf{q}\bullet(\mathbf{R}-\mathbf{r})$ ensures that $\mathbf{f}_{\text{perpendicular}}$ is always directed towards the Active Line. Of course many signs are one-sided in which one would set $\mathbf{f}_{\text{perpendicular}}$ equal to zero if $\mathbf{q}\bullet(\mathbf{R}-\mathbf{r})$ did not have the appropriate sign (plus or minus).

5.6.3.2 Simple examples

Figures 5.35a, b and c show a number of numerical experiments. Figure 5.35a has a single sign. Figure 5.35b has two signs and the constant $C = \dfrac{C_{\text{parallel}}}{C_{\text{perpendicular}}}$ is varied. Figure 5.35c has two signs and the position of the second sign is varied.

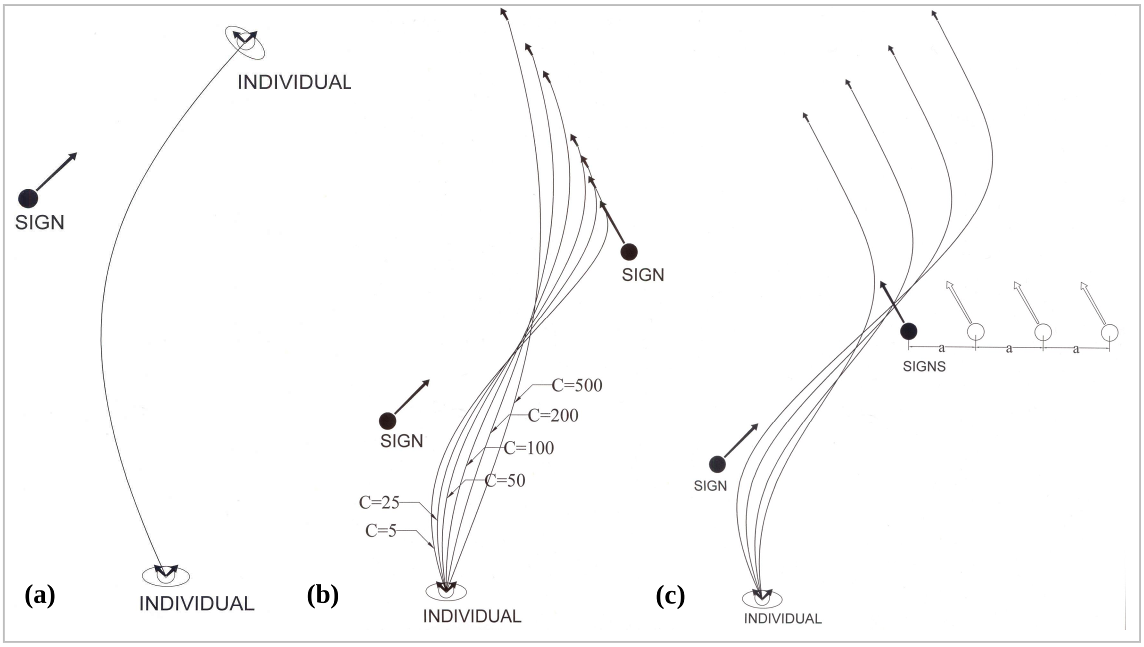


Figure 5. 35: a. Path curve line generated by sign effect, b. Different path curves applying C_{parallel} values, c. Different path curve lines changing sign position

5.6.3.3 Vortex motion

Figures 5.36 and 5.37 show more complex motion in which vortex motion sometimes occurs. This is where an individual ends up going round in ever decreasing circles.

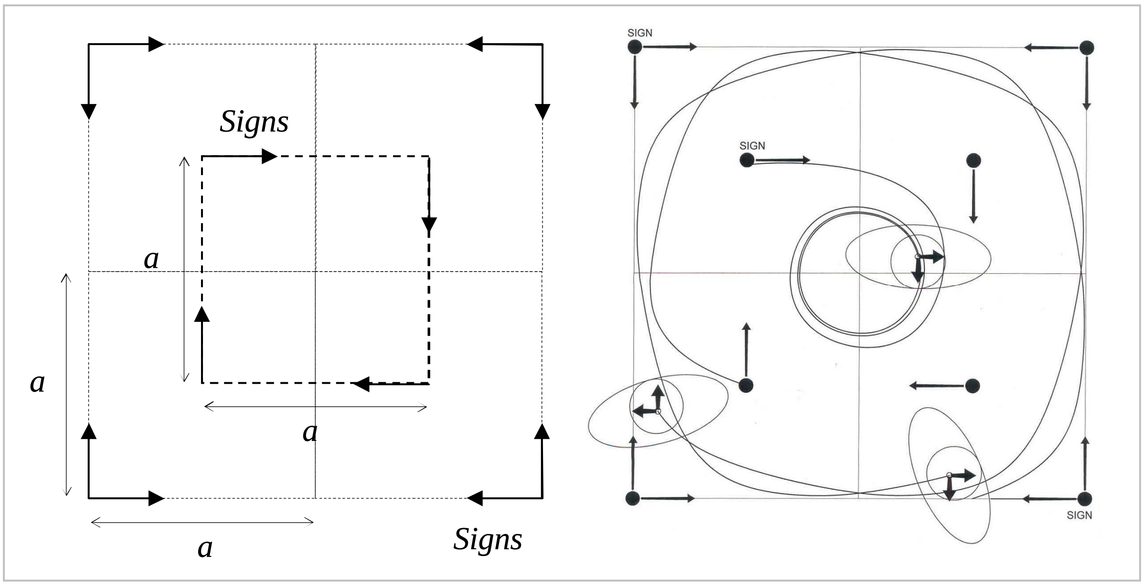


Figure 5. 36: Possible sign configuration and circulation diagrams

Figure 5.37c is especially interesting because the vortex motion occurs with just two signs. This problem could occur in a building with two sets of toilets and a person follows the sign to one set and then the sign to the other and so *ad finitum.*

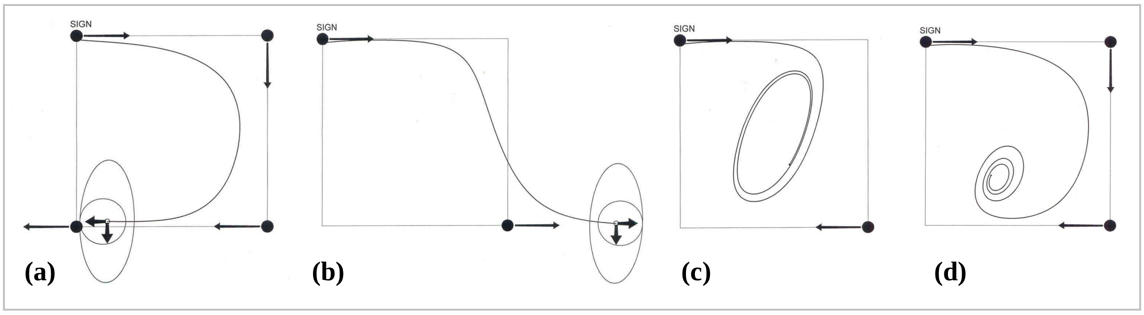


Figure 5. 37: Possibly arrangement of signs and different patterns

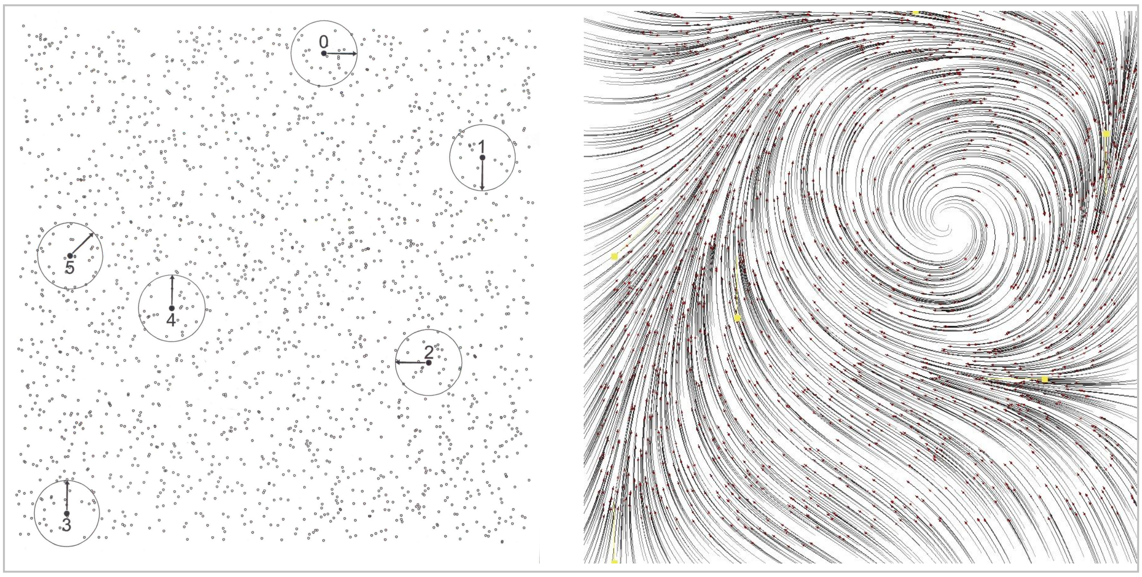


Figure 5. 38: Vortex motion

Figure 5.38 shows vortex motion in which people already in the vortex are joined by more people from outside the vortex. Figure 5.39 shows two groups of people, the red and the blue following two groups of signs which are intended to keep them to the red path or the blue path.

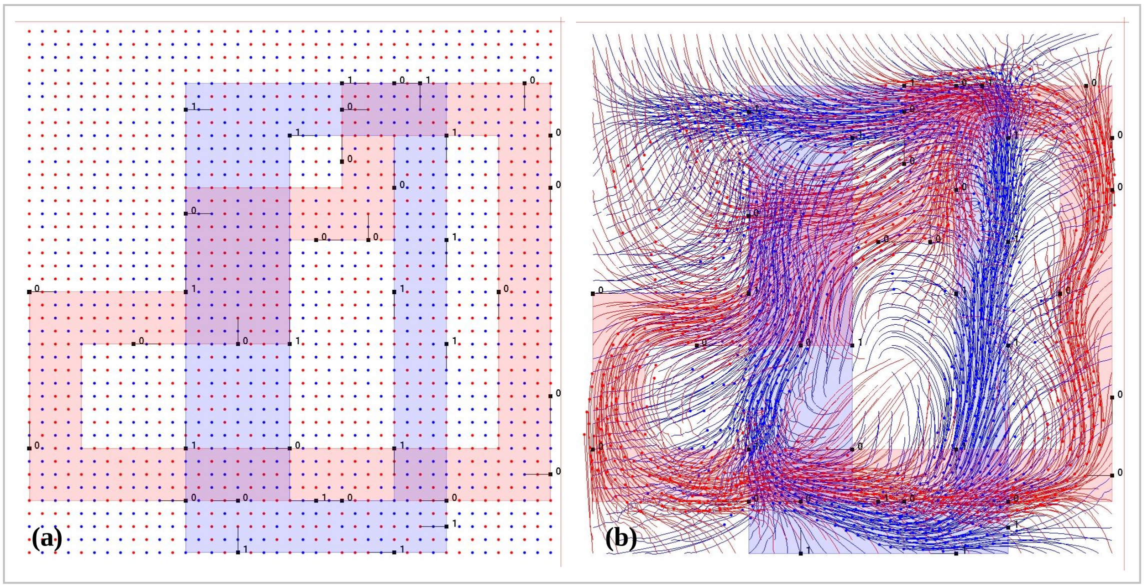


Figure 5. 39: Two groups of people following different signs

5.6.4 Resultant force

As in all agent based on Social Force [Helbing et al., 2005] models, the effect of different forces can be added together vectorially, that is the adding the *x* and *y* components of force separately.

The example in figure 5.40 shows an ‘Individual 0’ who is subjected to virtual social repulsive forces from other individuals and the virtual attraction-direction force from a sign.

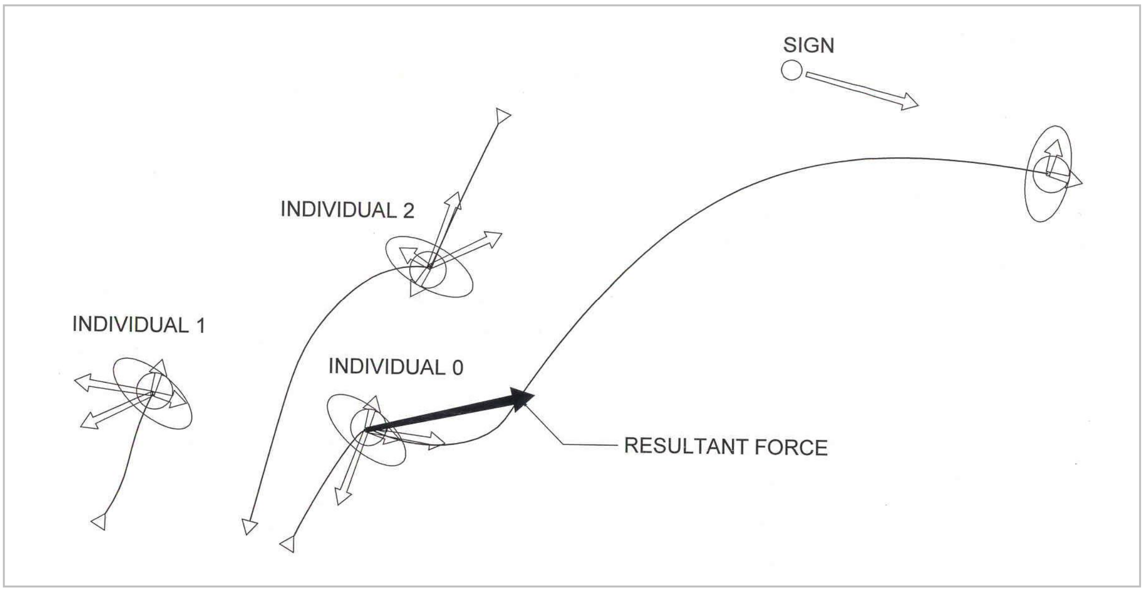


Figure 5. 40: Resultant force

5.6.5 Route choice behavioural rules

According to Arthur and Passini [1992], there are three factors influencing layout design at the conceptual stage:

i. Linkage and organization of space

ii. The identification of the constituent spatial units

iii. The grouping of spatial units into destination zones

The organization and linkage of units and zones refers mainly to the design of circulation systems. Their function is to link the spatial units that have been identified as having functional relationships. Arthur and Passini [1992] wrote:

> ‘Design process can take off from the circulation system and end up with an overall form, or it can start with the form and end up with the circulation system. Circulation is the key organizing force of a layout. If circulation system is understood, then spatial organization of building may be understood as well’

Arthur and Passini [1992] see *wayfinding* as a spatial problem-solving process. The process includes three interrelated parts. First is the decision making (plan action), second is the decision executing (take action), and finally, the information processing. The decision-making or plan action is basically a cognitive map that is a tree of action that follows a sequence of steps.

The decision-making process can be described as a hierarchical structure. In order for an individual to reach a destination, a group of additional decisions need to be made. If these decisions comprise a plan of action, then first the decision plan is structured and second it is hierarchical. Figure 5.41 shows a diagram of a hierarchical structured decision plan from Arthur and Passini [1992].

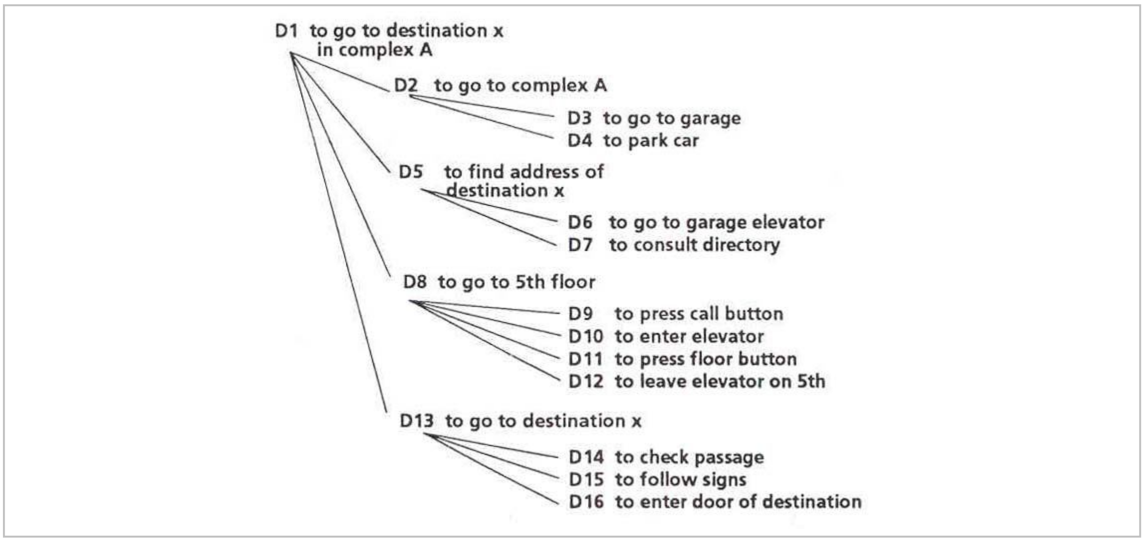


Figure 5. 41: Diagram of a hierarchically structured decision plan [Arthur and Passini, 1992]

Hoogendoorn et al. [2001] describe *route choice behaviour* as a problem of finding the ‘best’ path way towards a destination using the method of origin-destination and cost-based models of human-pedestrian behaviour.

A similar cognitive map is used in the current research by translating the tree of different actions into individual movement behaviour using signs and destinations which may lead to the generation of complete circulation patterns.

5.6.5.1 Categories of destinations

A destination is the goal of a movement action and it can be represented as functional unit, area, object, etc. Each individual follows a ‘list of destinations’ to guide him or her to the current destination point and eventually their final destination.

In an airport terminal a person’s current destination might be the check-in point, followed by passport control and security check followed by various destinations in departure lounge like Bagel Street[14] and toilet followed by his or her final destination, the departure gate. Clearly a person needs to pass through the security check or the departure gate but in the departure lounge he or she can select from a large number of destinations according to his or her personal desire. In order to capture the complexity of such route choice behaviour we categorize destinations as either:

i. Compulsory, or

ii. Optional

Each individual's list of destinations might contain only one or both categories of destinations.

5.6.5.2 An individual's list of destinations

An individual is only interested in signs pointing towards their current destination and will, for the time being, ignore all other signs. Also, because the sign effect dies off with distance, the individual will only be influenced by nearby signs to his or her current destination. It is, of course possible to program any rule provided that there is an appropriate algorithm. Thus one might use a random number generator to say that a certain proportion of people in departure lounge will select to go into the duty free shop on impulse when on their way to the bookshop.

Once an individual has reached his or her current destination he or she will remain at the destination for certain length of time before moving off. This length of time might be predetermined or it might also depend upon a random number. It could also depend upon the number of other people at that particular destination at that time.

Following sections demonstrate a number of numerical experiments.

- **Experiment type 1. Each individual has a compulsory destination**

Figure 5.42 shows individuals following paths towards different destinations. In this case each individual follows signs that lead to a compulsory destination. It would be nice to be able to investigate all the parameters controlling this behaviour, but clearly this is impossible, especially as the different parameters interact. Thus one might say that such a pattern indicates a qualitative picture of how people might behave, rather

---

[14] An excellent chain of eating establishments specialising in bagels. At Heathrow Terminal 5 it is impossible to find Bagel Street without following signs due to its out of the way location (which therefore, presumably, means a lower rent). At the time of writing, soon after the Terminal opened, it is almost exclusively used by airport employees.

than a quantitative picture of how they will behave. Clearly the effort required to reduce the errors in modelling depend upon the consequences.

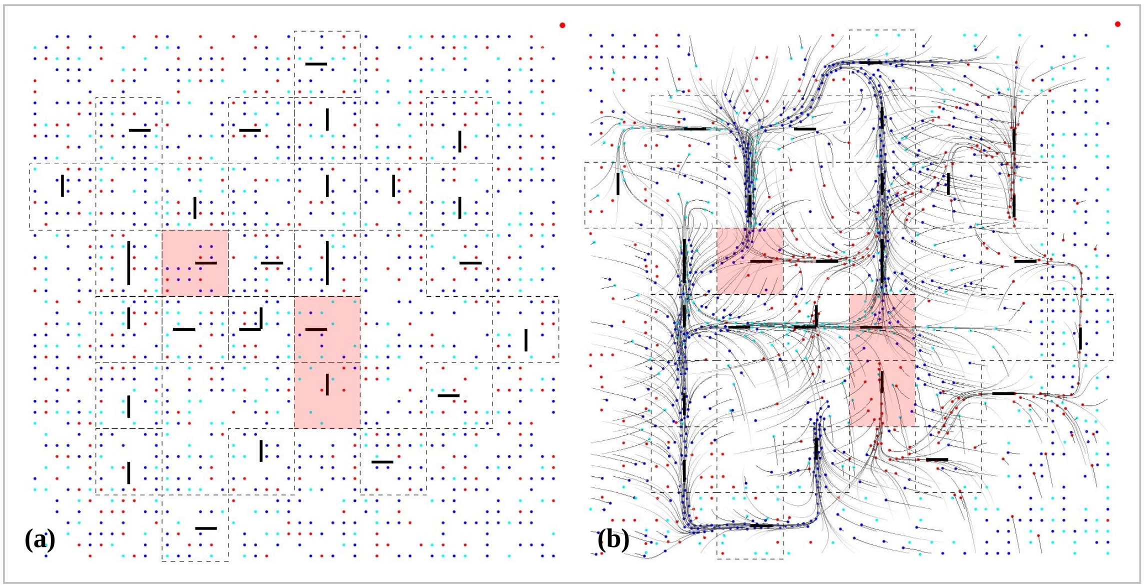


Figure 5. 42: a. Initial configuration of signs and destinations, b. Pattern produced by a compulsory destination for each individual

- **Experiment type 2. An individual's list of optional destinations**

Figure 5.43 shows another numerical experiment where an individual's list contains only optional destinations. Figure 5.43a shows the initial configuration, there are three destinations that are arranged on the perimeter of a circle. The circle is filled with random strings. The nodes on each string define the position of signs that are pointing towards the destinations.

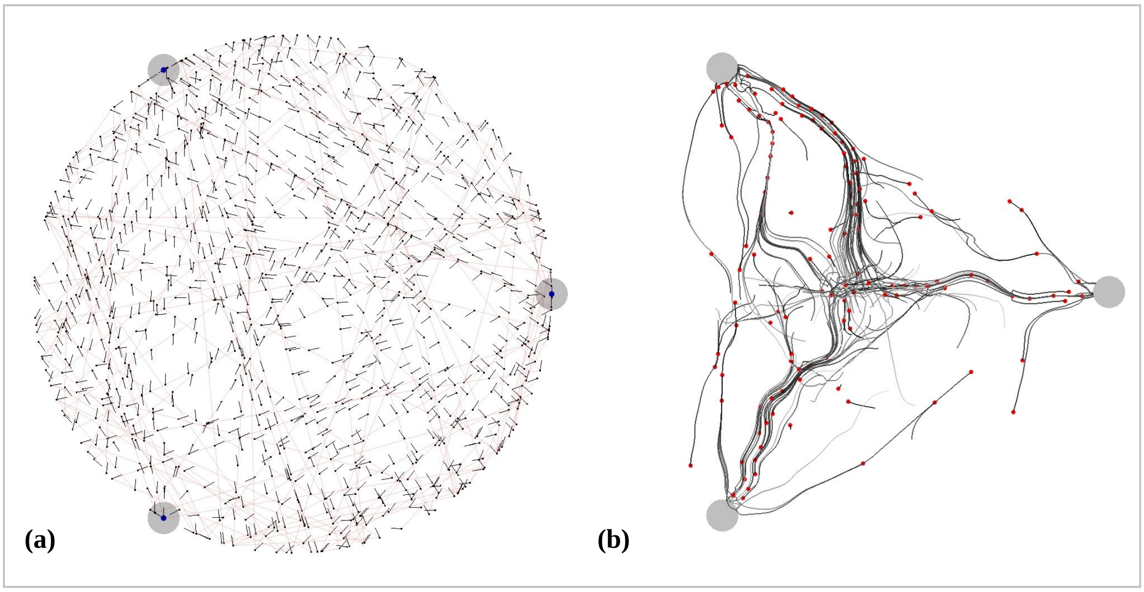


Figure 5. 43: Pattern produced by an individual's list of optional destinations

Individuals tend to move toward their destination following the signs closest to them. Individuals are influenced by all signs of their current destination, but the effect of far away signs is negligible due to the decay of the exponential function. Once the destination is reached, they switch their movement to the next destination according to their list of optional destinations or they are deactivated if their final destination has been reached. Figure 5.43b shows an emergent pattern generated by these rules.

The part of program that generates the list of optional destinations is written in *CreateData()* function and is shown below:

The array *LastWhichHowFar[Individual]* defines the total number of destinations for each individual. Then, array *WhichDestination[Individual][FutureWhichHowFar]* uses a random number generator to define the individual destination in each row of destination list:

```
WhichHowFar[Individual]=0.0;
LastWhichHowFar[Individual]=int((LastDestination*rand())/(1.0*RAND_MAX));
for(FutureWhichHowFar=0;
    FutureWhichHowFar<=LastWhichHowFar[Individual];
    FutureWhichHowFar++)
WhichDestination[Individual][FutureWhichHowFar]=
int((1.0*LastDestination*rand())/(1.0*RAND_MAX)+0.5);
```

The part of program that calculates the individual's selection of next and final destination is written in *CalculateMotion()* function and is given below:

```
for(Individual=0;Individual<=LastIndividual;Individual++)
{
  WhereTheyWantToGo=WhichDestination[Individual][WhichHowFar[Individual]];
  if(Active[Individual]==1)
  {
    CurrentTime[Individual]=ArrivalTime[Individual]+ClockTime-ArrivalTime[Individual];
    TargetDistance[Individual]=(Coord[0][0][Individual]-DestinationCoord[0][WhereTheyWantToGo])*
                               (Coord[0][0][Individual]-DestinationCoord[0][WhereTheyWantToGo])+
                               (Coord[0][1][Individual]-DestinationCoord[1][WhereTheyWantToGo])*
                               (Coord[0][1][Individual]-DestinationCoord[1][WhereTheyWantToGo]);

     if(TargetDistance[Individual]<LSq)
    {
        if(WhichHowFar[Individual]==LastWhichHowFar[Individual])
        Active[Individual]=2;
        else WhichHowFar[Individual]++;
     }
  }
}
```

- **Experiment type 3. An individual's list of compulsory and optional destinations**

Figure 5.44 shows in diagrammatic form the basic idea of an individual's list of compulsory and optional destinations. An individual starts from compulsory destination 0 and then moves to the compulsory destination 1. Then, he or she can randomly select optional destinations from 1 to 4. Once destination 4 is reached, individual switches his movement to the next compulsory destination 5. The number of compulsory and optional destinations for each individual varies according to the case study.

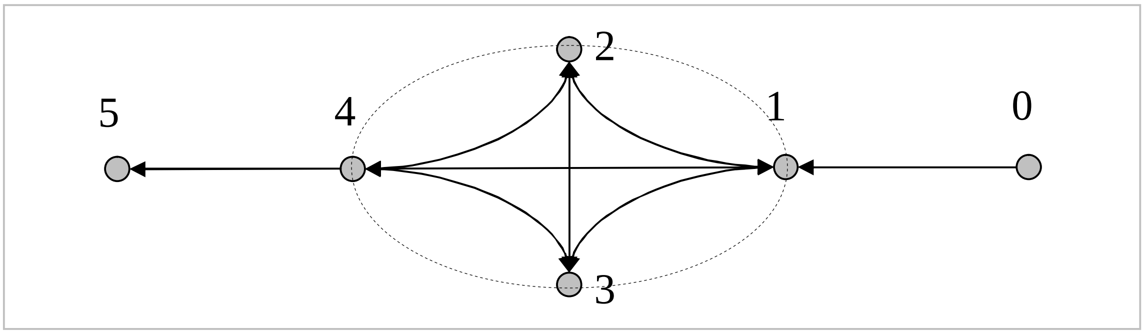


Figure 5. 44: Diagram of compulsory and optional destinations

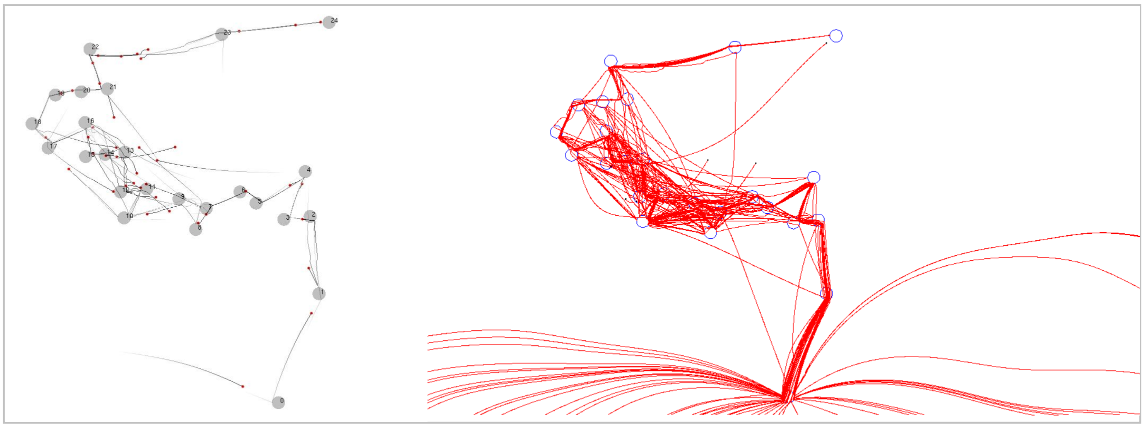


Figure 5. 45: Pattern produced by individual's list of compulsory and optional destinations

Figure 5.45 shows a pattern generated following an individual's list of compulsory and optional destinations. The next destination selection is based on local choice. An example might be where a person has the aim to 'buy a present for my mother' and so he or she has a number of possible shops to visit in a shopping mall.

The computer program consists of two main parts:

i. First, the program defines the compulsory and optional number of destinations for each individual.

The compulsory destinations are defined in *CreateData()* function as below:

```
WhichHowFar[Individual]=0.0;
LastWhichHowFar[Individual]=24.0;
for(FutureWhichHowFar=WhichHowFar[Individual];
    FutureWhichHowFar<=LastWhichHowFar[Individual];
    FutureWhichHowFar++)
WhichDestination[Individual][FutureWhichHowFar]=FutureWhichHowFar;
```

It is assumed that initially all individuals follow a sequence of compulsory destinations from 0 to 24. Two arrays are introduced here to define the individual's list of compulsory destinations:

i. The array *LastWhichHowFar[Individual]* defines the total number of destinations for each individual, which in this case is 24 (all destinations).

ii. Then, the array *WhichDestination[Individual][FutureWhichHowFar]* defines the destination in each row of the list, which is this case follows the sequence 0, 1, 2, …, 24.

The optional destinations are defined in *CreateData()* function as bellow:

```
OptionalWhichHowFar[Individual]=int((12.0*rand())/(1.0*RAND_MAX)+0.5);
OptionalLastWhichHowFar[Individual]=
int(OptionalWhichHowFar[Individual]+(11.0*rand())/(1.0*RAND_MAX)+0.5);
```

The array *OptionalLastWhichHowFar[Individual]* defines the total number of optional destinations and the array *OptionalWhichHowFar[Individual]* defines the first optional destination for each individual using a random number generator.

ii. Second, the program calculates the individual's selection of next and final destination. It is written in *CalculateMotion()* function as below:

```
for(Individual=0;Individual<=LastIndividual;Individual++)
{
   WhereTheyWantToGo=WhichDestination[Individual][WhichHowFar[Individual]];
   if(Active[Individual]==1)
   {
     TargetDistance[Individual]=(Coord[0][0][Individual]-DestinationCoord[0][WhereTheyWantToGo])*
                                (Coord[0][0][Individual]-DestinationCoord[0][WhereTheyWantToGo])+
                                (Coord[0][1][Individual]-DestinationCoord[1][WhereTheyWantToGo])*
                                (Coord[0][1][Individual]-DestinationCoord[1][WhereTheyWantToGo]);
      if(TargetDistance[Individual]<LSq)
      {
         if(WhichHowFar[Individual]==OptionalWhichHowFar[Individual])
         {
            for(FutureWhichHowFar=OptionalWhichHowFar[Individual];
               FutureWhichHowFar<=OptionalLastWhichHowFar[Individual];
               FutureWhichHowFar++)

               WhichDestination[Individual][FutureWhichHowFar]=
               int(OptionalWhichHowFar[Individual]+(OptionalLastWhichHowFar[Individual]*rand())/
               (1.0*RAND_MAX)+0.5);
         }
        if(WhichHowFar[Individual]==LastWhichHowFar[Individual])
```

```
        Active[Individual]=2;
        else WhichHowFar[Individual]++;
      }
    }
}
```

The if() statement `if(WhichHowFar[Individual]==OptionalWhichHowFar[Individual])` allows an individual to switch from compulsory to optional destination.

Another numerical experiment based on individual's list of compulsory and optional destinations is shown diagrammatically in figure 5.46. In this case there are three groups of destinations and each group consists of three destinations arranged in a circle. The first destination in each groups is compulsory (0, 3, and 6) and all others are optional.

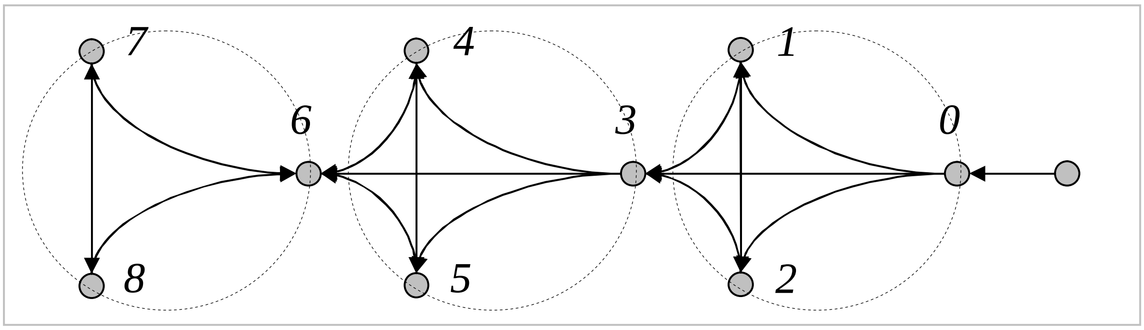


Figure 5. 46: Diagram represents three groups of compulsory and optional destinations

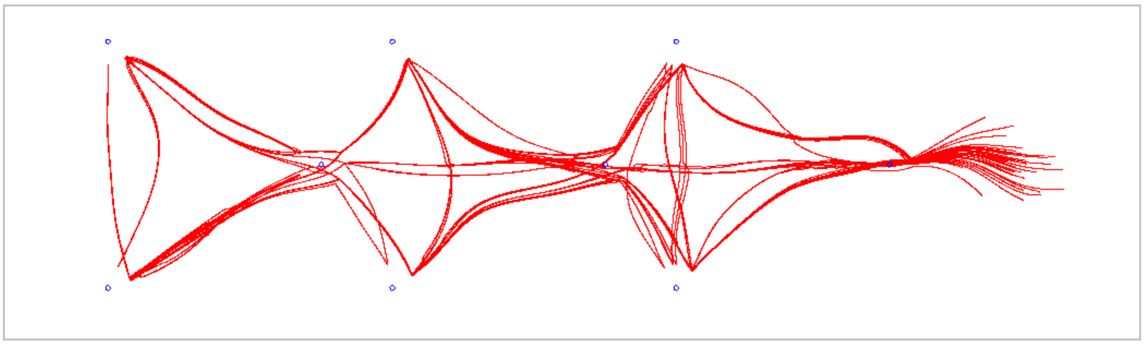


Figure 5. 47: Pattern generated by three groups of compulsory and optional destinations

Figure 5.47 shows possible patterns that can be generated by this arrangement of destinations. Such rules can be applied in terminal buildings where arrival and departure flow of movement fluctuate between compulsory (immigration, security check, etc) and optional destinations (café, toilets, etc.)

Individual's movement behaviour is calculated in *CalculateMotion()* function as can be seen below:

```
for(Individual=0;Individual<=LastIndividual;Individual++)
{
  WhereTheyWantToGo=WhichDestination[Individual][WhichHowFar[Individual]];
  if(Active[Individual]==1)
  {
```

```
    TargetDistance[Individual]=(Coord[0][0][Individual]-DestinationCoord[0][WhereTheyWantToGo])*
                                     (Coord[0][0][Individual]-DestinationCoord[0][WhereTheyWantToGo])+
                                     (Coord[0][1][Individual]-DestinationCoord[1][WhereTheyWantToGo])*
                                     (Coord[0][1][Individual]-DestinationCoord[1][WhereTheyWantToGo]);

    if(TargetDistance[Individual]<LSq)
    {
      if(WhichHowFar[Individual]==0)
      {
         for(FutureWhichHowFar=0;FutureWhichHowFar<=2;FutureWhichHowFar++)
         WhichDestination[Individual][FutureWhichHowFar]=
         int((3.0*rand())/(1.0*RAND_MAX)+0.5);
      }
      if(WhichHowFar[Individual]==3)
      {
        for(FutureWhichHowFar=3;FutureWhichHowFar<=5;FutureWhichHowFar++)
        WhichDestination[Individual][FutureWhichHowFar]=
        int(3.0+(3.0*rand())/(1.0*RAND_MAX)+0.5);
      }
      if(WhichHowFar[Individual]==6)
      {
         for(FutureWhichHowFar=6;FutureWhichHowFar<=8;FutureWhichHowFar++)
         WhichDestination[Individual][FutureWhichHowFar]=
         int(6.0+(2.0*rand())/(1.0*RAND_MAX)+0.5);
      }
      if(WhichHowFar[Individual]==LastWhichHowFar[Individual])
      Active[Individual]=2;
      else WhichHowFar[Individual]++;
    }
  }
}
```

5.6.6 Proposed generic model of people movement behaviour

Computer simulation of people movement can be generated using the rules described in previous sections. The choice of rules depends on the design problem under investigation and on the judgement of the architect to capture both movement behaviour and aesthetical design factors.

The rules can be summarised as follow:

i. Individuals cannot simultaneously occupy the same place in space.

ii. Their movement towards destinations is defined by their individual lists of destinations.

iii. The generation of movement is achieved at a local level predominately by the sign that is closest to the individual.

iv. If the movement towards a destination is obstructed by an obstacle (other individuals, boundaries, etc.) a virtual repulsive force is applied.

v. If an individual reaches a destination (that is his or her distance from the destination is less than the given length):

1. If the destination is the last one in the destination list, then the individual is deactivated, otherwise,

2. If the destination is not the last one, the individual switches to the next destination in his or her destination list.

Figure 5.48 shows flow chart of the generic movement behaviour model.

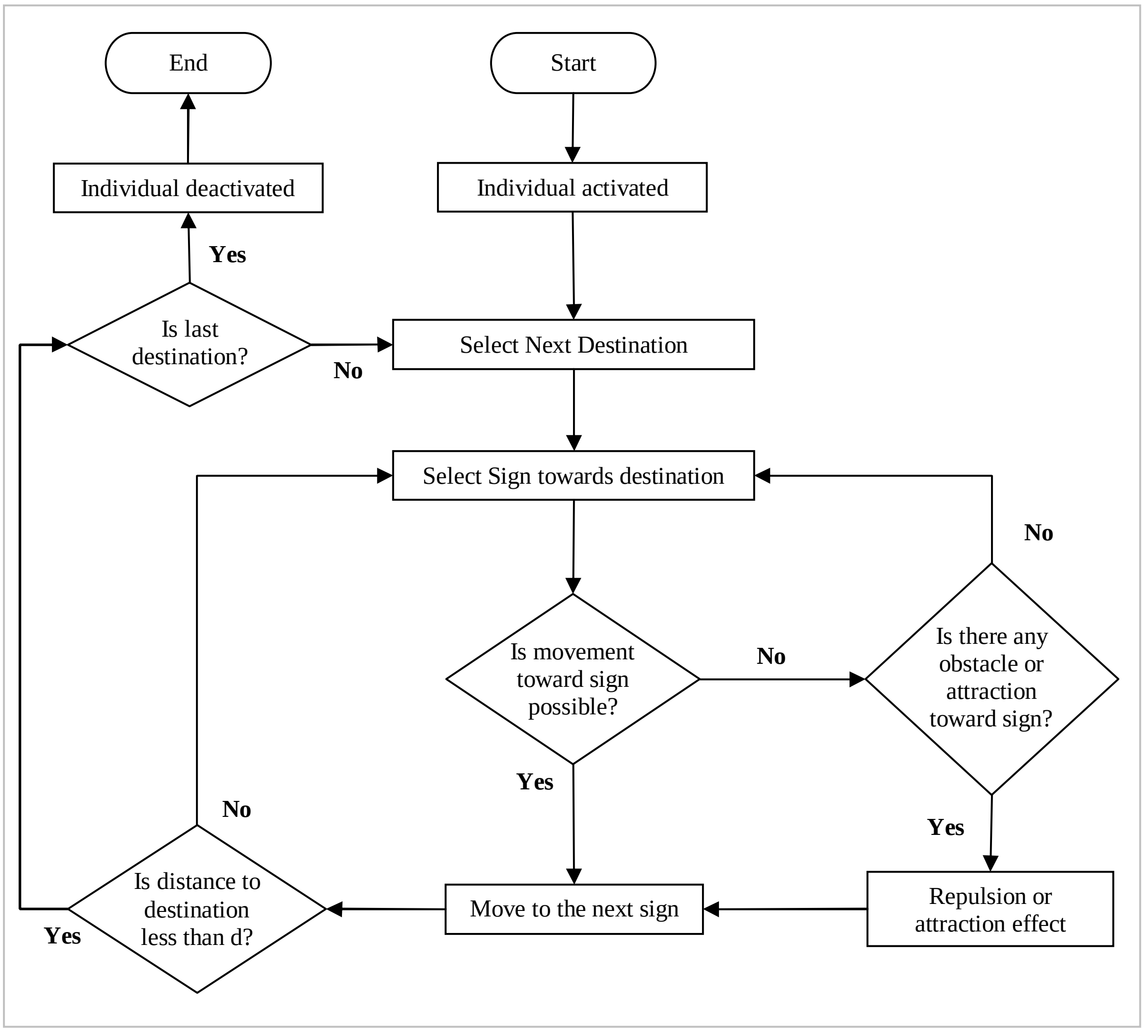


Figure 5. 48: Flow chart of the generic individual movement behavior model

The choice of movement rules is almost arbitrary, as the constants used within the rules. Thus under normal circumstances, the repulsive social force should be enough to make sure that people do not bump into each other, except occasionally (when in real life an apology would be called for). On the other hand during an evacuation the desire to get to the exit is much greater and may well overcome social repulsive forces so that contacts are frequently made. In each case the model is essentially the same, only the constants need to be changed, and this can only really be done by a trial and error or heuristic approach. Thus the following chapter examines the application of the Virtual Force model to the design and analysis of the circulation within a ferry terminal.

Chapter 6: Case study, a ferry terminal

6.1 Choice of building type

In order to examine the application of our technique, we need to choose a building type that can be used as case study. Various possible building types include:

i. Hospitals

ii. Stadia and sports arenas

iii. Museums and art galleries

iv. Office buildings

v. Theatres and opera houses

vi. Department stores

Each type is organized according to different circulation needs. In office buildings the circulation systems are used by a large number of people in short period of time, the rush hour. In department stores a large number of customers uses the space over a longer time period and the distinction between circulation and ‘shopping’ space is not particularly well defined. The shop owner hopes that you will come across something you didn’t know you wanted on your way to something that you are looking for.

Another building type is transportation interchange buildings including:

- Airport and ferry terminals
- Bus stations
- Railway stations

We have chosen a ferry terminal building as case study since:

1. It serves large number of people in short time period
2. It has a relatively simple circulation system

6.2 Ferry terminals

A ferry is a ship or boat carrying people and goods across a river, lake or short sea passage, up to a few hundred kilometres. A ferry could be a rowing boat, but here we are concerned with the circulation problems associated ships carrying hundreds of passengers. A cruise ship is different from a ferry in that the whole process is more leisurely, but the distinction is not always clear, especially on the longer ferry routes like Portsmouth-Bilbao.

The study of ferry terminals starts by giving a brief categorization of their basic types. Ferry terminals, as transportation interchange buildings, are located at transportation nodes. Figure 6.1 shows two sets of ferry routes between different transportation nodes.

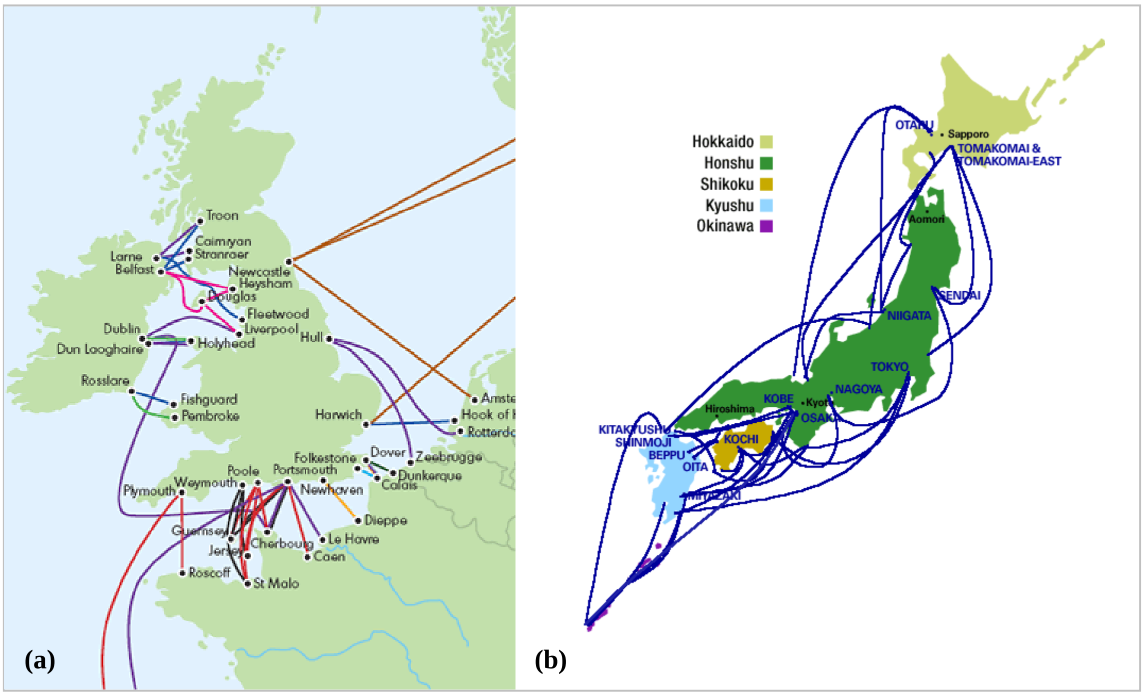


Figure 6. 1: a. Map of UK ferry routes [http://www.outandaboutlive.co.uk], b. Map of Japanese ferry routes [www.interq.or.jp]

Nodes can be observed individually or in relation to other nodes since they can function at a local and global level. At each transportation node, a ferry terminal serves as the entrance and the exit point for the port-city. A ferry terminal is the gate that allows passengers and vehicles to move in and out, arrive and depart. It has two main directions, towards the city and towards the ferries. Also, apart from its transportation role, it reflects the cultural and traditional characteristic of each place.

6.2.1 Port of Dover, UK

Figure 6.2 shows two aerial views of the Port of Dover in South East England, one of the biggest passenger ports in Europe and the nearest port to France.

Figure 6. 2: a. Port of Dover, b. Eastern Dock [Google Earth, 2008]

The port is multi-functional since it consists of two docks providing different services, the Western Dock that is used for cruises and the Eastern Dock that is used for freight and other traffic.

Cruise passengers have access to the ships through the cruise terminals (Terminal 1 + 2) in the Western Dock. Passengers proceed to check-in with their cruise operator, and then they wait in departure lounge till the announcement to embark. Then, they pass from the security point and after a short walk they board the ferries on foot.

The Eastern Dock Ferry Terminal serves foot passengers, cars, coaches, and trucks. Foot passengers need to pass from the travel centre for check-in and tickets sales. Then, they move to the embarkation lounge for passport check and then to the buses to take them to the ships. Passengers and drivers in cars, coaches and trucks pass through passport control in their vehicles (although coach passengers may need to get off and then on again) they wait in lanes before driving onto the ferry.

Figure 6.3a shows an aerial view of a berth with embarkation lanes and figure 6.3b shows vehicles driving onto the ferry. The ferries are Roll-on/roll-off (RORO) with vehicle doors at the bow and stern so that they dock bow first at one port and stern first at the other.

For most passengers Eastern Dock Ferry Terminal serves no purpose since they can wait for the ferry in their cars and then buy food and go to the toilet on the ferry.

Figure 6. 3: a. Aerial view of embarkation lanes and berth [Google Earth, 2008], b. Trucks ready to embark the ferry [Dover Harbour Board, 2007]

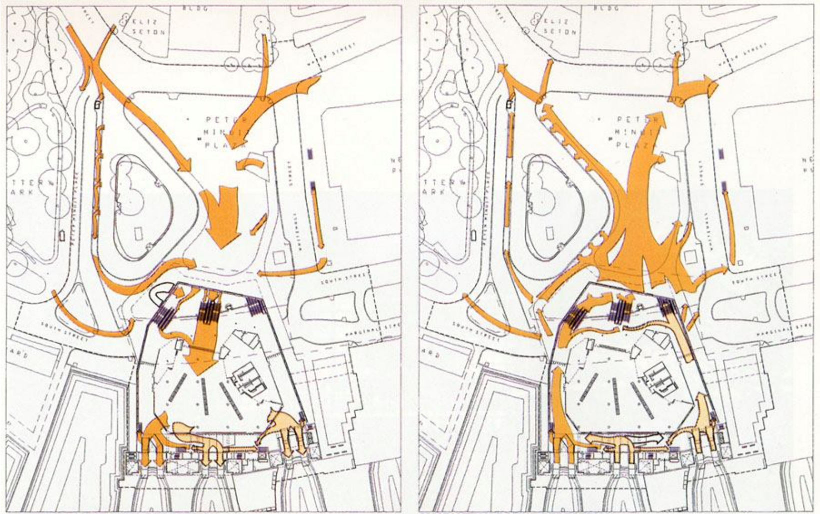
6.2.2 Whitehall Ferry Terminal, Lower Manhattan, New York, USA

Staten Island Ferry is a passenger ferry that began operation in 1817 and it connects the southern tip of Manhattan with Staten Island, both parts of New York City, USA. It is operated by the New York City Department of Transportation and the route is shown in figure 6.4a. The ferry ride is free of charge, takes 25 minutes and during the trip tourists or other passengers have views of Statue of Liberty and Manhattan.

Figure 6. 4: a. Ferry route connecting Manhattan with Staten Island, b. Aerial view of Staten Ferry terminal in Manhattan side [Google Earth, 2008]

The Staten Island Ferry terminal on Manhattan is also known as the Whitehall Terminal. The latest building on the site was designed by Venturi, Scott Brown and Associates and Ronald Evitts and Fred Schwartz Architects [http://www.revitts.com/Projects/whitehall.htm]. The final design scheme was the result of series design proposals that started from the winning of the international design competition in 1992 and the final design was completed in 1999.

In contrast to the Dover terminal, the Staten Island Ferry terminal (see figure 6.5) serves passengers and bicycles only. Previously vehicles were allowed as well but this operation was stopped following the 9/11 attacks. The movement of passengers to the embarkation and disembarkation points is done through the centric axis for departures and through the perimeter for arrivals as shown in figure 6.5.



STATEN ISLAND BOUND PEDESTRIAN FLOW CIRCULATION DIAGRAMS

MANHATTAN BOUND PEDESTRIAN FLOW CIRCULATION DIAGRAMS

Figure 6. 5: Staten Island and Manhattan bound pedestrian circulation diagrams. Whitehall ferry terminal [http://www.revitts.com/Projects/whitehall.htm]

There are four basic types of Staten Island ferry and the largest can carry 4,400 passengers. They are double-ended that means they have interchangeable bows and sterns, allowing them to shuttle back and forward between two terminals without having to turn around. Passengers embark or disembark at the end of the ferry so that the passengers who got on first get of first. This allows a quick passenger flow during embarkation and disembarkation.

Figure 6.6a shows a Staten Island ferry and figure 6.6b shows the Staten Island Ferry terminal on Manhattan seen from the sea.

Figure 6. 6: a. Staten Island ferry, b. Staten Island ferry terminal from the sea [Available from numerous internet sites]

6.2.3 Yokohama International Ferry Terminal, Japan

The Port of Yokohama is located on the north western edge of Tokyo Bay. It has a long history of more that 140 years as a leading international trade port. It has ten piers that serve different functions, but most of them are used for cargo and vehicle exports.

Figure 6.7a shows an aerial view of the Port of Yokohama and figure 6.7b shows Osanbashi pier that handles both ferry and cruise passenger traffic.

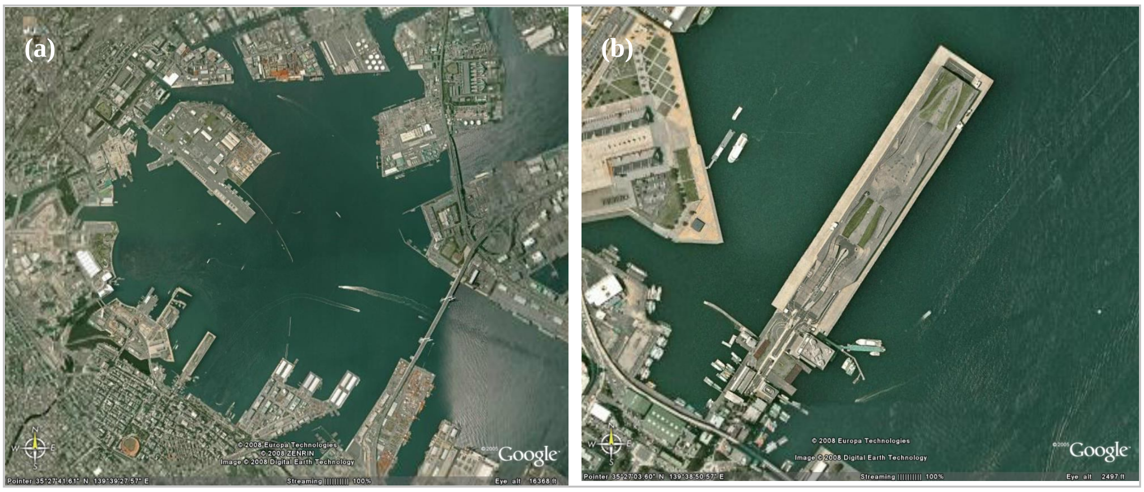


Figure 6. 7: a. Port of Yokohama, b. Yokohama International Passenger terminal in Osanbashi pier [Google Earth, 2008]

Yokohama International Passenger terminal is located on Osanbashi pier. It was designed by Foreign Office Architects (Farshid Moussavi and Alejandro Zaera-Polo) and was completed in 2002.

As is the case with the Staten Ferry terminal, the Yokohama terminal was designed to serve only passengers but in this case for national and international ferries and cruises. It has two main floors, plus a complex sloping roof that can be walked on in part. The ground floor that is mainly for passenger car parking and on the first floor there is the ‘lobby’, customs, immigration and quarantine facilities, and a multi-purpose hall. Figure 6.8 shows the basic plan of the first floor.

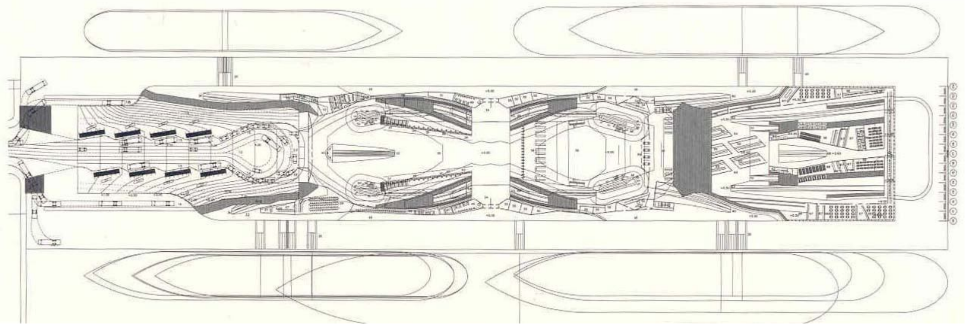


Figure 6. 8: Basic terminal plan of Yokohama Ferry terminal [Moussavi and Zaera-Polo, 2002]

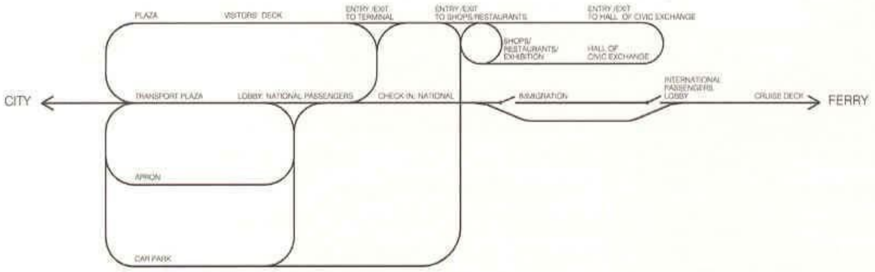


Figure 6. 9: Circulation diagram of Yokohama Ferry Terminal [Moussavi and Zaera-Polo, 2002]

Figure 6.9 shows the circulation diagram of the Yokohama Ferry Terminal and the architects wrote about circulation strategy [Moussavi and Zaera-Polo, 2002, p.11]:

> ‘There was something very interesting about transportation buildings that we were interested in exploring. Usually, a transportation building works as an input-output device, with very clear orientation: departures and arrivals. We were more interested in exploring the possibility of a transportation infrastructure that could operate less as a gate, as a limit, and more as a field of movements with no structural orientation. We also looked specifically to pier structures, to discover the characteristic linearity of these structures: you enter from the root, walk to the end, and you either leave in a ship, or you have to return on your own steps. This imposes a strong orientation on the space that we were interested to challenge. Our first move was to set the circulation diagram as a structure of interrelated loops that allow for multiple return paths. The connection between the circulation paths was always set as a bifurcation, so that rather than setting the program as a series of adjacent spaces with more or less determined limits, we articulated them in the continuity of a branched sequence along the circulator system. What we then called ‘the no-return diagram’ was basically the first attempt to provide the building with a particular spatial performance’

Another decision of the architects was to design a flat building in order to emphasize the ships and not to make the building the main ‘sign’. For this reason the height has been kept as low as possible with the rooftop functioning as public space where people can get close to the port area and could have close views of the sea and the ships.

Figure 6. 10: a. View of Yokohama International Ferry Terminal, b. Queen Elizabeth 2 cruise ship [Available from numerous internet sites]

Figure 6.10a shows a view of the Terminal with a ship and figure 6.10b shows the Queen Elizabeth 2.

6.2.4 Comparison of the three terminals

The three terminals at Dover, Manhattan and Yokohama are very different, but in each case the main design driver is the circulation system. At Dover most of the circulation is vehicular, at Manhattan the ferry forms part of a mass transportation system while at Yokohama the ferries and cruise ships are used for longer journeys. The study of the typology of various ports and ferry terminals shows that their function and design depends on their special needs and requirements that are different in each case.

6.3 Design competition in the Port of Limassol, Cyprus

6.3.1 The brief and background

The case study uses the real brief for a ferry terminal design competition in the Port of Limassol (Greek: *Λεμεσός*, *Lemesos*), Cyprus [Cyprus Ports Authority, 2003]. The competition was announced in 2003 aiming at the redevelopment of the port area and the design of various buildings including passenger ferry terminal in order to satisfy current and future needs. The author entered the competition in collaboration with Vardas Architectural Studio in Cyprus [http://www.vardastudio.com/].

The Port of Limassol is the biggest port in Cyprus and it serves as the main entrance point by the sea. It is located on the south coast and it is adjacent to the Limassol town (population 180,000). The port has operated since 1970’s as a trade port but following the Turkish invasion in 1974 it serves passenger traffic as well. The previous main port, Famagusta, is in the *de facto* Turkish Republic of Northern Cyprus.

Figure 6.11 shows possible ship routes between the Port of Limassol and other Mediterranean destinations. The map was designed based on the information derived from the Cyprus Ports Authority’s web site [http://www.cpa.gov.cy/].

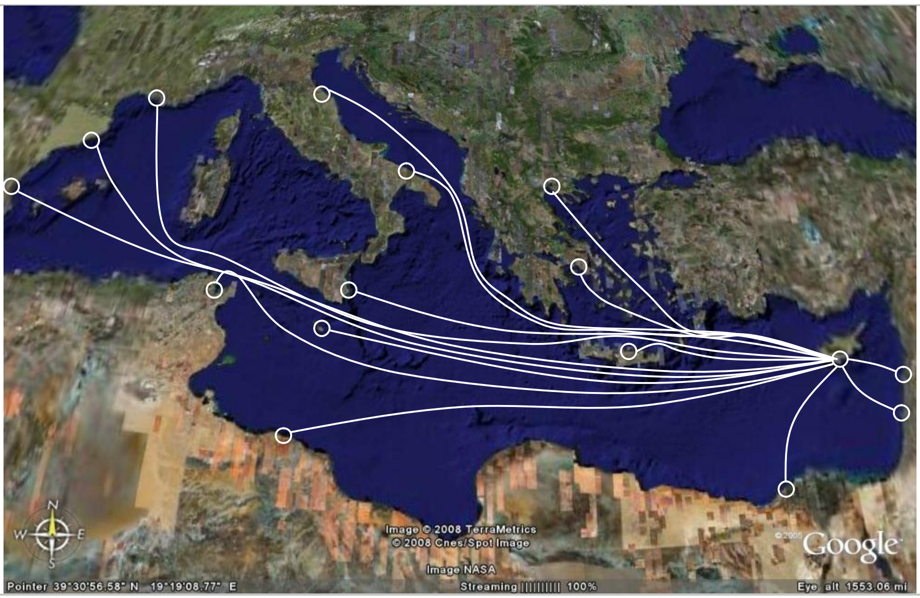


Figure 6. 11: Ship routes between Cyprus and Mediterranean cities [Google Earth, 2008]

The increasing cargo and the passenger traffic in the same port area caused many problems due to the crossing of different flows.

Figure 6. 12: Cargo ship and passenger ferry in the Port of Limassol [Original source unknown]

In the beginning of 1990’s a planning proposal suggested the organization of the port in two zones. The trade port that is located in the western area and the passenger port that is to be located in the eastern area.

Figure 6.13 is an aerial view of the Port of Limassol showing the trade and the passenger ports. The circle indicates the existing terminal building that is a temporary structure without any architectural interest.

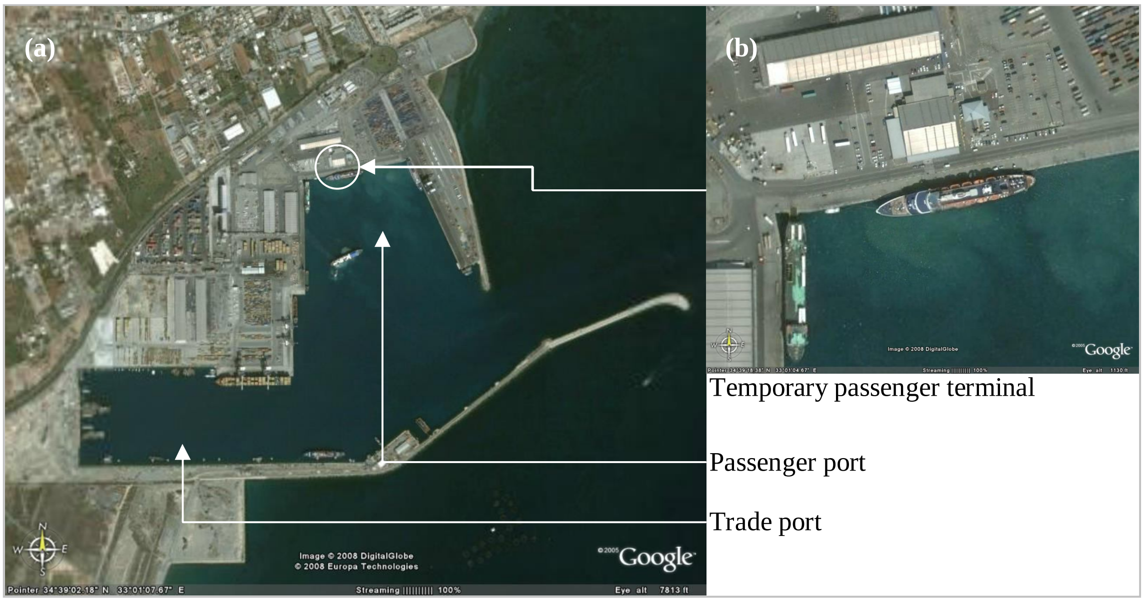


Figure 6. 13: a. An aerial view of the Port of Limassol, b. Temporary passenger terminal building [Google Earth, 2008]

The planning proposal was applied partially but it was not successful for a number of reasons:

i. The growing potential of passenger traffic

ii. The use of passenger port not only for passengers but also for cargo

iii. The location of the temporary terminal building

Based on the planning proposal and the future development plan, the competition brief suggested the location of the new passenger terminal building behind the eastern dock of the passenger port. This location was suitable since:

i. The building would be very near to the two docks (northern and eastern) that are used currently for passenger traffic

ii. It would allow possible connection with future extension of the passenger port on the eastern side of the port

Figure 6.14 shows an aerial view of the passenger port indicating with a red line the area of investigation. The main access to the area is by road via the roundabout on the north side of the passenger port as shown with orange arrow[15]. There is no surface or underground railway connection. The embarkation and disembarkation points for passengers and vehicles are illustrated with yellow squares (ferries).

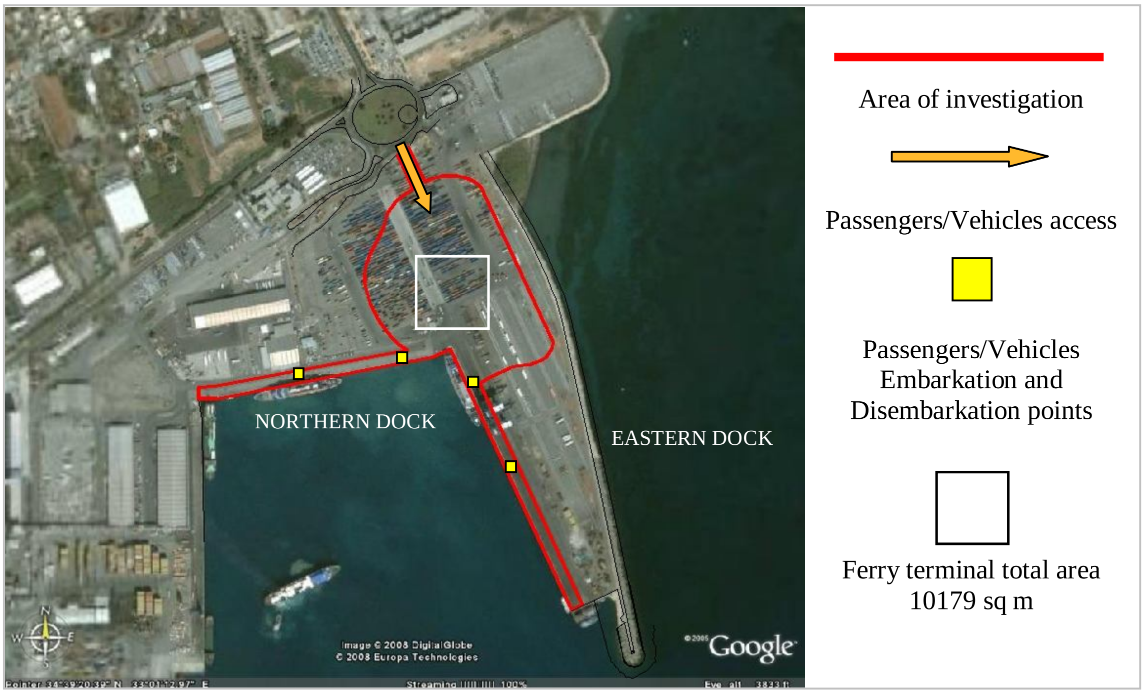


Figure 6. 14: Aerial view of the passenger port with northern and eastern docks [Google Earth, 2008]

Figure 6.14 also shows a white square with the area of the footprint of the ferry terminal building defined in the competition brief. The shape and location of the footprint was to be determined by the architects entering the competition. The maximum number of floors is two and the building height is not defined. (Limassol master plan: Zone BB2).

The passenger port that formed the competition brief was mainly to be used for passengers and limited number of vehicles. According to the competition brief [Cyprus Ports Authority, 2003] there are two main types of passenger:

i. Cruise passengers. Cruise passengers form the majority of passengers using the Port of Limassol which is an important cruise center in the Eastern Mediterranean. It is the permanent base of many cruise ships that make trips in the region on a regular basis. Domestic and international passengers travel by ship from Limassol to destinations in Syria, Israel, Lebanon, Egypt, and the Greek islands. Cruise passengers can be divided into: a. Round trip

---

[15] One of the competition requirements was the design of a road leading to the ferry terminal and to the accompanying facilities like the port authority administration building, the police authority building and the fire fighting building. These buildings are not included in the area of investigation but in other positions and their design is out of current research scope [Cyprus Ports Authority, 2003].

cruise passengers (start and end their trip in the Port of Limassol) and b. Transit passengers (start in Limassol and end their trip in other ports or *vice versa*).

ii. Foot and car passengers

Foot and car passengers are limited although some ferries connect Limassol with various destinations like Piraeus (Greece), Greek islands, Syria and Israel.

The passenger capacity of cruise and passenger ferries varies and this can be ranged from 1000 to 2500 or more according to the size of ferry.

Figure 6.15 and figure 6.16 below show the number of passenger arrivals and departures according to the Cyprus Ports Authority annual report in 2005. Figures show that the number of passengers has decreased in recent years due to the political instability in the Eastern Mediterranean region.

Figure 6.15 shows the passenger traffic in the Port of Limassol (Lemesos) and the Port of Larnaca (Greek: *Λάρνακα*, *Larnaka*), also on the south coast of Cyprus, the years 2004 and 2005.

Αφίξεις - Αναχωρήσεις Επιβατών
Passenger arrivals - Departures

ΛΕΜΕΣΟΣ/LEMESOS

	ΑΦΙΞΕΙΣ/ARRIVALS	**ΑΝΑΧΩΡ.**/DEPART.	**ΣΥΝ.**/TOTAL
2004	228379	226602	454981
2005	250448	250817	501265

ΛΑΡΝΑΚΑ/LARNAKA

	ΑΦΙΞΕΙΣ/ARRIVALS	**ΑΝΑΧΩΡ.**/DEPART.	**ΣΥΝ.**/TOTAL
2004	27066	27092	54158
2005	33385	33938	67323

(a) **(b)**

Figure 6. 15: Passenger arrivals and departures for the years 2004 and 2005, Annual report [Cyprus Ports Authority, 2005]

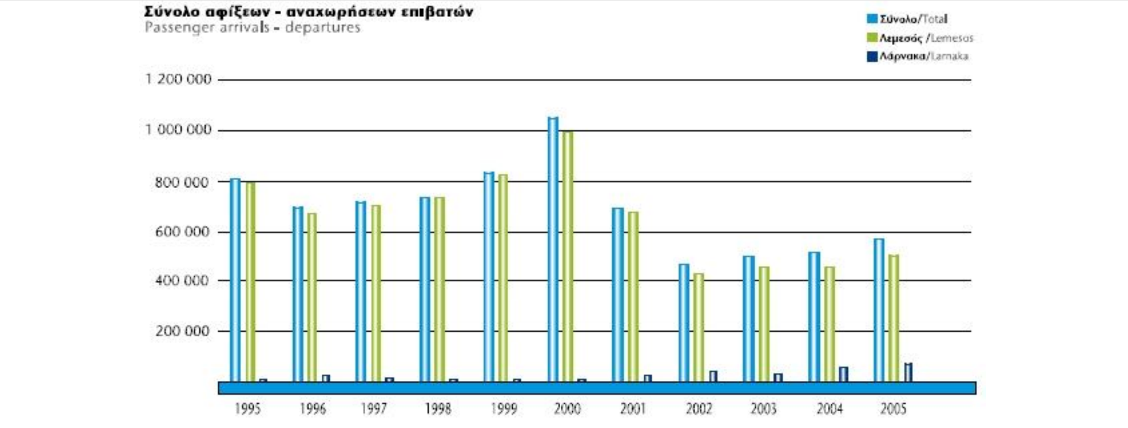


Figure 6. 16: Passenger arrivals and departures from 1995 till 2005, Annual report [Cyprus Ports Authority, 2005]

Figure 6.16 shows the total number of passenger arrival and departures (blue bar) from 1995 till 2005. Passenger traffic in the Port of Limassol is shown with the yellow/green bar.

Cruise and ferry passengers[16] have access to the area of investigation using buses (traveling as groups), private cars and taxis. Passengers can leave their cars in the parking area of the terminal building. There is sufficient outside ground space for there to be no need for a parking to be incorporated within a building.

The number and frequency of passenger arrivals and departures depends on the season. The peak passenger traffic period is during the summer due to the national and international holidays. During the winter period the number of passengers and the traffic rate decreases dramatically.

The part of ferry terminal that serves vehicular traffic is separate from the passenger terminal and hence it is not investigated in the current research.

The ferry terminal building is an entrance/exit point of people in Cyprus. Thus, its purpose, according to the competition brief, is dual [Cyprus Ports Authority, 2003]:

i. To control effectively the passenger traffic in and out of the country. Some years ago the passenger control was done on the ferries and not inside the terminal building. However, the control rules have been changed due to the Schengen Agreement (1990), Cyprus joining the European Union (2004), the danger of terrorist attacks and illegal immigration. In order to follow all requirements the design of ferry terminal must follow all international standards regarding passport checks and the security control. Passenger arrivals from Schengen countries (that is most European countries) and non-Schengen countries need to pass through different check points in arrival hall. The organization of passport check and control points is discussed later in this dissertation.

ii. To provide a functional and pleasant environment.

---

[16] The data regarding the number of passengers is derived from the design competition brief. Although, the number of people is not given explicitly, the functional requirement list includes the number of people using each space. Also, the area of each space is given in square meters and this may help the further estimation of the visitors' usage of space in different periods. The data is assumed to be accurate.

6.3.2 First organization of case study

- **Criteria of evaluation**

The space allocation and organization of facilities in the ferry terminal (entrances, exits, departure lounges, etc) is directly related to the satisfactory passenger flow. In order to specify the evaluation criteria of passenger flow, we can draw comparisons with the criteria for structural performance of buildings. Building structures are evaluated based on the ultimate and serviceability 'limit states'. The ultimate limit state means collapse of all or part of the structure causing serious injuries and death. The serviceability limit state means buildings have 'minor' (that is not immediately life-threatening) structural problems (cracks, deflections, leaking, etc.).

In architectural terms the equivalent of the ultimate limit state is if the building totally fails to function, large numbers of people miss their ferries or escape passport control. The serviceability limit state is violated in architectural terms if people are uncomfortable, or they don't like being in the building. The difference between the two limit states is that violating the serviceability limit state is very unfortunate, but violating the ultimate limit state is unthinkable.

This means that any design process must first of all show that the ultimate limit state is not violated and then consider the serviceability limit state.

The evaluation criteria for satisfactory passenger flow include:

1. The effective passenger flow:

   The movement of passengers through the departure and arrival points should be as smooth as possible due to the large number of passengers using the space at the same time. Phenomena that are observed in bottlenecks and intersection points between flows that may cause panic situations and serious dangers for the passengers must be avoided. The failure to fulfill this criterion is an ultimate problem.

2. The efficient use of space:

   The space should be clearly marked without complicated signs and the passengers should be able to move without taking complicated path systems. The failure to fulfill this criterion is a serviceability circulation problem.

3. The aesthetical design aspects:

   The facilities must be shaped and organized in such way that will help the effective use of space without obstacles that block movement and so on. Also, the form of space (static or dynamic) should be considered since it might influence the way passengers move and experience it. This is a serviceability problem.

- **Scales of investigation**

Passenger movement and circulation design exists at different design scales like urban, regional and building. The computer code will be tested using three different scales that represent different degrees of complexity:

i. Macro-scale
ii. Meso-scale
iii. Micro-scale

- **Set-up environment**

Figure 6.17 shows the digital[17] representation of the area of investigation. It includes the northern and eastern docks, the streets, the surrounding buildings, the passengers and vehicles access as well as their embarkation and disembarkation points (see figure 6.17b).

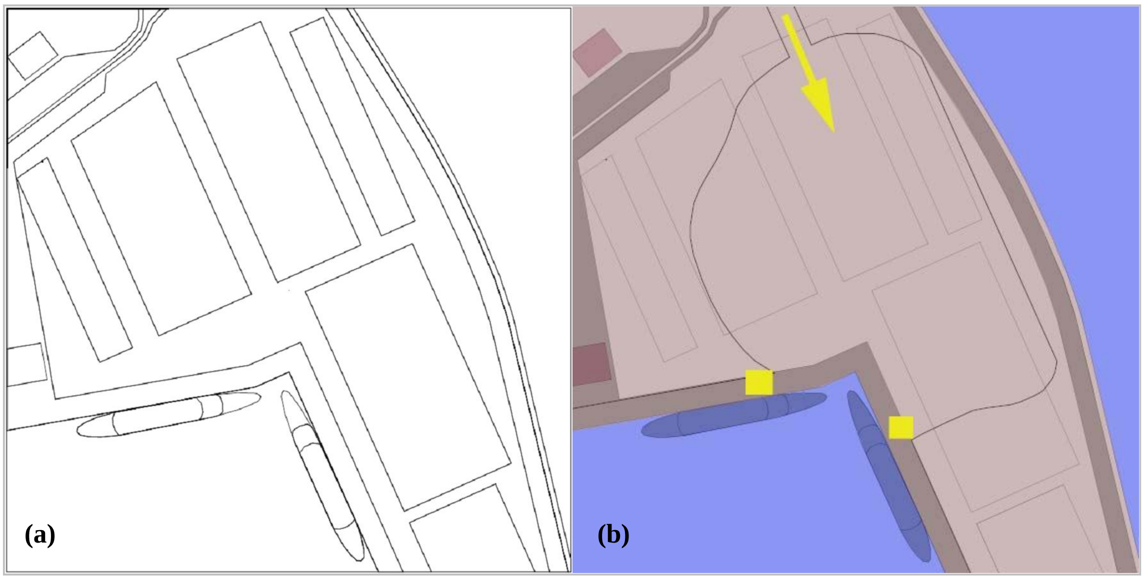


Figure 6. 17: Passenger port and area of investigation for use in the computer model

---

[17] The data for background was imported from 3DStudio using a technique based on upon that described in http://www.spacesimulator.net

6.4 Macro-scale: Path systems design

Path systems connect functional areas and formulate the overall organizational structure of the design problem.

- **Analytic design tasks**

Analytically, the study investigates path systems (conceptual level) in the entire area including both the outdoor (public spaces) and the indoor areas of building. Paths can provide initial solutions for the efficient use of space and at the same time improve the spatial configuration of the design.

- **Computational tasks**

Passenger movement behaviour is applied as a driving force towards the generation of path systems. The computer model does not aim to accurately capture a real situation, but to assist the design decision-making process.

- **Known information**

Known information includes:

i. The geometrical definition of the environment.

ii. The initial positioning of the functional areas, entrances, exits, immigration control points, and so on. Also, the passengers' entry and the destination points.

iii. The rules that formulate passenger movement behaviour.

- **Evaluation of results**

The evaluation will be based on:

i. The results from various passenger behaviour phenomena that can be found in intersection points, and spaces of blockage

ii. The architect's aesthetical and other movement criteria

6.4.1 Passenger flow in terminals

Passenger flow in a ferry terminal is similar to that in an airport. Two main flows are the departures and the arrivals, which are directly related with the baggage flows.

Figure 6.18 and 6.19 show two representative arrival and departure flow diagrams together with baggage flow that is represented by a dotted line. Departing passengers drop their baggage at check-in and reclaim them at the arrivals baggage collection point. Check-in and baggage claim points are indicated with red and blue circles respectively.

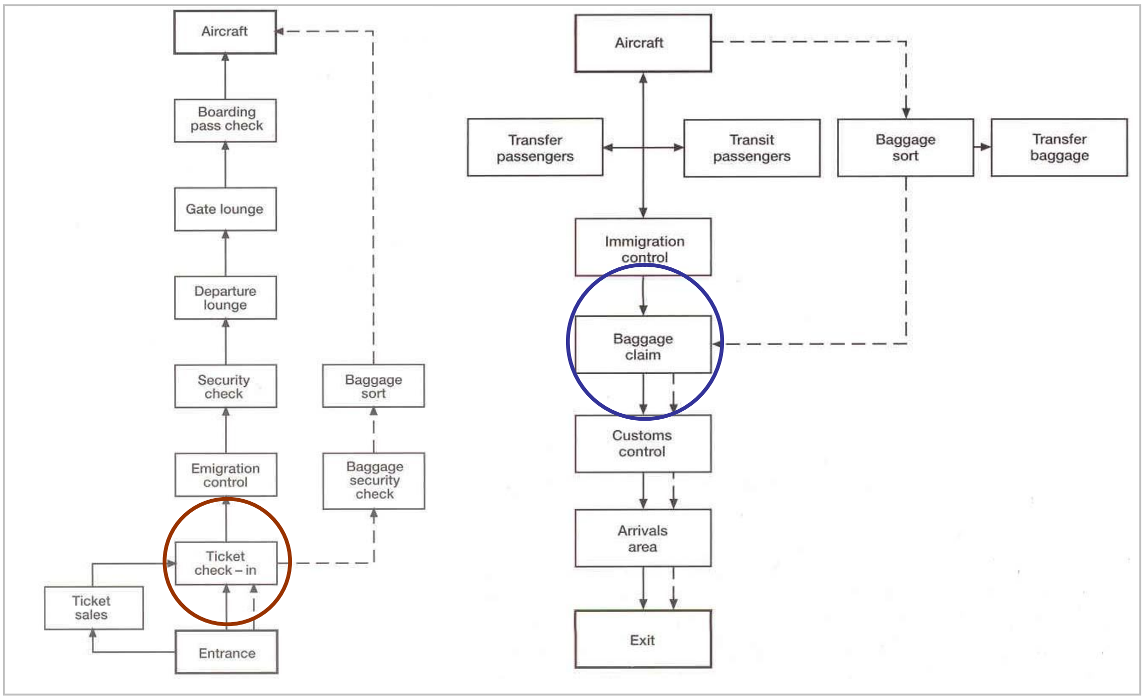


Figure 6. 18: Departure and arrival flow diagrams [Edwards, 1998]

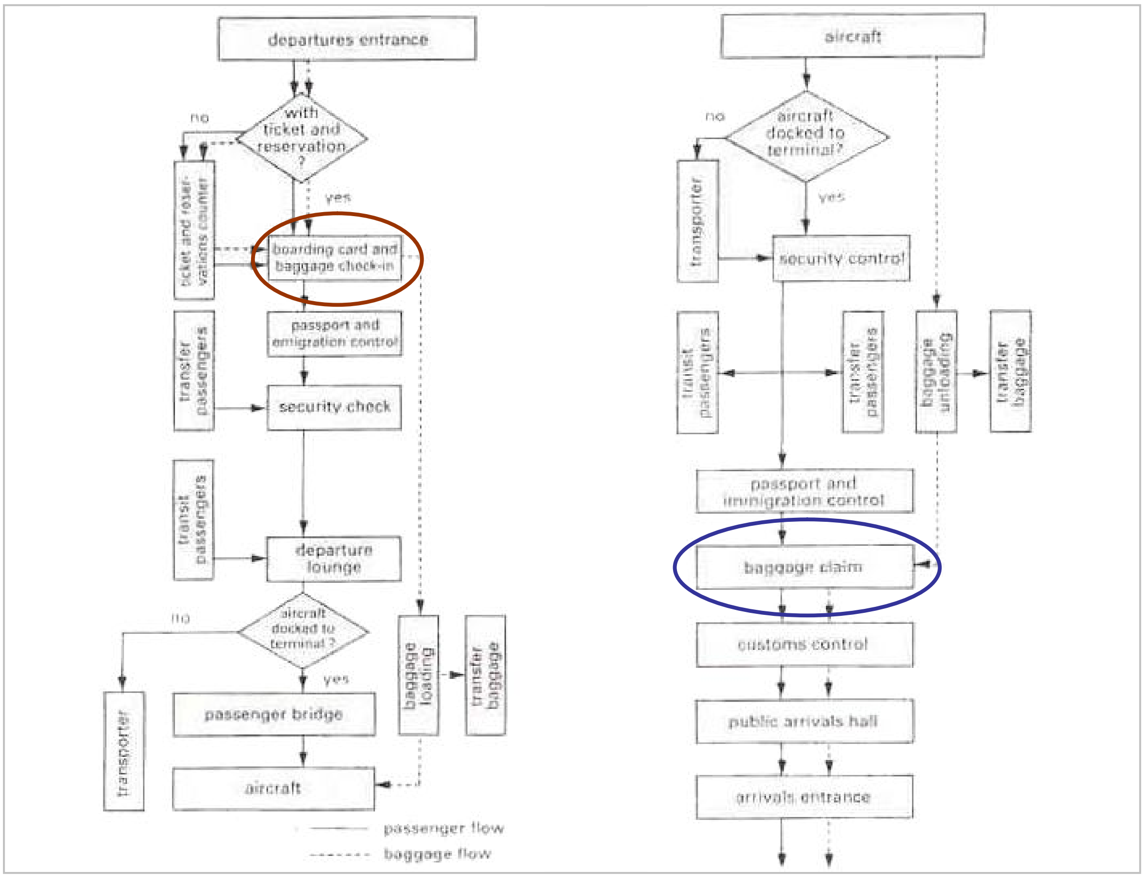


Figure 6. 19: Functional diagram of a terminal building [Neufert, 2000]

Figure 6.19 shows apart from the organization of departure and arrival flows the transit passenger flow.

Check-in and baggage claim together with the immigration control, security check and custom control points are necessary facilities in any terminal building due to security and other reasons. Terminals have different locations for these facilities based on various criteria. We start our study on terminal buildings by examining briefly the organization of facilities in two characteristic examples.

Figure 6.20 shows the check-in lounge in London Heathrow terminal 1.

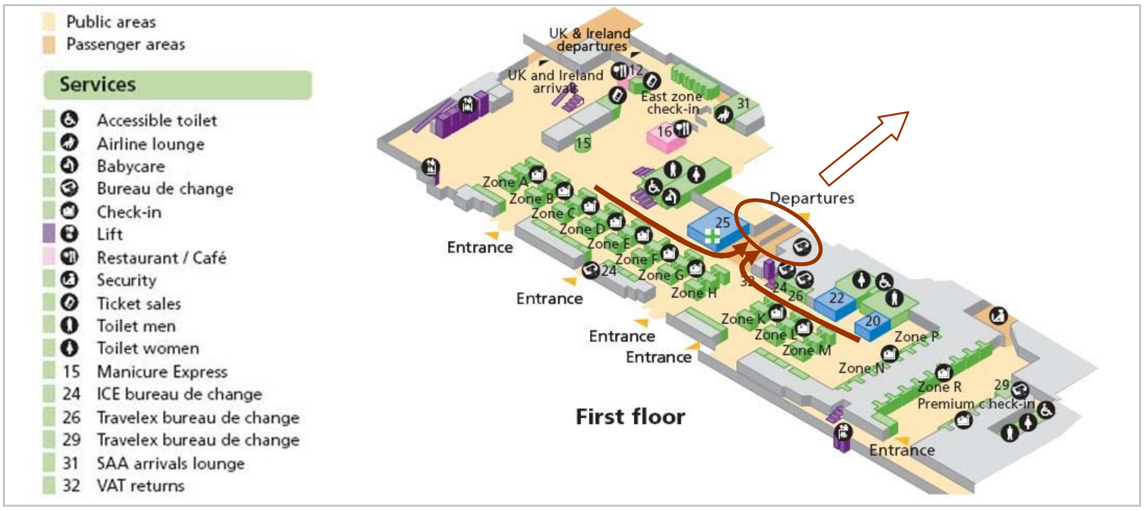


Figure 6. 20: Check-in map in London Heathrow terminal 1 [http://www.heathrowairport.com/]

Passengers drop their baggage and move to the passport control that is indicated with red circle. After passengers pass from the security check-point they move to the departures lounge as shown in figure 6.21.

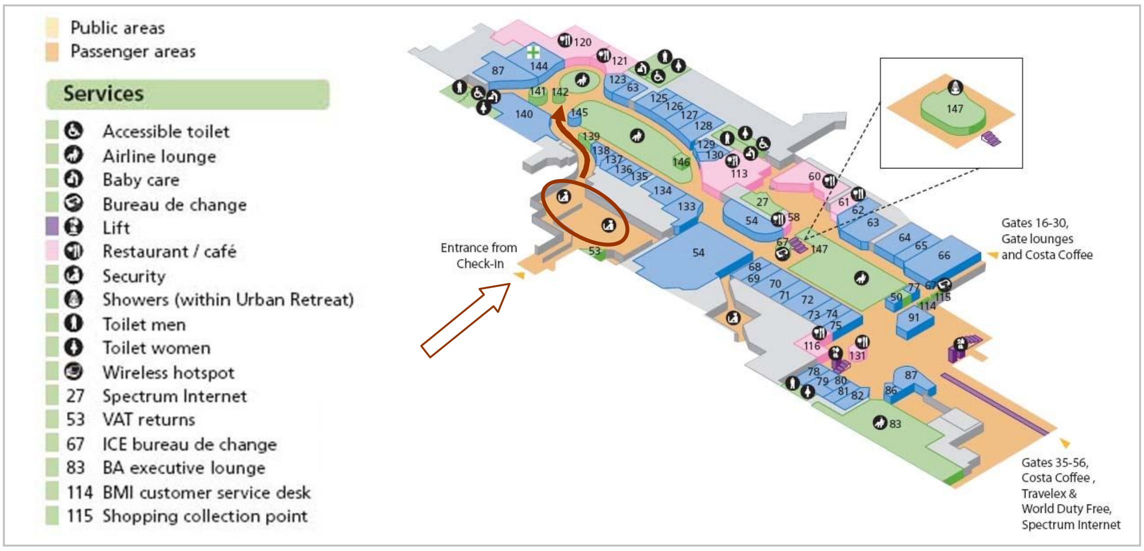


Figure 6. 21: Departures in London Heathrow terminal 1 [http://www.heathrowairport.com/]

Figure 6.22 shows the arrival flow where after passport/immigration control passengers move to baggage claim, then to custom control point and finally to the arrival hall.

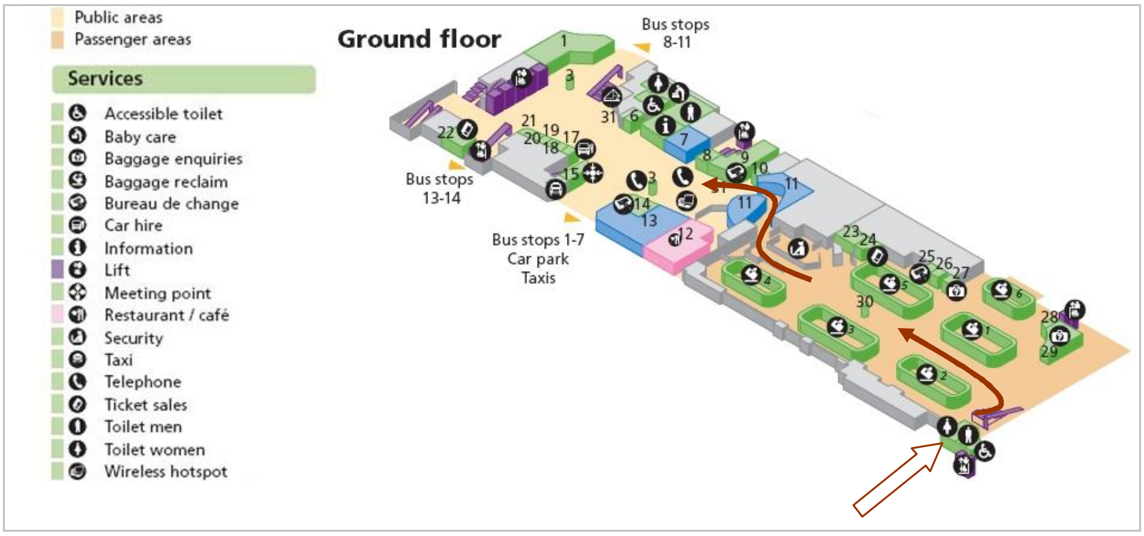


Figure 6. 22: Arrivals in London Heathrow terminal 1 [http://www.heathrowairport.com/]

The brief analysis of London Heathrow terminal 1 shows that departures and arrivals are separated and passenger do not meet in any part of the building as shown in figure 6.23.

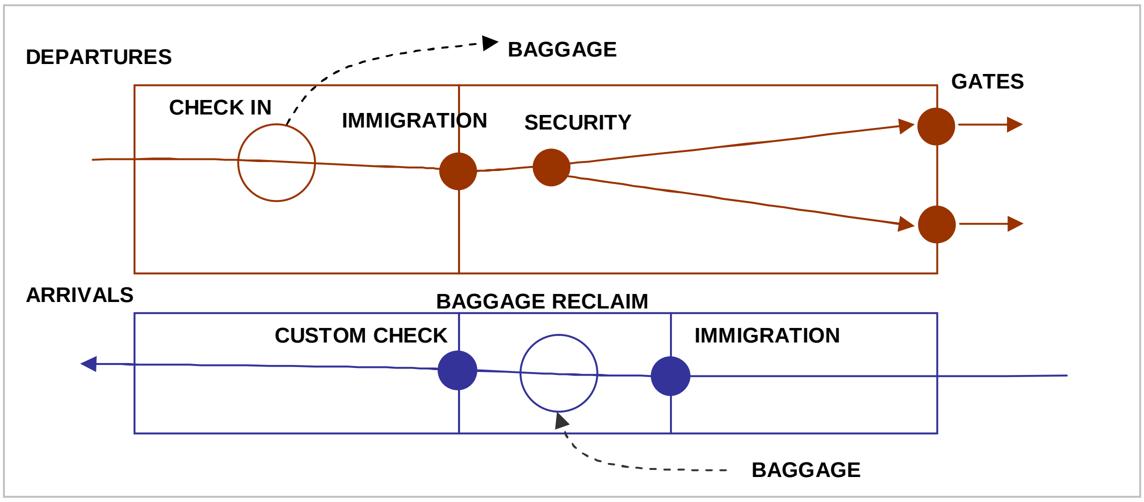


Figure 6. 23: Analysis of departures and arrivals in London Heathrow terminal 1

At Lulsgate International Airport, Bristol, the organization of terminal facilities is different. The terminal is a two storey building with the check-in lounge and arrivals on the ground floor and departures on the first floor.

Figure 6.24 shows the ground floor with the check-in lounge.

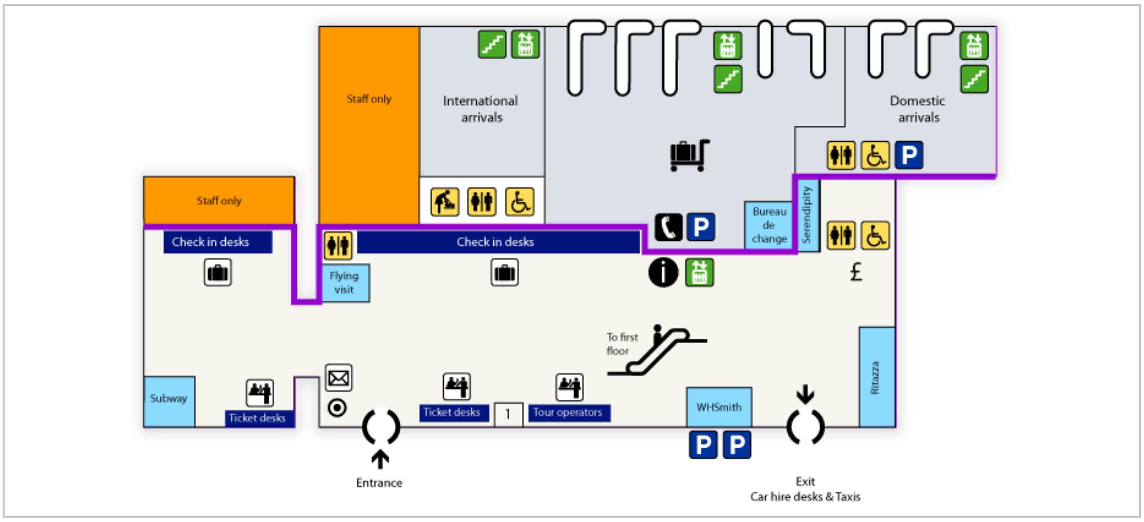


Figure 6. 24: Bristol terminal ground floor/check-in lounge [http://www.bristolairport.co.uk/]

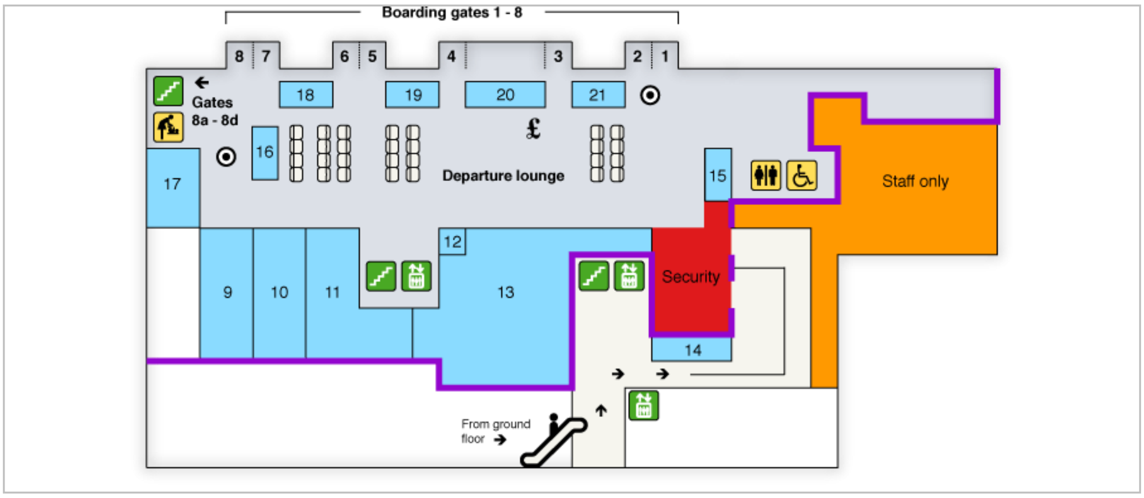


Figure 6. 25: Bristol terminal first floor/departure lounge [http://www.bristolairport.co.uk/]

After check-in passengers move to the first floor using the escalator or the lift. Figure 6.25 shows the first floor. Passengers pass from through passport control, security check and then to the departure lounge. Passengers move to the airplanes on foot or by busses.

Arrival area is in the ground floor. After passengers pass through immigration control they move to baggage claim area. Then they pass through customs control and move to the exit that is in the same space as the check-in lounge.

Figure 6.26 shows an analysis of Bristol terminal. Although departures and arrivals are in the same building they are organized on different floors.

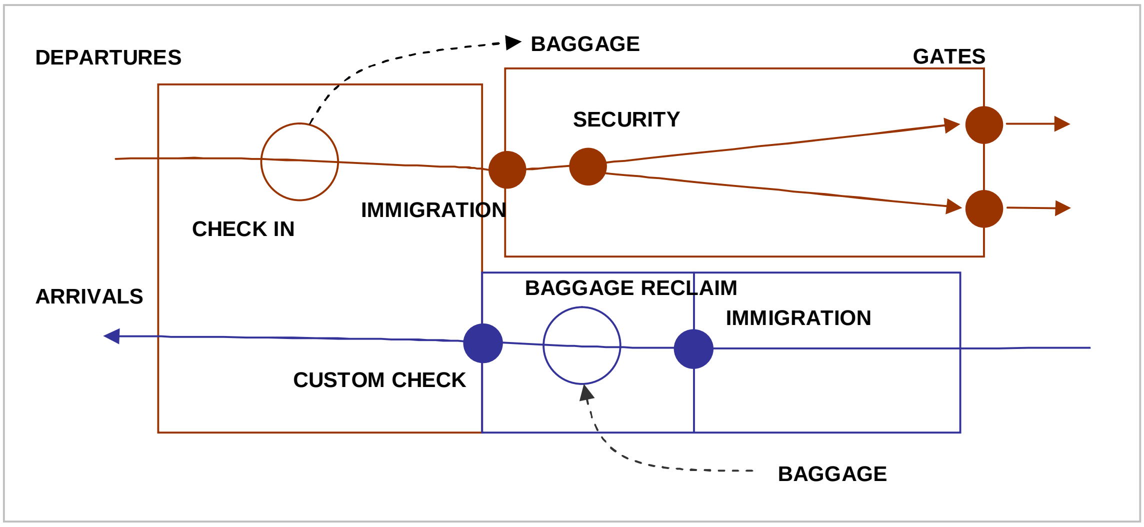


Figure 6. 26: Analysis of departures and arrivals in Bristol terminal

Terminal buildings may be designed in different ways though the main organization of their control points remains the same. Ferry terminals that handle national and international passenger traffic follow the same standards.

6.4.2 Analysis of brief in Limassol ferry terminal

Our investigation starts by analysing the competition brief[18]. The table 6.1 below is an attempt to describe the various spaces (offices, departures lounge, arrival hall, etc.), their interaction with other spaces and their area in square meters.

	A/A	DESCRIPTION	INTERACTIONS	sq.m
	1.0	**FOYER/PASSENGERS HALL**		
1	1.1	**Foyer**	Interacts with 1.13, 1.5, 1.12, 1.6, 1.9, 1.10, 1.4, 4.9A, 1.8, 1.7, 9.1, 9.6, 7.1, 7.2, 7.3, 11.5, 11.3, 11.2, 11.1, Arrivals	1000
2	1.2	**Baggage conveyors/Convey belts**	Interacts with 1.3, 1.5	120
3	1.3	**Baggage control**	Interacts with 1.2, 11.6	15
4	1.4	**Bank/ATM Machines**	Interacts with 1.1	45
5	1.5	**Check-in**	Interacts with 1.1, 1.2	120
6	1.6	**Information desk**	Interacts with 1.1, 1.11	12
7	1.7	**Shops**	Interacts with 1.1	100
8	1.8	**Restaurant**	Interacts with 1.1	60
9	1.9	**Internet + Post office**	Interacts with 1.1	25
10	1.10	**Car rent office**	Interacts with 1.1	30
11	1.11	**Office/Storage for lost items**	Interacts with 1.6, 7.1	15
12	1.12	**Public phones**	Interacts with 1.1	---
13	1.13	**Trolleys for departures**	Interacts with 1.1	---
	2.0	**DEPARTURES**		
14	2.1A	**Departures hall /Phase A**	Interacts with 5.9, 9.6, 2.1A.1, 2.1A.2, 2.1A.3, 2.1A.4, 2.1A.5, 2.1A.6, 2.1B	1100
15	2.1A.1	**Gate 1**	Interacts with 2.1A, 11.7, Ferries/Departures	
16	2.1A.2	**Gate 2**	Interacts with 2.1A, 11.7, Ferries/Departures	
17	2.1A.3	**Gate 3**	Interacts with 2.1A, 11.7, Ferries/Departures	
18	2.1A.4	**Gate 4**	Interacts with 2.1A, 11.7 Ferries/Departures	
19	2.1A.5	**Gate 5**	Interacts with 2.1A, 11.7, Ferries/Departures	
20	2.1A.6	**Gate 6**	Interacts with 2.1A, 11.7, Ferries/Departures	
21	2.1B	**Departure hall/Phase B**	Interacts with 2.1A, 2.1B.1, 2.1B.2, 2.1B.3, 1.7	600
22	2.1B.1	**Gate 1**	Interacts with 2.1B, 11.7, Ferries/Departures	

---

[18] The original competition brief provides with the following information: analytical description of spaces and their interactions, the number of spaces, their areas in square meters, the number of people, and the total area for each space in square meters [Cyprus Ports Authority, 2003].

23	2.1B.2	**Gate 2**	Interacts with 2.1B, 11.7, Ferries/Departures	
24	2.1B.3	**Gate 3**	Interacts with 2.1B, 11.7, Ferries/Departures	
25	2.2	**Restaurant**	Interacts with 2.1B, 11.7	80
26	2.3	**Shops**	Interacts with 2.1B, 11.7	500
27	2.4	**VIP hall**	Interacts with 2.1B, 11.7	35
	3.0	ARRIVALS		
28	3.1	**Arrivals hall**	Interacts with Ferries/Arrivals, 4.10A, 4.6, 6.8, 6.3, 6.2, 6.1, 5.11	500
29	3.2	**Trolleys for arrivals**	Interacts with 3.4A	40
30	3.3A	**Open air covered space for baggage/ Arrivals/ Phase A**	Interacts with 3.4A, 3.3B	500
31	3.3B	**Open air covered space for baggage/ Arrivals/ Phase B**	Interacts with 3.3A, 3.4B	250
32	3.4A	**Baggage reclaim hall/ Arrivals/ Phase A**	Interacts with 3.5, 4.10A	600
33	3.4B	**Baggage reclaim hall/ Arrivals/ Phase B**	Interacts with 3.4A, 3.3B	600
34	3.5	**Customs check space for arrivals**	Interacts with Arrivals, 3.4A	400
35	3.6	**Cyprus Tourism Organization (CTO)**	Interacts with 3.4A, 9.4	25
	4.0	IMMIGRATION DEPARTMENT	Interacts with 4.2, 9.2, 4.5, 4.4	20
36	4.1	**Immigration administrator office**	Interacts with 4.1, 4.3	32
37	4.2	**Immigration staff offices**	Interacts with 4.2	18
38	4.3	**Storage room for immigration department**	Interacts with 4.1	18
39	4.4	**Immigration department archive**	Interacts with 4.1	10
40	4.5	**Immigration department storage**	Interacts with 3.1, 4.8, 4.10A	10
41	4.6	**Immigration administrator office for arrivals**	Interacts with 4.9A	16
42	4.7	**Immigration office for departures**	Interacts with 4.6	16
43	4.8	**Interrogation room**	Interacts with 4.1, 4.7	15
44	4.9A	**Passport check space for departures/ Phase A**	Interacts with 4.9A	30
45	4.9B	**Passport check for departures/ Phase B**	Interacts with 3.1, 3.4A, 4.6, 4.10B	12
46	4.10A	**Passport check for arrivals/ Phase A**	Interacts with 4.10A	35
47	4.10B	**Passport check for arrivals/ Phase B**	Interacts with 4.2, 9.2, 4.5, 4.4	14
	5.0	CUSTOM DEPARTMENT		
48	5.1	**Custom administrator office**	Interacts with 5.2, 2.1A, 6.8	20
49	5.2	**Custom check office**	Interacts with 5.1, 5.6, 5.7, 5.10, 5.9, 3.5, 5.5	48
50	5.3	**Custom treasurer office**	Interacts with 3.5	12
51	5.4	**Duty-free shop office**	Interacts with 2.3	18
52	5.5	**Computer office**	Interacts with 5.2, 5.6, 5.7	15
53	5.6	**Custom server office**	Interacts with 5.2, 5.7, 5.5	6
54	5.7	**Check office**	Interacts with 5.2, 5.5, 5.6	32
55	5.8	**VAT office**	Interacts with 2.1A	16
56	5.9	**Custom check departures**	Interacts with 5.2, 6.9, 6.5, 6.4, 4.9A, 5.7	30
57	5.10	**Custom storage**	Interacts with 5.2	35
58	5.11	**Money Exchange office**	Interacts with 2.1A	20
	6.0	HARBOUR POLICE		
59	6.1	**Harbour police chief officer**	Interacts with 2.1A, 3.1, 6.2, 6.3, 6.7, 11.9	20
60	6.2	**Harbour police assistant officer**	Interacts with 3.1, 2.1A, 6.1, 6.3	20
61	6.3	**Staff office**	Interacts with 3.1, 2.1A, 6.1, 6.2	16
62	6.4	**Inspector office**	Interacts with 5.9	16
63	6.5	**X-Ray machines**	Interacts with 5.9, 6.9	40
64	6.6	**Interrogation room and wc**	Interacts with 6.7	10
65	6.7	**Detention room**	Interacts with 6.6, 6.1	13
66	6.8	**Narcotics prosecution office**	Interacts with 3.1, 5.1	20
67	6.9	**Bodily control room for departures**	Interacts with 5.9, 6.5	10
	7.0	PORT AUTHORITY		
68	7.1	**Port authority office**	Interacts with 9.4, 1.1, 7.2, 7.3	20
69	7.2	**Meeting room**	Interacts with 1.1, 7.1, 7.3	40
70	7.3	**Security room**	Interacts with 1.1, 7.1, 7.2	20
	8.0	CAR ARRIVALS/DEPARTURES		
71	8.1	**Immigration office for departures**	Interacts with 8.9, 8.2, 8.4	12
72	8.2	**Custom office for departures**	Interacts with 8.9, 8.1, 8.4	12
73	8.3	**Money exchange report office for departures**	Interacts with 8.9, 8.11	10

74	8.4	**Harbour police office for departures**	Interacts with 8.9, 8.1, 8.2	12
75	8.5	**Immigration office for arrivals**	Interacts with 8.11, 8.6	12
76	8.6	**Custom office for arrivals**	Interacts with 8.5, 8.11	30
77	8.7	**Harbour police office for arrivals**	Interacts with 8.11, 8.6	12
78	8.8	**Car parking for departures**	Interacts with Car park entrance, 8.9	50p.
79	8.9	**Car check space for departures**	Interacts with 8.8, 8.1, 8.2, 8.4, Ferries/Departures	60
80	8.10	**Car parking for arrival**	Interacts with Car park exit, 8.11, Ferries/Arrivals	30p.
81	8.11	**Car check space for arrivals**	Interacts with 8.10, 8.5, 8.3, 8.7, 8.6	100
	9.0	OTHER SPACES		
82	9.1	**First aid room**	Interacts with 1.1, 11.8	25
83	9.2	**Rest room**	Interacts with 4.1, 9.4	40
84	9.3	**Storage for shops**	Interacts with 2.3	150
85	9.4	**Staff toilets**	Interacts with 9.8, 3.6, 9.2, 7.1	---
86	9.5	**Passengers toilets**	Interacts with 1.1,9.8	---
87	9.6	**Screens for announcements**	Interacts with 2.1A, 1.1	---
88	9.7C	**Restaurant /Closed space**	Interacts with 1.1, 9.7O	400
89	9.7O	**Restaurant/Open space**	Interacts with 9.7C	400
90	9.8	**Storages** In different parts of the building	Interacts with 9.5, 9.4, 9.5	---
	10.0	MECHANICAL SPACES		
91	10.1	**Computer room**	---	20
92	10.2	**UPS room**	---	15
93	10.3	**Electricity generator room**	---	20
94	10.4	**Mechanical room**	---	250
95	10.5	**EAC room**	---	30
96	10.6	**Electrical panels room**	---	15
97	10.7	**Telephone centre room**	---	10
98	10.8	**BMS check**	---	12
	11.0	PARKING SPACES		
99	11.1	**Vehicle drop-off and pick-up parking**	Interacts with 1.1	20p.
100	11.2	**Taxi drop-off and pick-up parking**	Interacts with 1.1	30p.
101	11.3	**Bus drop-off and pick-up parking**	Interacts with 1.1	20p.
102	11.4	**Staff car parking**	Interacts with 4.1	100p.
103	11.5	**Public car parking**	Interacts with 1.1	600p.
104	11.6	**Carrier parking**	Interacts with 1.3, Ferries/Departures	12p.
105	11.7	**Bus parking for passengers**	Interacts with 2.1A.1, 2.1A.2, 2.1A.3, 2.1A.4, 2.1A.5, 2.1A.6, 3.1	4p.
106	11.8	**Ambulance parking**	Interacts with 9.1	1p.
107	11.9	**Harbour police parking**	Interacts with 6.1	3p.
	12.0	**FERRY TERMINAL TOTAL AREA A+B FACE**		
	12.1	**Ferry terminal Phase A/Area**		6630
	12.2	**Circulation space, toilets, walls etc. 30% Phase A**		1989
	12.3	**Ferry terminal total area Phase A**		8619
	12.4	**Area Phase B**		1200
	12.5	**Total area Phase B**		1560
	12.6	**Total area Phase A+B**		10179

Table 6. 1: Analysis of competition brief based on the original brief [Cyprus Ports Authority, 2003]

Figure 6.27 shows one possible functional diagram of our terminal building including facilities for passengers and staff. It represents the connections between interacting spaces or functions. The different colours represent different functional groups as they are specified in the competition brief. For instance, the Foyer/Passenger hall group is coloured red and is numbered 1, and it shows all the interrelated activities in this area.

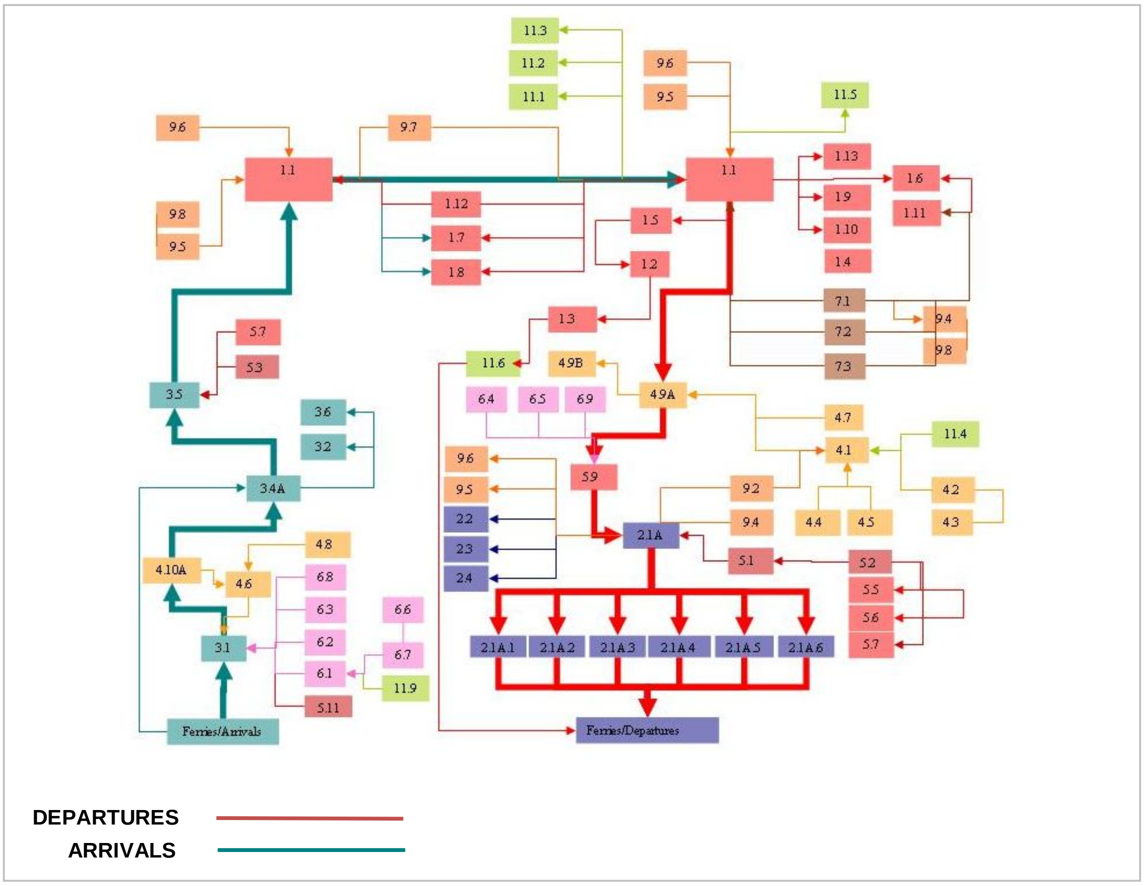


Figure 6. 27: Functional diagram of departure and arrival flows in Limassol ferry terminal case study

Emphasis is given to the passenger functional areas and their connections that form the departure (red thick line) and the arrival (blue thick line) flows. Such areas include check-in, immigration control, and security check points.

6.4.2.1 Organization of route choice strings

We start our case study by assuming that departure and arrival flows will occur in a single area that is divided in three sub-areas according to different functions. Briefly described, area A includes the departures check-in and arrivals exit, area B the immigration control and security check, and area C the departures lounge and arrivals hall.

Figure 6.28 shows a possible outline of the ferry terminal building with the initial location of various activities. Facilities in areas A and C are initially distributed randomly and facilities in area B are fixed.

We expect that passengers will move through different activities and they will possibly intersect each other. Results will help us to re-adjust and find possible locations for each activity within this area.

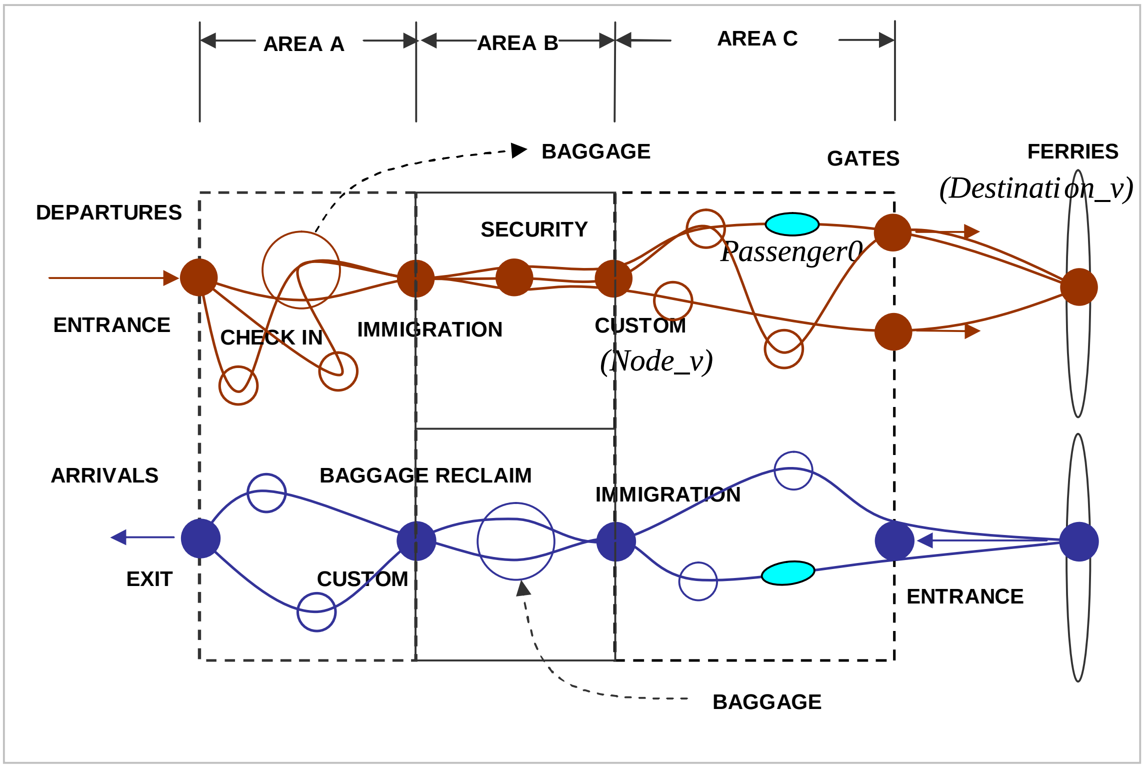


Figure 6. 28: Possible initial location of various facilities and passenger route choice strings

In order to generate path systems for departure and arrival flows, we first need to create route choice strings for each passenger (see figure 6.28). Each passenger is given a route of movement that consists of a number of nodes (red or blue points). Each node represents a functional area: entrance, check-in, immigration, and so on. Nodes are organized in a sequence of movement steps towards their compulsory final destination (red or blue points).

Figure 6.29 shows the actual diagram of passenger route choice strings in relation to the functional areas. It is generated using four route choice strings for departures (red lines) and two for arrivals (blue lines). An example of route choice string for departures is as follows:

1.vehicle drop-off, 2.entrance, 3.foyer, 4.restaurant, 5.check-in, 6.immigration control, 7.security check, 8.custom control, 9.restaurnant, 10.dpearture lounge, 11.gate_3, 12.seat_5.

The spaces that are contained within the black curved lines are the passenger control areas for departures and arrivals that are included in area B (see figure 6.29) and their facilities remain fixed.

The different nodes (functional areas) in each string may influence the passenger movement behavior. For instance, the node that represents the immigration control acts as ‘passing point’. In case of a foyer area, the node is not just a ‘passing point’ but acts as ‘crossing point’ as well since it serves both the arrival and the departure flows.

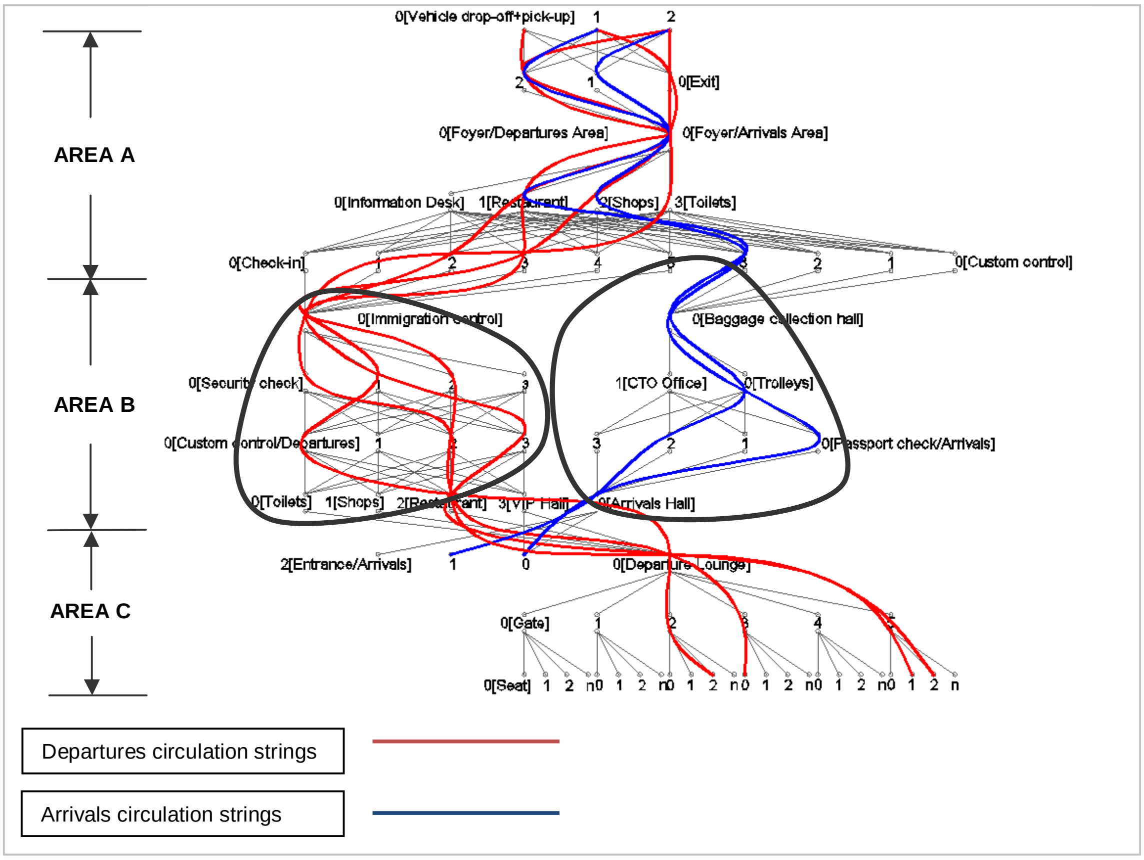


Figure 6. 29: Actual passenger circulation strings for departures (red) and arrivals (blue)

One of the main roles of the modelling is to observe the phenomena that occur in different ‘passing’ and ‘crossing’ points. Then, the circulation problems can be understood, evaluated, and new design solutions can be suggested.

6.4.2.2 Signs and destinations positioning

The configuration of signs and destinations [19] is based on the general passenger circulation tree as it is shown in figure 6.29. Nodes are represented by signs[20] and they are organized in a sequence of movement steps (foyer, check-in, and so on) until the compulsory final destination.

---

[19] The signs and destinations are initially positioned using an input .txt data file. During the simulation the entry and the destination sign points remain fixed. However, the position of all other signs can be changed during the simulation or alternatively can be changed after the program is terminated and the output .dxf file is created. During the simulation this can be done using the mouse function. The computer program gives to architects the ability to use the mouse in order to drag, move and re-arrange the position of signs. Also, the computer program can rewrite these last changes in an output .txt file. Alternatively, this can be done using a .dxf file that can be opened in any CAD software package. Then, the coordinate positions of signs can be written to a .txt file. This procedure is found to be more difficult although useful when the positioning needs to be entirely changed.

[20] In this case, the term sign takes a broader meaning. It may be a landmark, an entrance or a ferry that is translated into a ‘visible clue’ and can be motivate the movement of passengers.

A sign's heading is specified according to different criteria:

i. In the direction of the current destination,

ii. In the direction of a next sign,

iii. To provide an interesting path design, based on the architect's aesthetic criteria.

6.4.2.3 Passenger movement and interaction behaviour

Passenger behavioural rules are divided into passenger movement and interaction rules. Passenger movement rules use the 'Experiment type 1: Each individual has a compulsory destination[21]' (see section 5.6.5.2).

Figure 6.30 shows the initial organization of departure (red) and arrival (blue) circulation strings in the area of investigation together with the departure and arrival entry and destination points. The departure flow has only one entry point that is close to the roundabout area (red point - entry to the port area). Also, the departures have only one final destination point, the ferryboat on the east dock (red point - right hand side).

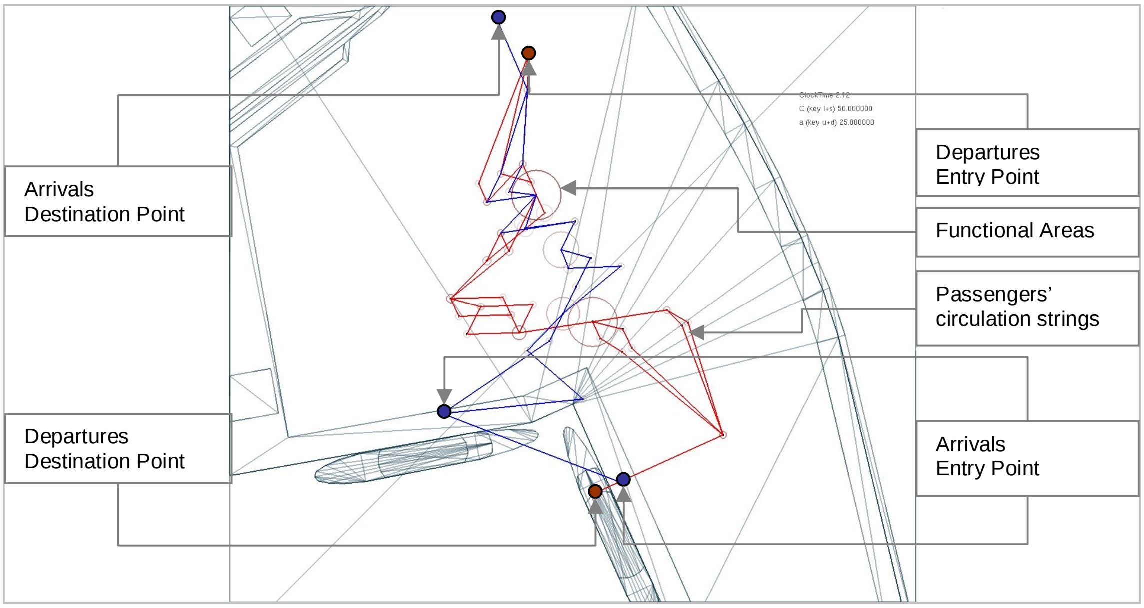


Figure 6. 30: Departures and arrivals destination and entry points

The arrival flow has two entry points (blue points) close to the ferryboats, one close to east dock and one close to north dock, on the right and on the left hand side respectively. Also, the arrivals have a compulsory final destination point (blue point) close to the roundabout.

---

[21] Each passenger has only one compulsory destination. He or she can not select optional destinations from a list since the main design task is the generation of path systems and not the examination of the local passenger behaviour.

Interaction behaviour between the passengers is achieved by applying the repulsive effect. According to this, each passenger is subjected to a virtual repulsive force in order to avoid collision with other passengers.

The number of the passengers and the groups that can be applied are flexible and depend upon different cases. The current simulation consists of four groups of passengers that are distributed in the virtual environment according to the organization of circulation stings.

6.4.3 Initial simulation

Figure 6.31 shows the computer-generated results from the initial passenger movement behaviour simulation[22]. Figure 6.31a shows the initial configuration of design. This includes the passengers' first positioning (red and blue groups), the position of a sign in each red circle (functional areas), as well as the entry the destination points. Figure 6.31b shows the development of the circulation pattern as the result of the passenger movement and interaction behaviour.

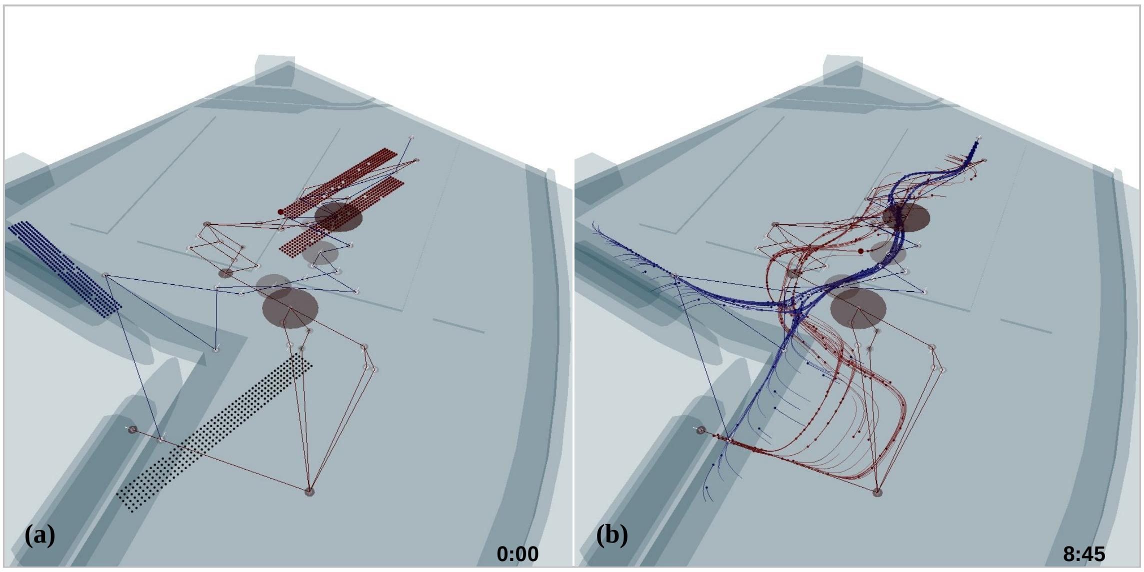


Figure 6. 31: a. Initial configuration and passenger first positioning, b. Generated circulation diagrams for departure (red trails) and arrival (blue trails) flows (see Appendix D.1)

In the first stage of this investigation, the results show possible generated path systems. Basically they represent routes that the passengers may use to reach their destinations. The generation of path systems in the whole area of investigation can be further elaborated and used for the organization of space.

---

[22] Results can be derived while the program runs or using the output drawing files. The passengers leave trails or patches behind them when they move resulting in the generation of the passenger circulation diagrams.

6.4.3.1 Evaluation of the model

Figure 6.32 shows our first attempt to propose a building envelope (black curved line) that is derived from the initial generated path systems. The envelope shows the passenger flow in relation with time in a logical way.

It was found that the building responds to path systems since all passengers need to gather in the area C that includes the departure lounge and the arrival hall. The area A is used as temporary passing area since it serves the departures as check-in lounge and the arrivals as the exit point of the building.

Figure 6. 32: A + B + C areas of investigation (see Appendix D.3)

Figure 6.33 shows the passenger flow rate - time relationship in the area B.

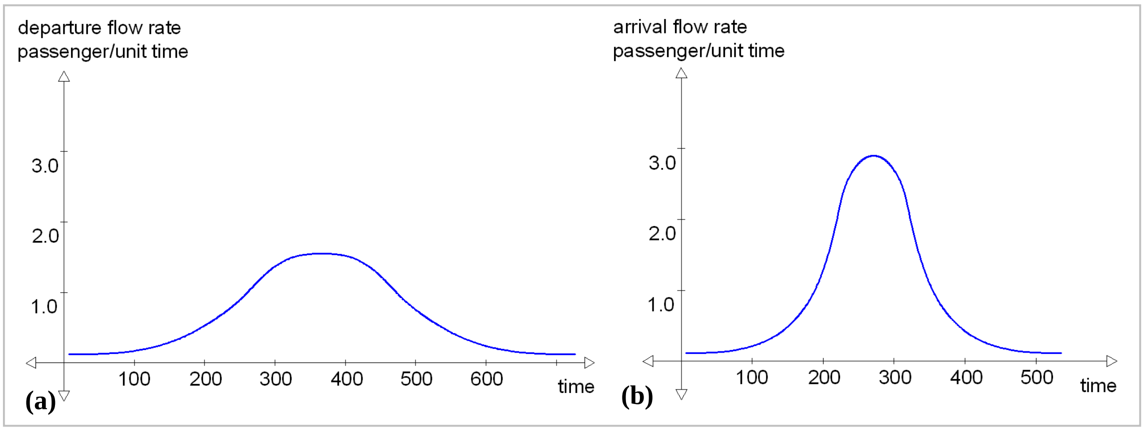


Figure 6. 33: Passenger flow rate - time relationship in passenger control area B

The total area of graphs remains the same and it indicates the total number of passengers who pass from the area B. However, figure 6.33a shows that the departure flow rate distribution occurs over a longer time period because passengers for departures enter and pass through this area over a flexible time. The arrival flow rate distribution (figure 6.33b) occurs over a shorter time period because passengers pass the control point as groups within a specific short time period.

The initial simulated results in areas A and C of intersection between the two passenger flows (see figure 6.32) show that these do not satisfy the passenger movement criterion of effective flow. The intersections might cause problems in the movement of passengers towards the city (area A) and towards the port terminal building (area C).

Figure 6.34 shows a zoom in on the area A where the arrangement of functions in the foyer area[23] effects the movement of passengers. It is obvious that the main reason is the unclear distinction between the entrances and the exits for departures and arrivals respectively.

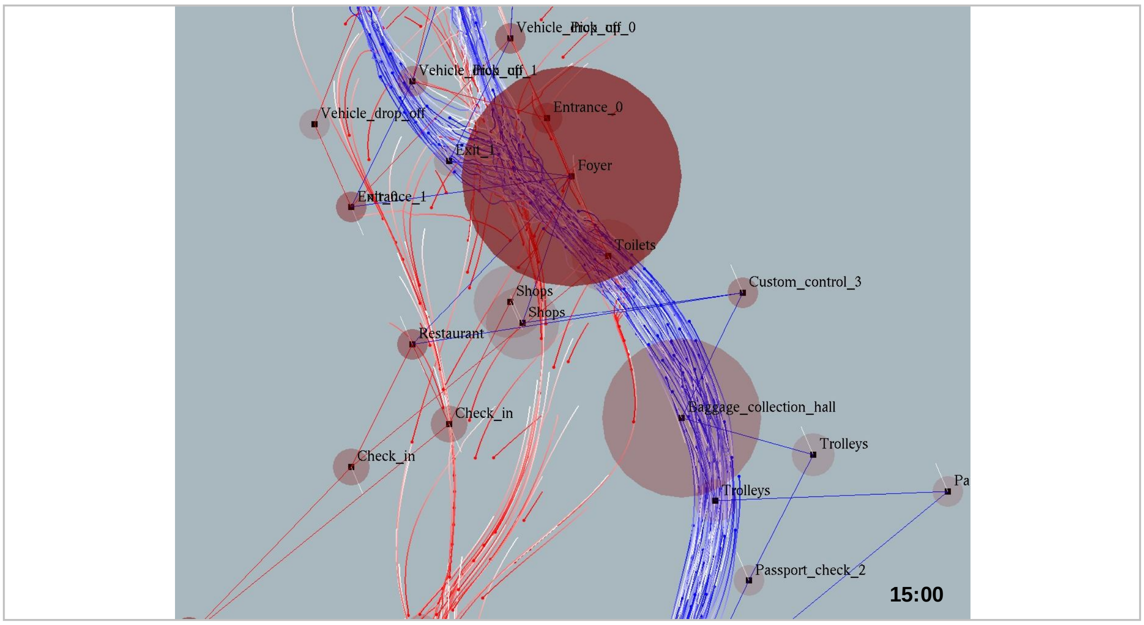


Figure 6. 34: Area A of passenger intersection after zoom in (see Appendix D.5)

Figure 6.35 shows a zoom in on the area C close to the ferryboats. Clearly, the initial organization of the arrival hall and departure lounge causes problems of intersection since the passenger motion towards (departures) and from (arrivals) the ferryboats are not clearly distinguished. A clear distinction could be the arrangement of the arrival

---

[23] According to the competition brief, the foyer in the area A is a large space that can be used for both, the departures and the arrivals. In this space the entrances and the exits are included however a clear distinction between them is not described explicitly in the brief. In the initial design configuration, the foyer has been specified as common space where the departure and the arrival flows can meet. Also, the entrances and the exits are the passing points for both the departure and the arrival flows.

flow in the north dock (left hand side) and the departure flow in the east dock (right hand side) however the competition brief does not allow such separation[24].

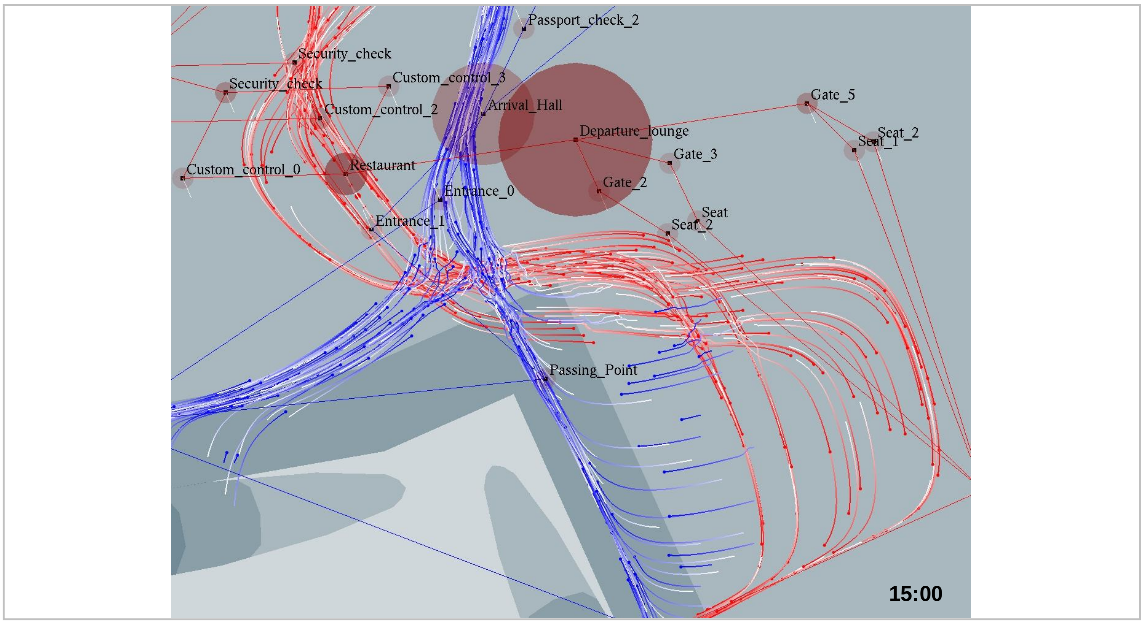


Figure 6. 35: Area C of passenger intersection after zoom in (see Appendix D.6)

The area B of the model was found to be satisfactory mainly because the passenger flows are not mingled since control points are clearly distinguished[25] in departures and arrivals.

6.4.3.2 Suggested solution

In order to eliminate the movement problems that may occur, a number of other possible other passenger movement models were investigated.

The problem of the intersecting passenger movement in the area A can be eliminated or drastically reduced by re-locating the entrances and the exits[26].

---

[24] According to the competition brief, the arriving and the departing ferryboats can use either the north or the east dock. This may result in passenger cross movement where the departure and the arrival flows are operating at the same time. Such phenomena can occur during the peak traffic period of the summer.

[25] Such distinction is one basic requirement of the competition brief and it needs to be clearly specified in the proposed design. Also, the separation of flows may provide design solution with additional advantages. One such advantage is the generation of a common space between the two flows, which can serve as the common area used by the port authority or the building's staff. This area may include the immigration department for both the departures and the arrivals.

[26] The entrances and the exits need to be separated in order to avoid unnecessary collision between departure and arrival flows.

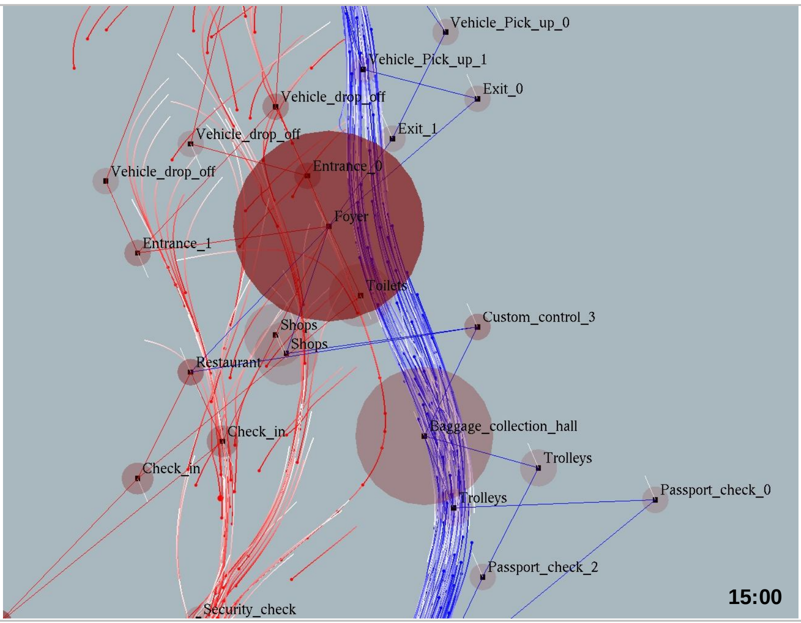


Figure 6. 36: Solving the passenger intersection problem in area A (see Appendix D.11)

Figure 6.36 shows a possible solution, locating the entrances on the left side and the exits on the right side of foyer. This allows the effective passenger flow in departures and arrivals since the movement is not interrupted by intersections between the two passenger groups.

The intersection of passengers in the area C (see figure 6.35) can be avoided if the departure and the arrival flows are not at the same level. It is suggested that the area of departures (departure lounge and gates) be located at the first floor level and the area of arrivals (arrival hall entrances) in the ground floor. This will provide a smooth flow of passengers and possible intersections can be avoided. One reason for having departures on first floor and departures on ground floor is that people expect to wait for longer periods in departures and therefore views out of windows are more important.

6.4.4 Other possible solutions

A number of other solutions can be derived by modifying the parameters in the computer model. These may include the number and the location of the passengers' entry points, the number and the location of passengers' destination points, the number of passengers, the time of appearance, the number and the location of the signs or functional areas and so on.

Apart from these parameters, the variables[27] in the computer model describing passenger behaviour will influence their behaviour and hence the generation of the circulation diagrams. Such variables include among others, the sign strength C_{parallel}, described in section 5.6.3.2, which can affect the curved line of movement in relation to the sign position. At a global level, the sign strength C_{parallel} can influence the general appearance of the diagrams. Also, the variable $a_{\text{interperson repel}}$ influences the repulsion between the passengers.

6.4.4.1 Sign strength C_{parallel}

Figure 6.37 shows two circulation diagrams that can be generated by changing the variable C_{parallel}. In this case variable C_{parallel} takes values ranged from $\text{C}_{\text{parallel}} = 5$ force units to $\text{C}_{\text{parallel}} = 25$ force units.

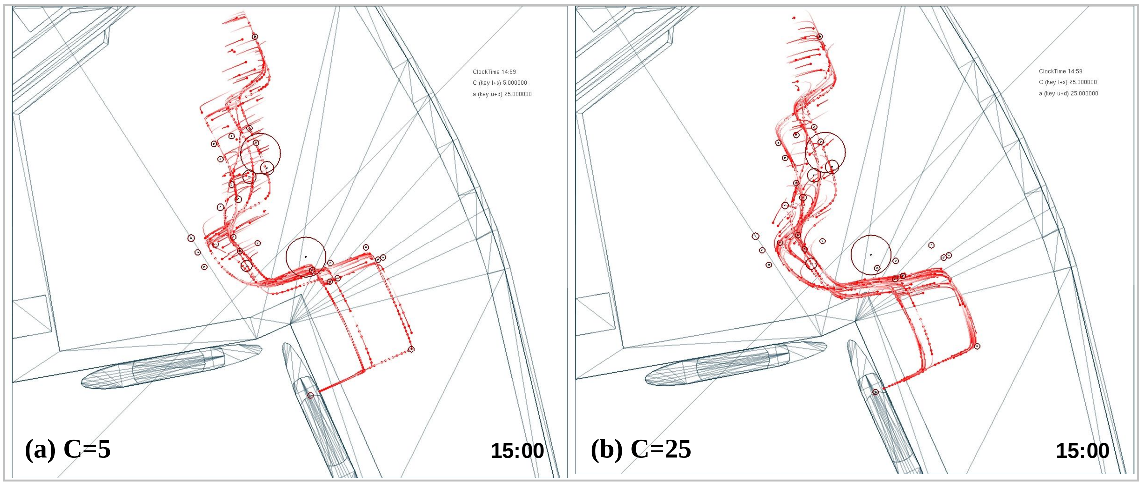


Figure 6. 37: Passenger movement curve line by changing sign strength C_{parallel} [a. $C_{\text{parallel}} = 5$, b. $C_{\text{parallel}} = 25$]

It can be seen that values C_{parallel} close to 0 ($C_{\text{parallel}} = 5$) can motivate people to move perpendicularly towards the signs and hence the generated circulation diagrams have shapes close to rectilinear forms (see figure 6.37a).

---

[27] Variables can remain constant or can be changed during the simulation by using the key function. The modification of the parameters may influence the general passenger behaviour. Also, the changes may affect the density of passengers or the form of generated circulation diagrams.

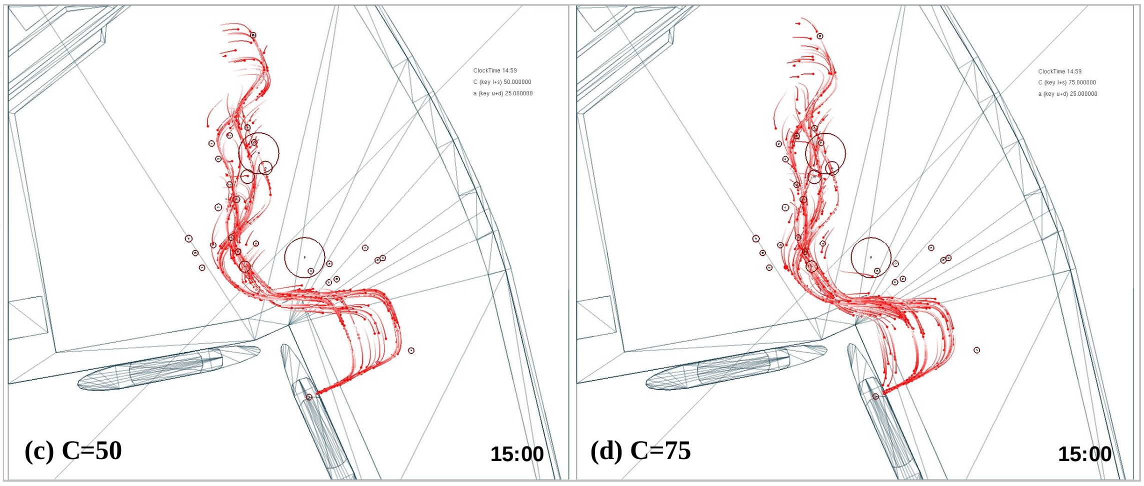


Figure 6. 38: Passenger movement curve line by changing the value of sign strength C_{parallel} [c. $C_{\text{parallel}} = 50$, d. $C_{\text{parallel}} = 75$]

On the contrary, by applying $C_{\text{parallel}} = 75$ force units, the generated circulation diagrams are represented as curve lines (see figure 6.38d).

6.4.4.2 Variable $a_{\text{interperson repel}}$

Changes of the variable $a_{\text{interperson repel}}$ influences the repulsive distance between the passengers. A large value of the variable $a_{\text{interperson repel}}$ tends to separate the passengers and this can also influence the formation of the circulation diagrams. By applying a large $a_{\text{interperson repel}}$ value, the diagrams become diffuse, and applying a small value the passenger paths accumulate.

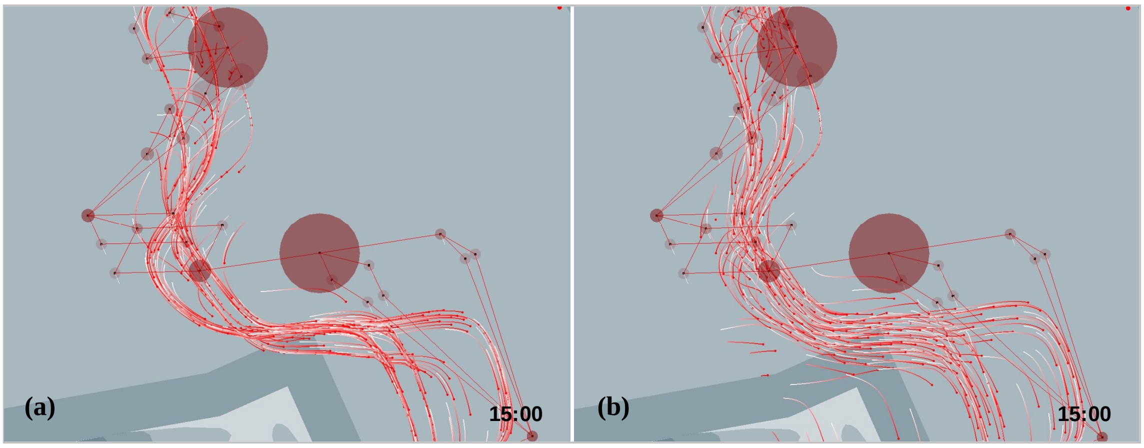


Figure 6. 39: Changes in variable $a_{\text{interperson repel}}$ influence repulsive distance between passengers [a. $a_{\text{interperson repel}} = 25$, b. $a_{\text{interperson repel}} = 50$]

This is illustrated in figure 6.39. Figure 6.39a shows the pattern that is produced when $a_{\text{interperson repel}} = 25$ length units. Figure 6.39b shows the circulation pattern when

$a_{\text{interperso n repel}} = 50$ length units. In this case the distance between the passengers is increased and the passengers become diffused.

The accumulation or the diffusion of the passengers in relation to the circulation diagrams can influence the dimensions of corridors. Thus, the variable $a_{\text{interperso n repel}}$ can be used to specify the width of corridors. Issues related to the actual distances between the passengers will be investigated in meso-scale and micro-scale studies.

6.4.5 Outer loop in Design Framework

The implementation of computer program in all cases (macro, meso and micro-scale) is based on the structure of our Design Framework. Macro-scale study uses the outer loop mechanism of the Design Framework as shown analytically in figure 6.40.

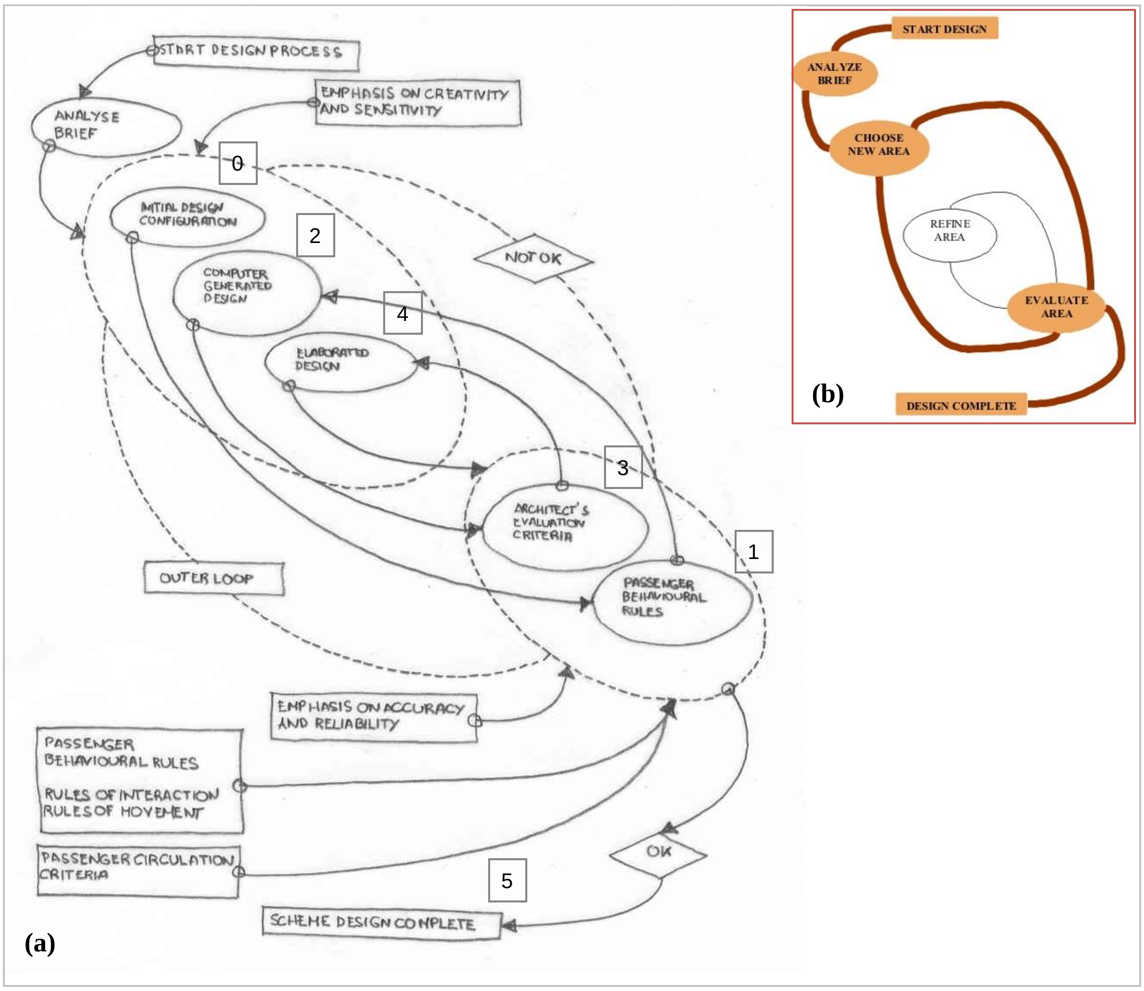


Figure 6. 40: a. Outer loop of Design Framework, b. Design Framework

The numbers indicate the steps that might be followed in this procedure. First step (0) is the initial configuration that is part of creativity and sensitivity. Then, in the evaluation part the passenger movement rules are applied (1), which can help architects to navigate the initial configuration towards better areas in the design space (2). Then, the computer-generated design output (2) is again evaluated based on the architect's circulation criteria (3), and an elaborated design is formulated in the creativity part (4). Finally, if the elaborated design satisfies the criteria of accuracy and reliability, the design is completed (5). If not, the process follows the same loop mechanism until the design satisfies both the computer and the human circulation criteria.

6.5 Meso-scale: Functional areas and circulation design

At the meso[28]-scale, we examine analytically the relation between functional areas and the design of circulation diagrams. The level of investigation is ‘detailed’, thus ‘localized’ models are developed. The ‘localized’ models zoom in on specific areas of circulation design that have been generated at the macro-scale.

- **Analytic design tasks**

The new study will isolate a single area of the scheme design. The selected area is the foyer[29] that serves both the departure and the arrival flows. This area contains activities and facilities including check-in, information, shops, and restaurants. The complex system of movements may result in circulation problems such as crossing points or two opposite flows.

The meso-scale study will suggest circulation diagrams, functional configuration, and plan design that solves in satisfactory way the complex movement and circulation problems in the foyer area.

- **Computational tasks**

The computer program will again be used as a design generator for circulation diagrams. However, the ‘detailed design’ also presupposes further investigation of other important design issues.

In this case the role of computer is to offer satisfactory circulation diagrams and to drive further the design decision-making. Thus, the computer models will attempt to provide other information, not only at the design level but also quantitative data of the passenger movement.

- **Known information**

Known information includes:

i. The data and the ‘scheme design’ that have been derived from the macro-scale including the location of functional areas and the general passenger flow.

---

[28] From Greek μέση meaning ‘middle’.

[29] The competition brief defines the foyer or the passenger hall as: a single space for passengers and public where check-in-starts (departures). Also, the last space before passengers exit to the city (arrivals). Visitors must find their way easily inside the space. It must be clear division between the movements of the passengers who depart and passengers who arrive but not physical division. Passengers who arrive must be directed easily to the exit. Additional functions such as coffee shops, restaurants, banks, and shops are inside the space in order to give liveliness.

ii. The functional areas are updated by adding new areas that influence the passenger movement like the vehicle drop-off and the pick-up points, and the entrances and the exits. Their positioning is based on actual space requirements in square meters.

iii. Passenger interaction rules that are based on the repulsive effect.

iv. The passenger route choice rules like the sign effect and the individual's list of compulsory and optional destinations.

- **Evaluation of the results**

The results will be evaluated based upon:

i. The effective flow through different functional areas and spaces in order to avoid intersecting flows, blockage of movement, and so on.

ii. The efficient passenger movement in different functional areas in order to avoid spending time or wandering around aimlessly inside the building.

6.5.1 Initial organization of design in foyer area

The foyer is the first and the last space that is reached by the departure and the arrival flows respectively. It consists of different functions that can make the space lively much of the time, especially in the peak traffic hours.

Figure 6.41 shows a 'localized' functional diagram that represents all the possible connections between the functional areas in the foyer.

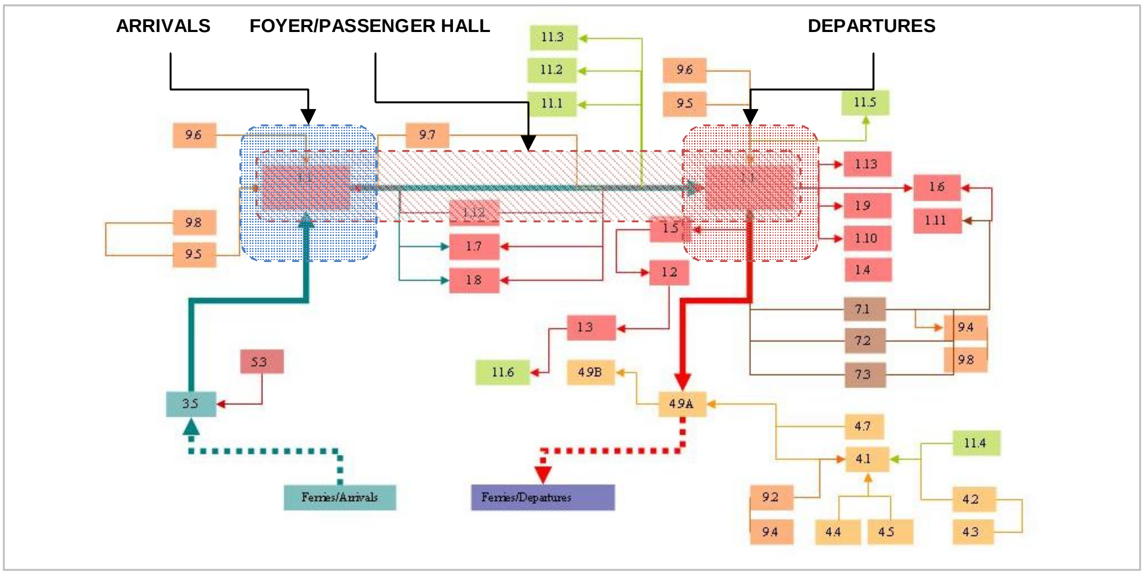


Figure 6. 41: Flow of functional diagram in foyer area (see table 6. 1)

The foyer area is described as a single space, which is divided in two sub-areas that represent the departures and the arrivals. The foyer is coloured red and is marked as 1.1. The departure and the arrival flows are represented by red and blue thick lines respectively. The foyer is directly connected with other facilities like the check-in, the shops, the restaurants, and the cash point machines.

6.5.1.1 Organization of activity destination points

We start our analytical examination of the functional configuration and the passenger movement behaviour in the foyer area assuming that passengers (departures and arrivals) can move in the single area A as it was defined in macro-scale study.

Figure 6.42 shows an outline of the foyer area with possible locations for various activities. The location of activities like check-in and restaurant can be modified in order to satisfy functional and passenger movement criteria. However, activities like immigration/passport and custom control points remain fixed.

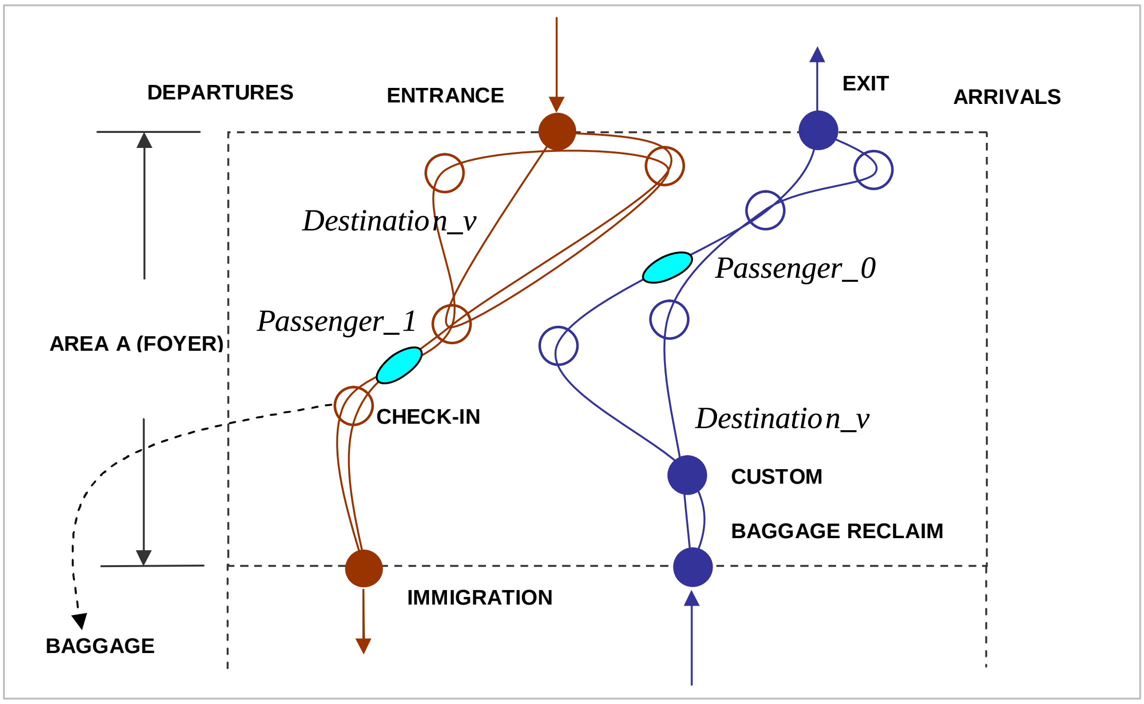


Figure 6. 42: Possible initial location of various activities and destination points

Passenger movement behaviour[30] is generated using activities as destinations (red and blue points). This is different from the macro-scale study in which the movement was

---

[30] It is well known that the human movement behaviour is influenced by the type of building itself. For instance in museums visitors can be attracted towards the artworks following a specific routing that is provided in a map or is randomly selected according to personal desire and choice.

‘linear’ (sequence of signs and compulsory final destination). At the meso-scale emphasis is given to the simulation of ‘natural’ passenger movement[31].

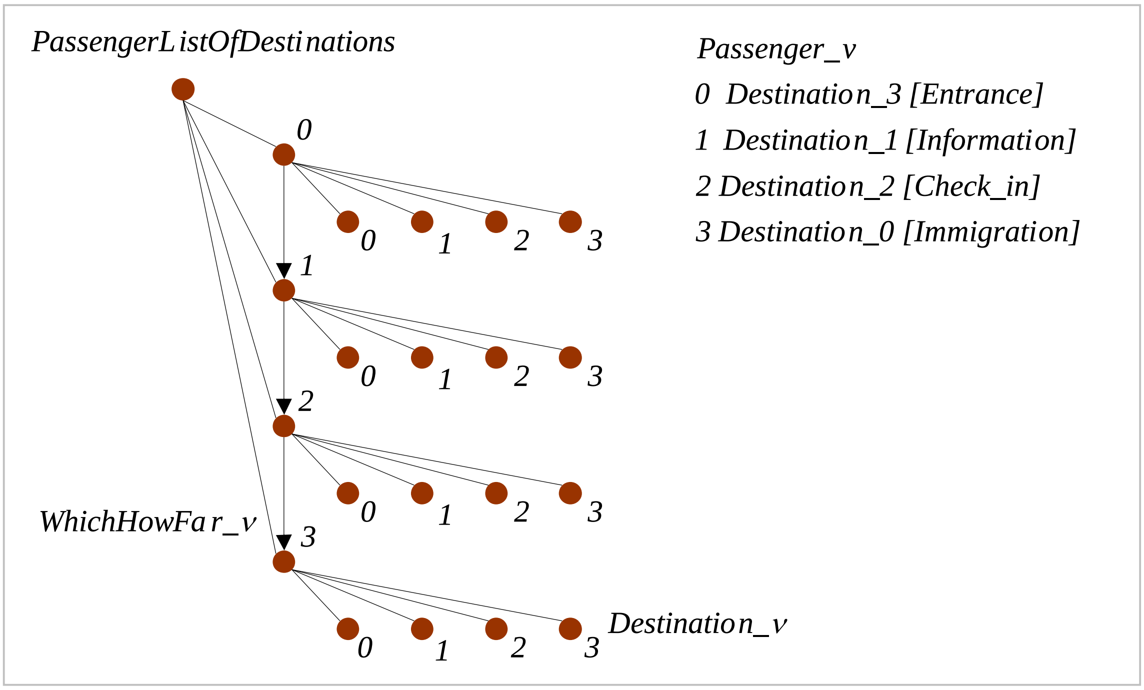


Figure 6. 43: Diagrammatic representation of passenger list of destinations

Figure 6.43 shows a possible diagrammatic representation of ‘natural’ movement in the form of list of destinations for one passenger in departures. The list of destinations for each passenger is based on the hierarchical decision plan that was described as the ‘Experiment type 3: individual’s list of compulsory and optional destinations’ in section 5.6.5.2.

6.5.1.2 Signs and destination positioning

Figure 6.44 shows two possible sign positioning and heading configurations in relation to the defined destination points. Figure 6.44a shows the distribution of the signs in a grid and figure 6.44b shows the signs that are distributed randomly. In the random distribution the signs are accumulated towards the centre and defused towards the boundaries of the area under investigation. The distance between the signs and hence their density may also influence their ability to attract and direct passengers.

---

[31] In ferry terminals the main functional purpose is to organize the departure and the arrival flows. For this reason, the passenger route choice includes compulsory and optional destination. For instance, a passenger from the departure flow needs to pass from the foyer entrance that is compulsory destination and then can optionally select a list of activities like the coffee-shop and the information point. Then, he or she can move to the check-in point and finally he or she passes from the immigration control point that is compulsory destination choice.

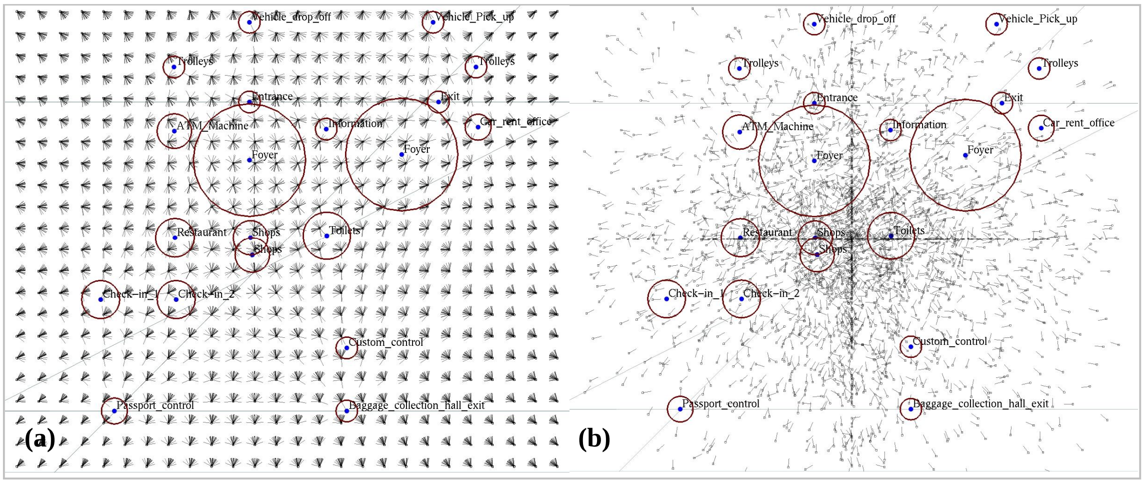


Figure 6. 44: Signs and destinations positioning

The destinations (red circles) are positioned[32] according to the functional areas and they act either as simple ‘passing’ points (entrance and exit destinations) or as ‘crossing’ points (foyer destination).

6.5.1.3 Passenger movement and interaction behaviour

Based on the individual’s list of compulsory and optional destinations technique, our actual passenger movement scenarios are formulated as follows:

Figure 6.45 shows the initial configuration and numbering of destinations. Also, it shows the entry and destination points for departures and arrivals.

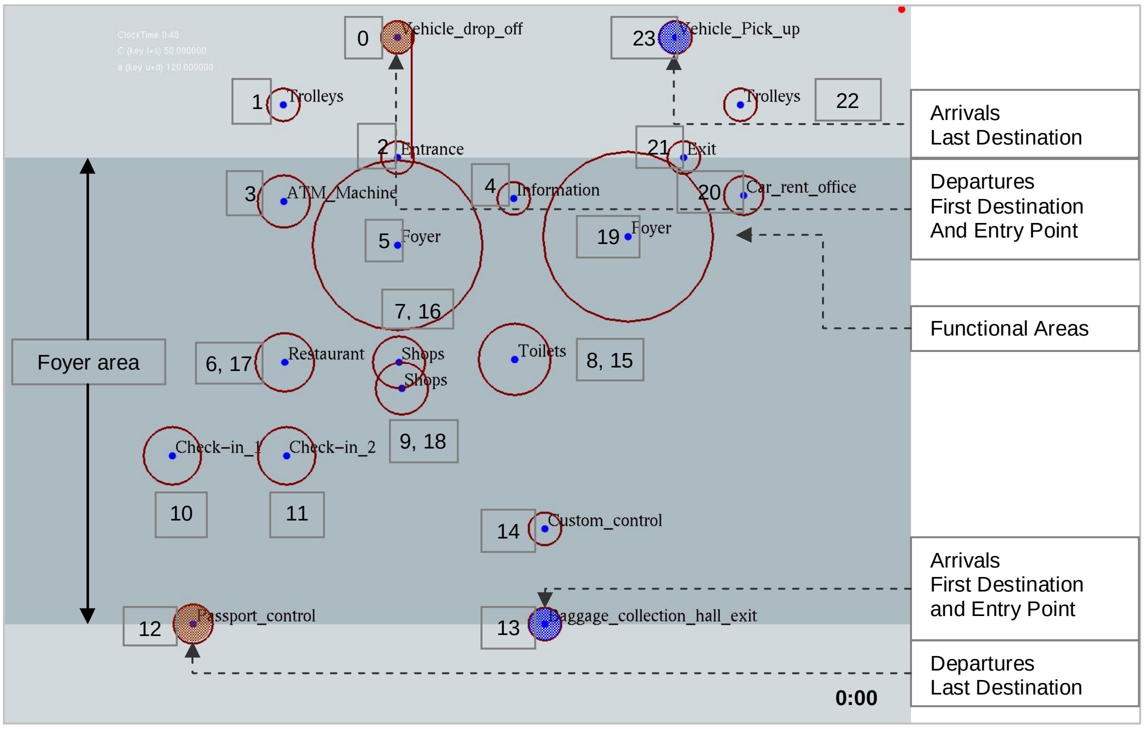


Figure 6. 45: Initial configuration and numbering of destinations in foyer area

---

[32] The destinations can be initially distributed using an input .txt file. Their position can be changed in real time or after the program is terminated and a .dxf file is produced (see section 6.5.2.2).

Figure 6.46 shows the graphical representation for the departure flow movement scenario. Destinations are as follows: 0.vehicle drop-off, 1.trolleys, 2.entrance, 3.atm machines, 4.information, 5.foyer, 6.restaurant, 7.shops, 8.toilets, 9.shops, 10.check-in_1[33], 11.check-in_2, and 12.immigration[34].

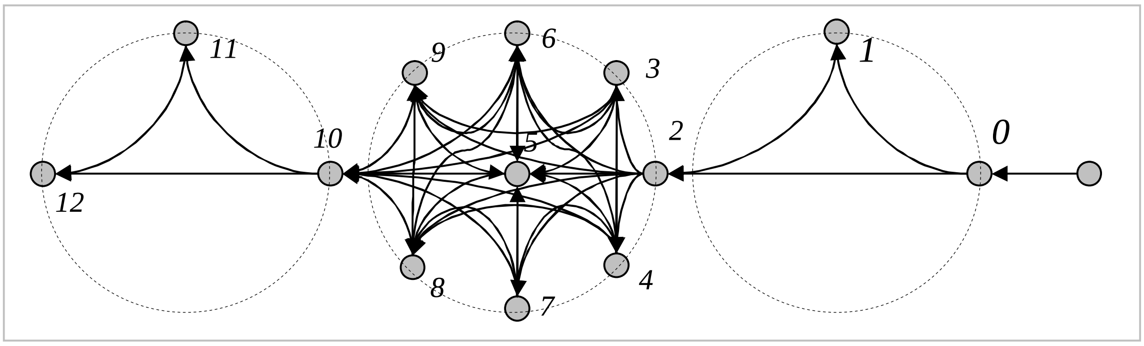


Figure 6. 46: Departure flow destination diagram

Departure flow has one entry close to the vehicle drop-off point. The immigration control point is the last destination.

Figure 6.47 shows the graphical representation of the arrival flow movement scenario. Destinations are as follows: 13.baggage collection hall exit, 14.custom control (last control point), 15.toilets, 16.shops, 17.restaurant, 18.shops, 19.foyer, 20.car rent office[35], 21.exit, 22.trolleys, 23.vehicle pick-up.

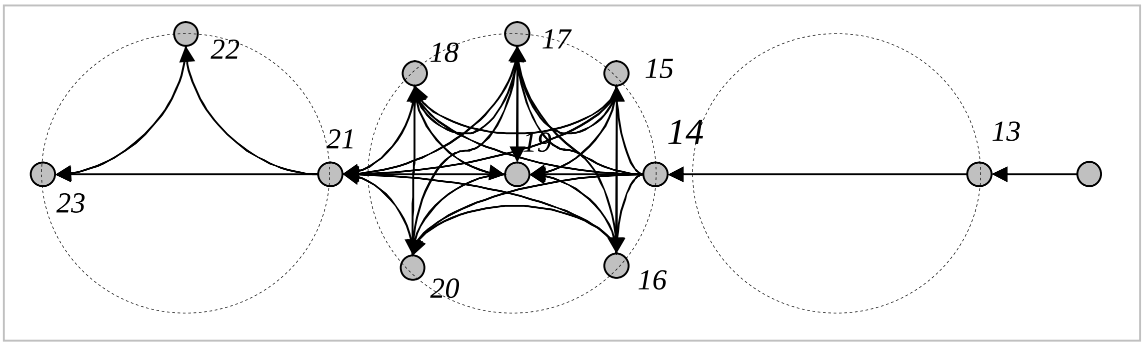


Figure 6. 47: Arrival flow destination diagram

The arrival flow has one entry point close to the baggage collection hall exit. Vehicle pick-up point is the last destination.

In both movement scenarios, we assume that passengers in each group have the same entry and last destination points. However, another possibility is to have a random entry

---

[33] The space for the check-in is separated from the rest of the foyer in order to be accessed just from the departure flow.

[34] Immigration control point for the departure flow is between the foyer and the departure hall and preferably has no optical connection with the foyer area.

[35] Passengers can choose their transportation, rent a car or buy bus ticket.

and last destination points for each passenger. Also, their number and time of appearance may vary as well.

In order to capture the collision avoidance between passengers, the repulsive effect is applied. Either for movement or for interaction behaviour, various parameters[36] can be modified to investigate different scenarios.

6.5.2 Initial simulation

Figure 6.48 shows possible computer generated circulation patterns[37] together with the initial configuration that is based on the data derived from the macro-scale study. The red circles represent functional areas in actual square meters. Also, figure 6.48 indicates the first positioning of the departure flow on the top (red points) and the arrival flow on the bottom (blue points) of the design scheme.

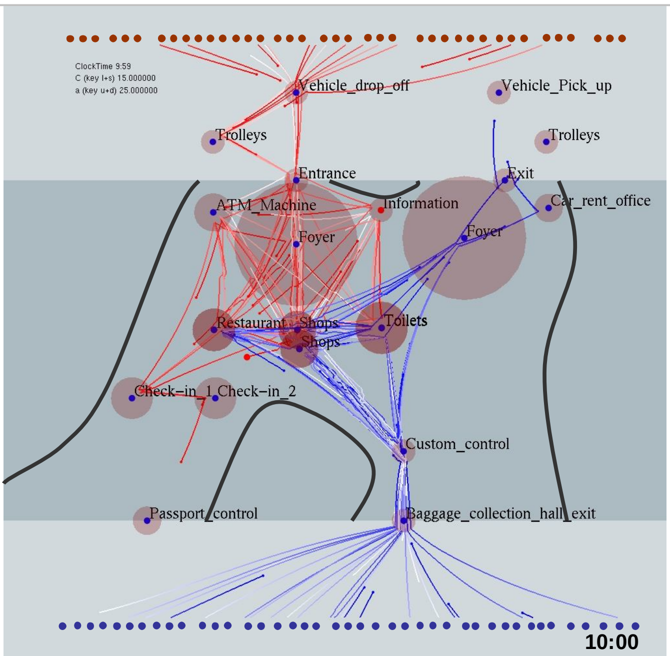


Figure 6. 48: Initial configuration of functional areas and computer-generated circulation diagrams (see Appendix E.1)

What can be initially derived from the first simulation at a global level is the possible connection tendency between different destinations in the form of circulation diagrams.

---

[36] The selection of variables may influence the passenger behaviour and hence the design representation. For instance, the repulsive distance between the passengers (variable $a_{\text{interperson repel}}$) can be changed, either by modifying the program or in real time. Also, passenger behaviour is influenced by other factors such as the sign positioning which in this case is random. Also, sign strength C_{parallel} influences the curve line of movement towards the signs. In this case it has been chosen $C_{\text{parallel}} = 15$ force units.

[37] The visual representation of diagrams can be achieved either using trails or patches. The trails give an idea of possible connections between the functional areas and can suggest further design development around these circulation diagrams. The patches provide an idea of the passengers' area occupation and space requirements.

6.5.2.1 Evaluation of the model

In the evaluation stage, the results are examined based on different movement criteria that have been specified in section 6.3.2. These include the effective passenger flow, the efficient movement and the aesthetic criteria of a desirable design scheme.

Figure 6.49 shows the results that were obtained from the initial simulation stage where the departure flow trails are coloured red and the arrival flow trails are coloured blue. Also, it shows our attempt to create a building envelope that is based on the circulation diagrams.

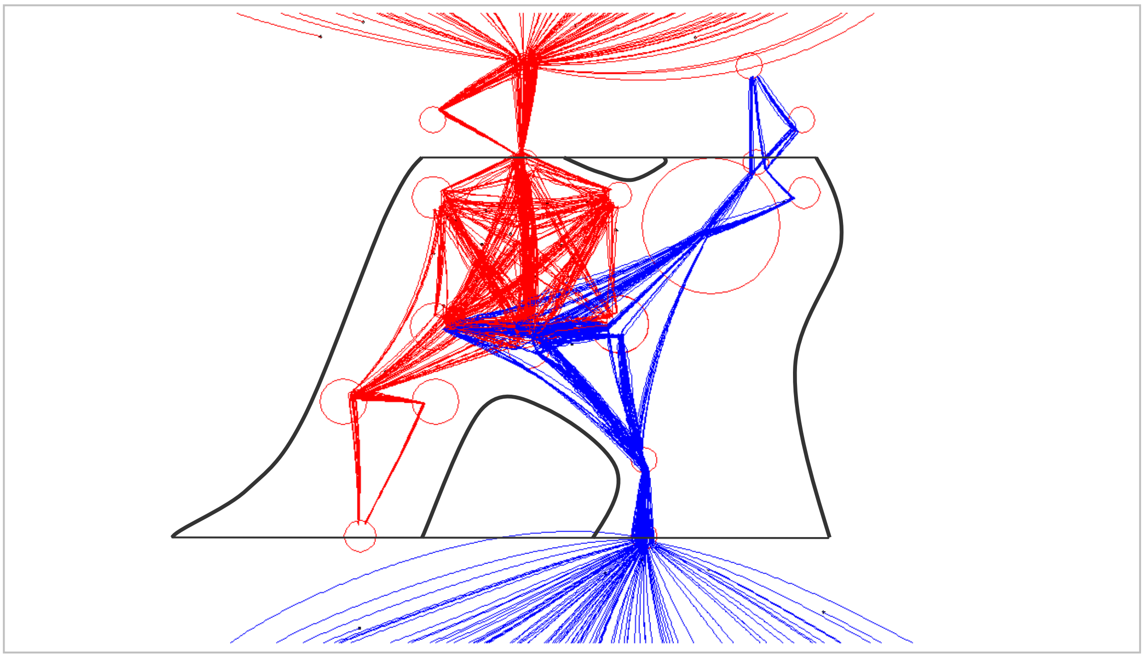


Figure 6. 49: Departures and arrivals passenger flow diagrams

First observations show that:

i. The foyer area can be separated into the departures and the arrivals sections though it can be used for both flows by applying different route choice behaviour.

ii. The use of space in the departures section is more intense and complex than in the arrivals section.

iii. The common activities (restaurants, toilets, etc) seem to cause increasing complexity of movement.

- **Detailed analysis of passenger behaviour**

Figure 6.50a shows the analytical diagram of the departures flow behaviour (red trails).

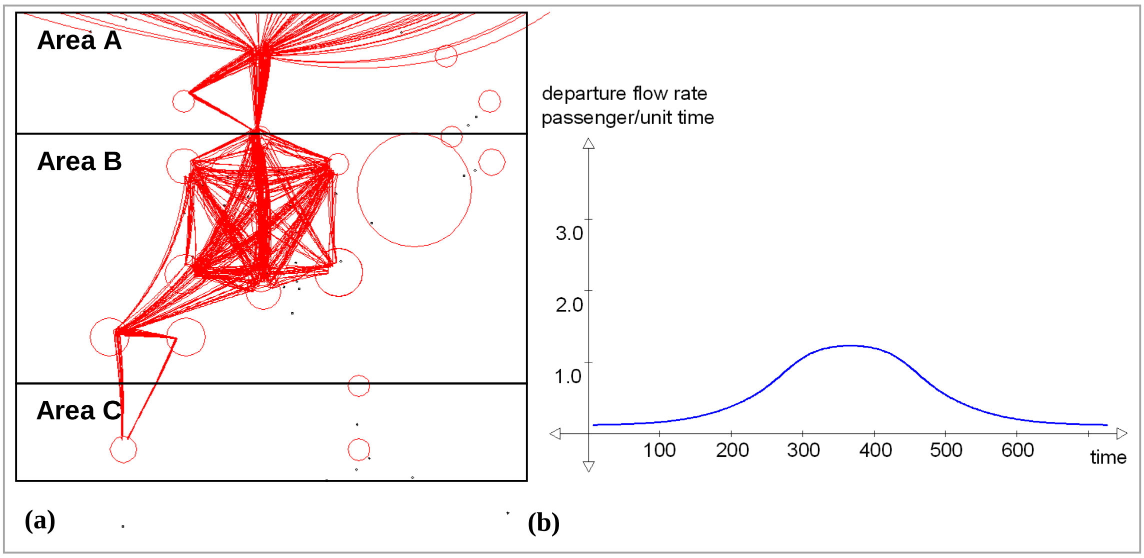


Figure 6. 50: a. Initial generation of departure flow diagram, b. Departure flow rate – time relationship in area C

It can be observed that in the area B (foyer) the density of movement is high since passengers select to move towards different activities. This specific configuration of functional areas allows the use of the entire space. The common activities (restaurant, toilets, etc.) attract large number of passengers. The density of movement in area C (immigration control) is low since the passengers enter the foyer at different times (different route choice behaviour). The passenger flow rate – time relationship in this area is shown in figure 6.50b.

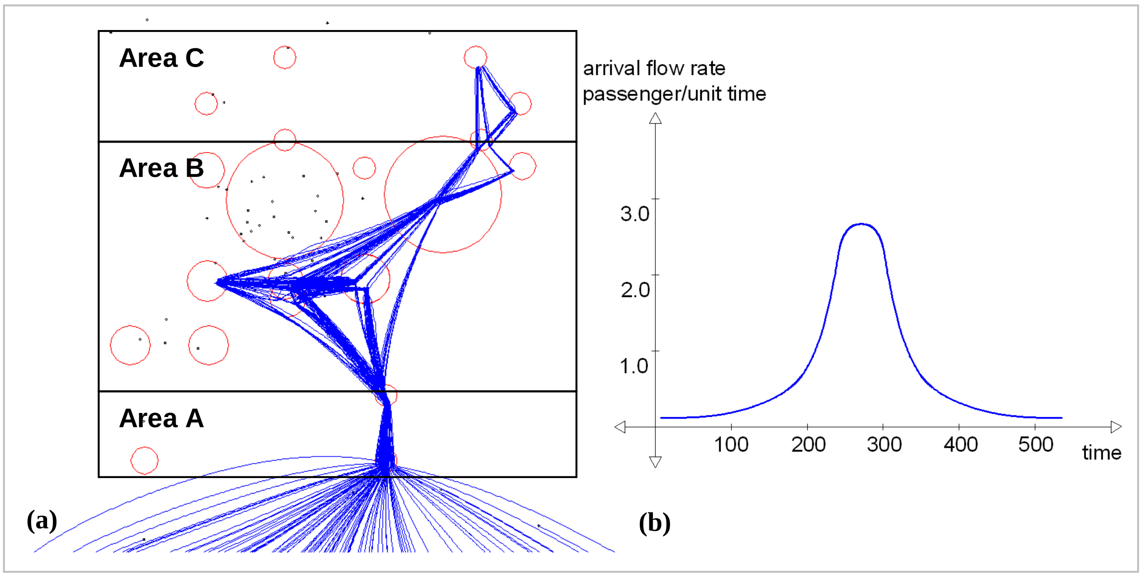


Figure 6. 51: a. Initial generation of arrival flow diagram, b. Arrival flow rate – time relationship in area A

Figure 6.51a show the analytical diagram of the arrival flow behaviour (blue trails). It can be observed that the density of movement is high in the custom check point (area A) since the passengers arrive more or less at the same time. The arrivals flow rate can be described as shown in figure 6.51b. In the area B a large number of passengers move towards common activities (shops, restaurant, etc) and gradually they are directed to the arrival foyer area and then to the exit.

- **Common activities in the area B**

Figure 6.52 shows that the common activities (restaurant, toilet, etc.) in the area B are highly used from both flows[38]. Also, their configuration may not satisfy the circulation criteria of the effective flow and the efficient movement. Parts of the model show obstructions of flows and intersection phenomena. Also, this arrangement causes blockage of passenger movement and of their further movement towards their destinations.

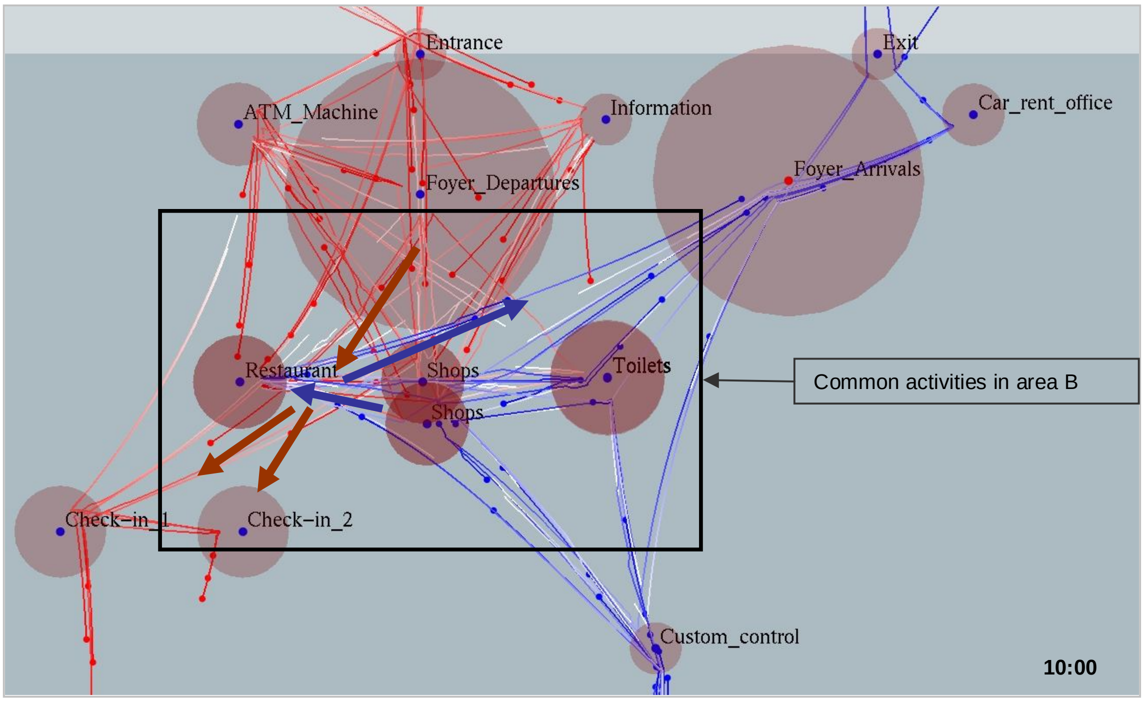


Figure 6. 52: Common activities and circulation diagrams in the area B (see Appendix E.2, E.3 and E.4)

The zoom in on the common activities (see figure 6.52) shows that the location of the restaurant in the left side of foyer cause blockage and intersection phenomena between arrival and departure flows. Specifically, the movement of departures (red arrow)

---

[38] The area of investigation is located between the foyer (departure and arrivals) and before the passport/immigration control for the departures and after the custom control point for the arrivals.

towards the check-in points is obstructed by the movement of arrivals (blue arrow) towards the restaurant area in both directions.

- **Other observations**

Other parts of the foyer area are found to be satisfactory mainly because there is a clear distinction between the departure and the arrival flows. Generally it can be observed that:

The foyer's use for departures is busier than that for arrivals due to the type of activities that are included. The area is mainly used as waiting space for passengers before trips, thus it provides all facilities for making the waiting time comfortable and pleasing. Also, the departure flow speed is slow.

The foyer's use for arrivals functions in completely different way. It is the last destination before the exit that leads to the city. For some passengers it is the end of their trips and for some others in the beginning of their tour to the city. Thus it serves as a 'passing' point and not as a place for rest. This might influence the passenger flow rate and hence the proportion of space that is given for each flow[39].

6.5.2.2 Suggested solutions to improve design configuration and flow

Passenger flow problems occur due to the initial configuration of functional areas. In order to eliminate possible intersections between the departure and the arrival flows suggestions are made including new configuration and size of existing functional areas[40] as well as splitting existing areas or adding new areas.

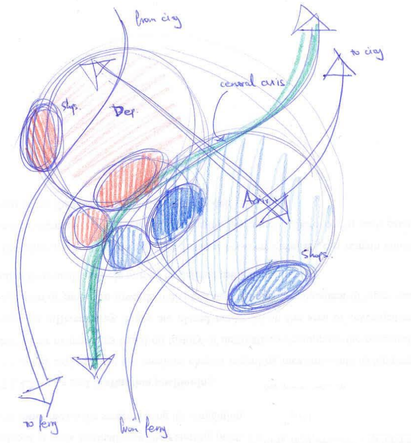


Figure 6. 53: Diagrammatic organization of functional areas

---

[39] Initially it has been assumed that the foyer area in square meters has to be equally shared. However, further design must take into account the proportion of space that can be used for the two different flows.

[40] According to the competition brief requirements, the positioning of restaurants, shops or other common facilities must be located in a central point in order to serve both flows. Also, the space must be clearly divided, though it is suggested to avoid physical separation.

Figure 6.53 shows the diagrammatic organization of functions in the new design. It is based on the idea of a central axis that allows departure and the arrival flow without any intersection.

Figure 6.54 shows the virtual organization of functional areas and the new computer generated circulation diagrams. Functional areas are located on both sides of the central axis. The axis separates and also organizes the restaurants[41] , the toilets, and the shops[42] that serve departure and arrival flows. The re-location[43] of functions (destinations) and the computer generated diagrams can be achieved in parallel and in real time.

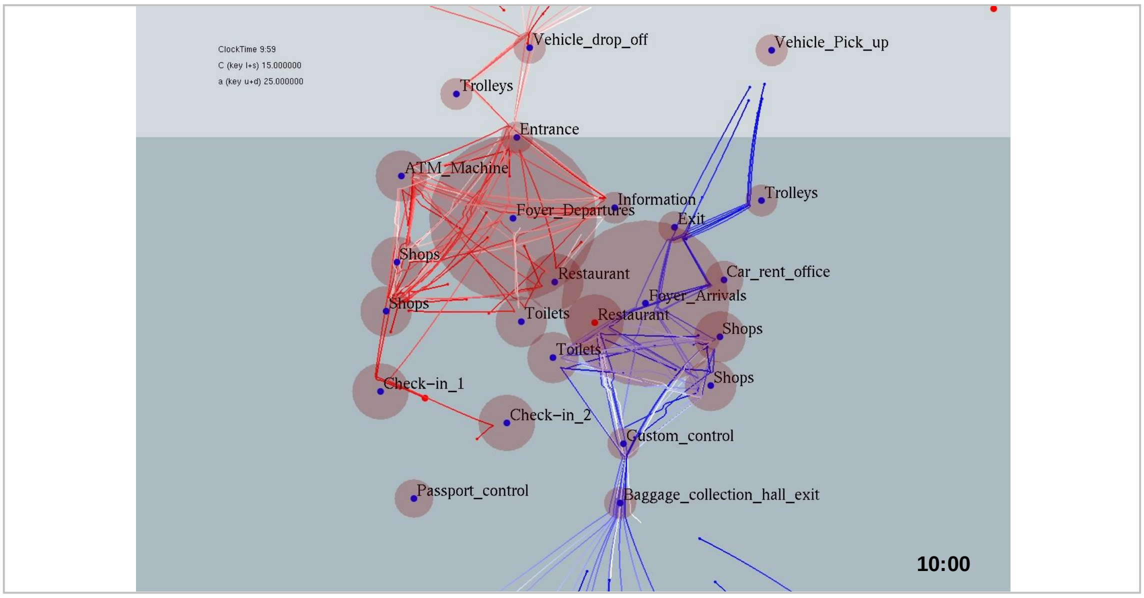


Figure 6. 54: New organization and generated circulation diagrams (see Appendix E.5)

Figure 6.55 shows that some important criteria of circulation design have been fulfilled. First, the foyer area remains a single space. It consists of two parts that are not separated physically. Also, the departure and the arrival movement are distinct and clear.

---

[41] Arrangement allows restaurant area to face on the large foyer space. In this way, the passengers may have better optical view with the central space.

[42] The shops are located symmetrically according to the central axis for both flows.

[43] The re-location of functions has been achieved using the mouse function for drag and move. The mouse function allows the architect to make changes in real time and see the passenger movement behaviour while the changes occur. Also, the method provides an easy and effective way to investigate a large number of possibilities in a short time period. Then, the changes can be used as the starting point for a new simulation.

In addition, the program automatically re-arranges the sign headings towards the functions (destinations) and the coordinate positions of the functional areas are updated in the data file. Also, architect can have the last modified version of design in a .dxf file after the program is terminated.

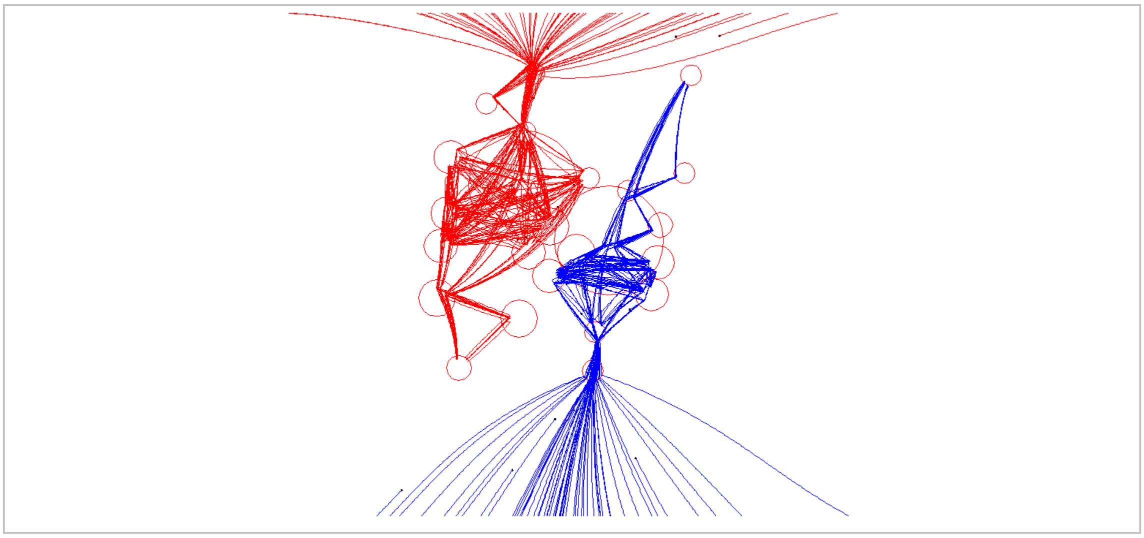


Figure 6. 55: Circulation diagrams

Common facilities (restaurant, toilets, etc.) provide visual and physical connection with the central foyer space for both the departures and the arrivals. The central area can be modified from an empty space with little movement into a sitting or waiting space that is used by all passengers.

Shops (departures and arrivals) are symmetrical and opposite to the central common area. They can serve independently the two flows, which seems to be ‘good’ arrangement in terminal buildings.

In general, both flows have their functional areas and their activities on their route towards their destination. This may achieve the progressive flow without obstructions or backwards movements that may cause confusion or intersecting flows. However, the intersecting behaviour that may occur between the passengers in the same flow is an inevitable phenomenon due to the complex configuration of activities.

6.5.3 Other possible solutions

Possible design solutions are almost unlimited and their investigation is beyond the current research scope or the amount of time available in a design office. However, different representations of the same scheme show that the selection of the design criteria is, to a certain extent, arbitrary.

- **Design scheme and area occupation**

The following example demonstrates one possible elaborated design. In all possible designs, the main principle is the parallel use of the computer program to model the movement behaviour and the architect’s design ability to provide aesthetically satisfactory design solutions.

In figure 6.56 the functional areas are represented as square forms. The building envelope is defined based on the location of functional areas and the circulation diagrams.

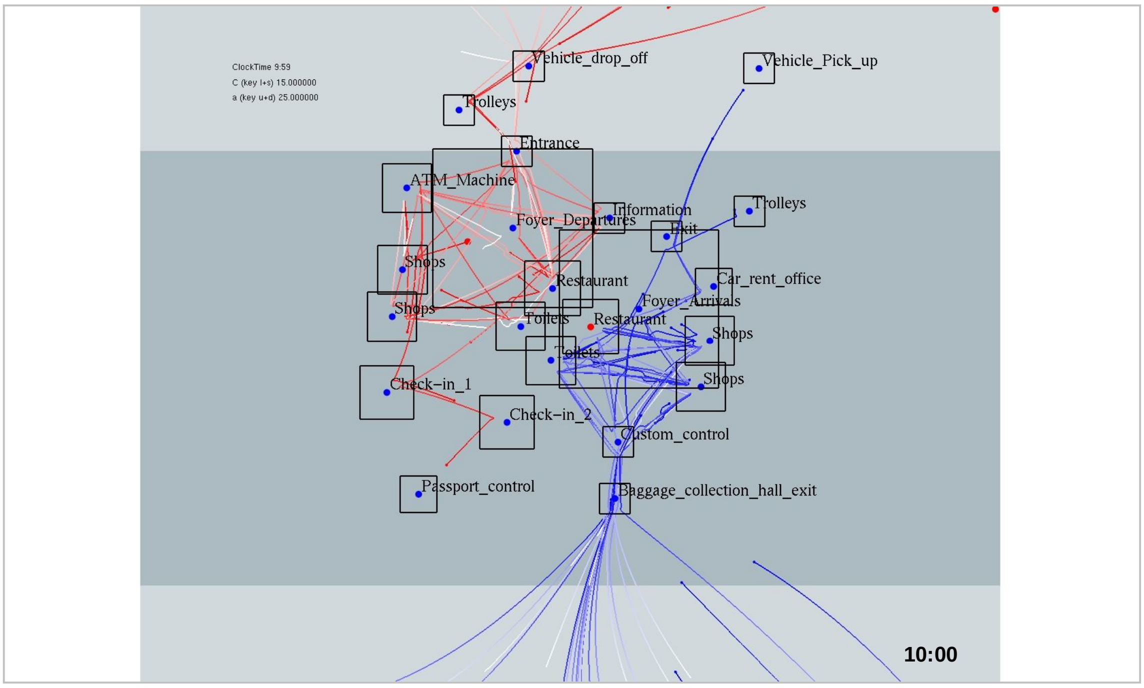


Figure 6. 56: Functional areas are represented as square forms (see Appendix E.7)

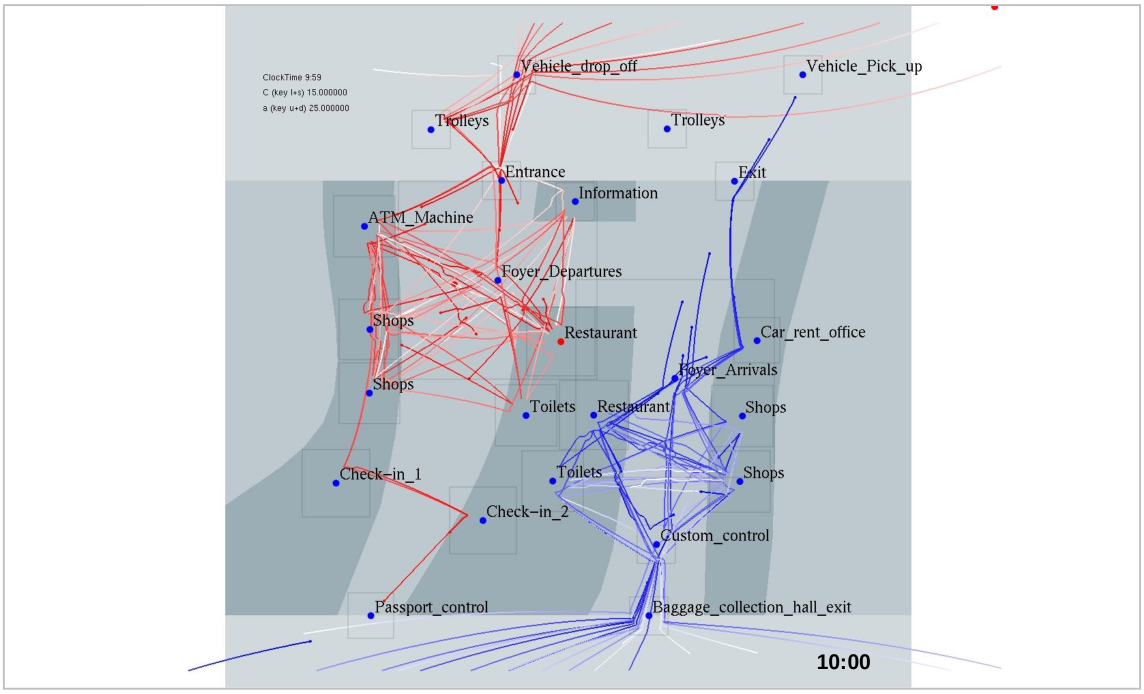


Figure 6. 57: Suggested design and computer-generated circulation diagrams (see Appendix E.8)

Figure 6.57 shows one possible plan of the foyer. In this case, the building envelope is designed in actual square meters. The drawing shows clearly the two main passenger streams, the departure flow on the left hand side and the arrival flow stream on the right hand side.

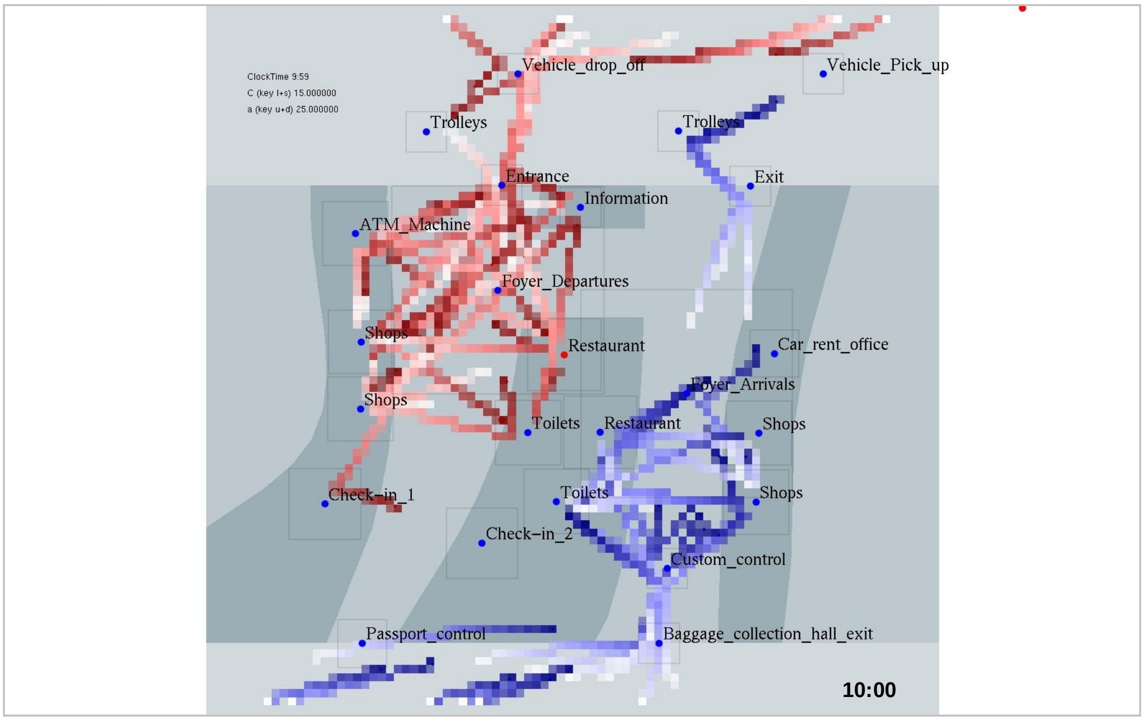


Figure 6. 58: Area occupation of passengers using 'patches' (see Appendix E.9)

Figure 6.58 shows the area occupation of passengers using 'patches' instead of trails. This representation might give an indication of passenger current position, the area occupation and the density of space usage. The red shades represent the departure flow and the blue shades the arrival flow. The current position for each passenger is shown with dark colour and a number of previous steps with light colours.

Figure 6.59 shows a perspective view of the same model.

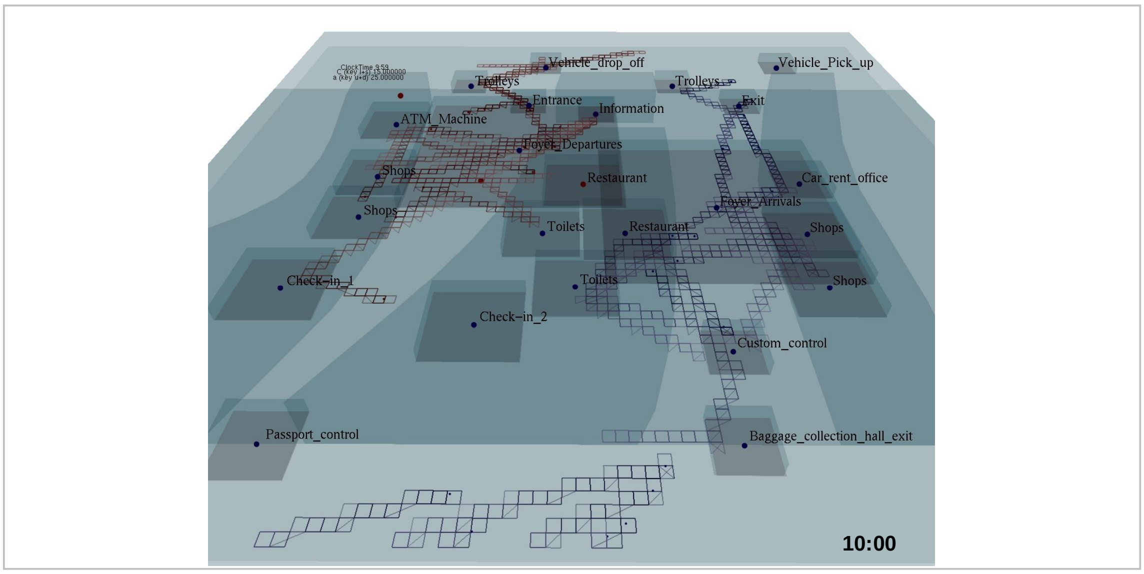


Figure 6. 59: Perspective view of model (see Appendices E.11 and E.12)

6.6 Micro-scale: Spatial configuration, circulation, and facilities design

The micro-scale studies investigate aspects related to the spatial configuration of functional areas, the design of circulation diagrams and of the passenger facilities. Again the design is ‘detailed’ and it will be developed using ‘localized’ models by zooming in on the foyer common area of restaurants, coffee-shops, and so on.

- **Analytic design tasks**

The design is now focused on suggestions for the configuration of facilities like ‘tables’ in the restaurant areas, ‘chairs[44]’ in sitting areas, and so on. Their position can influence and be influenced by passenger movement and the circulation diagrams in the area of investigation. The passenger facilities are indispensable parts of the foyer area, and they have to be taken into account during the design process.

- **Computational tasks**

Again the computer program will be used as a circulation design generator. However, in this scale the emphasis will be given on the ‘organization’ of facilities based on an ‘automated’ design approach. According to this, the program can re-adjust the position of facilities through the interaction with the passengers in real time.

This may raise other questions that deal with the general role of the computer programs in the Design Framework. It is well known that this role varies and can be interpreted in many different ways. ‘Assisted’ or ‘automated’, the roles have different advantages and disadvantages, and they will be discussed at this scale of investigation.

- **Known information**

Known information includes:

i. The ‘detailed design’ that has been derived from the meso-scale including the basic configuration of the functional areas.

ii. The initial configuration of the entry points, the destination points, the boundaries, and the passenger facilities including tables, sitting areas, bar, cashier, and so on. The design dimensions for all elements are based on actual square meters.

---

[44] Note that the rubber feet of the chairs in Richard Roger’s new Heathrow Terminal 5 are all coming off, reminiscent of Jacques Tati’s *PlayTime*.

iii. The passenger interaction rules that are based on the repulsive, the obstacle and the boundaries avoidance effects.

iv. The passenger movement behaviour rules that are based on the sign effect and on the individual's list of compulsory and optional destinations technique.

- **Evaluation of the results**

Results will be evaluated based on:

i. The effective flow of passengers through the different destinations, the functional areas, and the facilities.

ii. The efficient movement in space.

iii. The aesthetic design criteria for the satisfactory organization of the facilities.

6.6.1 Initial organization of the design in the restaurant area

Based on the results from the meso-scale, the common restaurant area is located in a central position of foyer as shown in figure 6.60. It is used by passengers (departures and arrivals) and by the visitors as a space to sit, to have refreshments or to await departure or arrival announcements (if meeting someone).

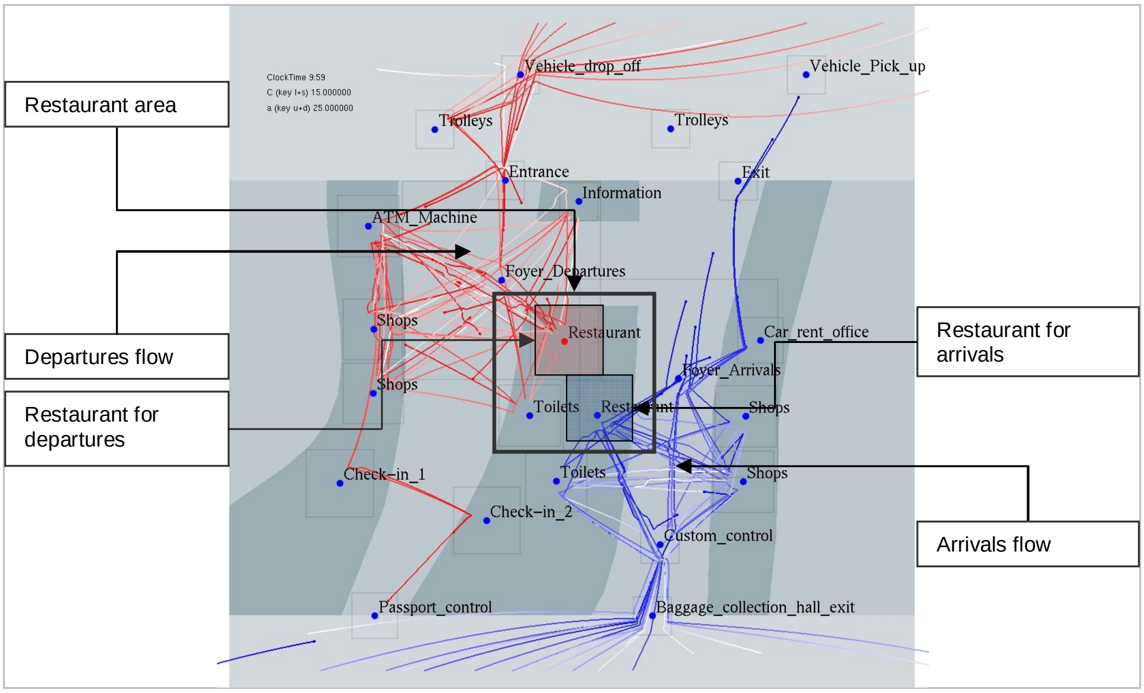


Figure 6. 60: Indication of restaurant area and passenger flows

Restaurant area is divided into two parts in order to serve the departure (top left side) and the arrival (bottom right side) flows. However, the initial study does not clearly separate these parts but it suggests the creation of a single open space where both flows can move in and out and can have direct connection with the rest of the foyer. Such space can be highly used by all passengers during the whole day.

Current design-decision making process examines the important relations between:

i. The spatial configuration of functions,

ii. The passenger movement behaviour, and

iii. The arrangement of passenger facilities.

6.6.1.1 Abstract organization of passenger facilities

Passenger facilities may include tables, seats, and so on. In this section the example of tables is used to demonstrate our approach. The criteria of organizing tables in restaurant or coffee-shop areas might be economic, movement, etc. and according to use and the intended clientele[45].

We assume that tables (blue points) are initially located in the outline area of a restaurant as shown in figure 6.61.

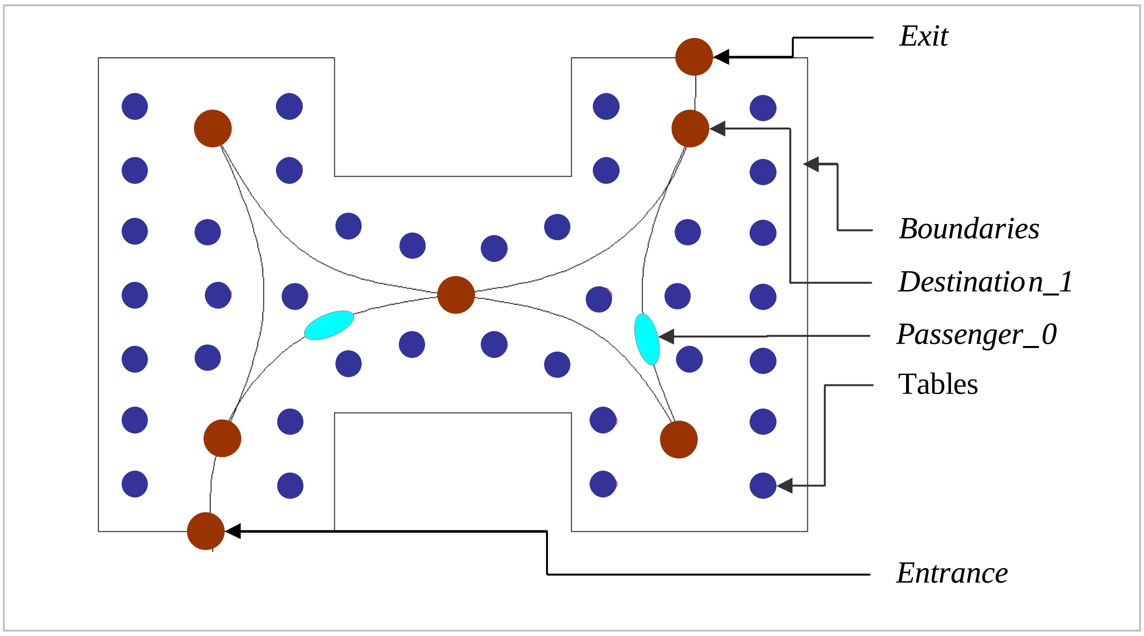


Figure 6. 61: Possible configuration arrangement of table in a restaurant area

We expect that their position will be re-arranged according to their proximity with each other, with the boundaries and the movement of people (circulation diagrams) and by using the rules of avoidance.

---

[45] For instance, economical criteria may influence the number and the arrangement of tables:

i. Expensive restaurants may provide comfortable areas for movement and with few tables in order to attract a special category of customers,

ii. Other restaurants with low prices may provide a large number of tables in order to attract as many customers as possible but with less space for movement.

The parameters that influence the avoidance effect such as repulsive force distance are directly related to the spatial organization of tables. If the repulsive distance increases, fewer tables can be arranged but with more space for movement. If the distance decreases, more tables can be arranged but with less comfortable space for movement.

The passenger movement in the restaurant area is random and can not be predetermined apart from the movement towards the entry and last destination points, which are fixed and they represent the entrance and the exit points respectively (see figure 6.61).

6.6.1.2 Signs and destinations positioning

Figure 6.62 shows the actual outline of the restaurant area together with the arrangement of the signs[46] and destinations. In the first case the signs are distributed on a grid and in the second example they are positioned randomly. The different parameters of the sign effect might influence the circulation outcomes[47].

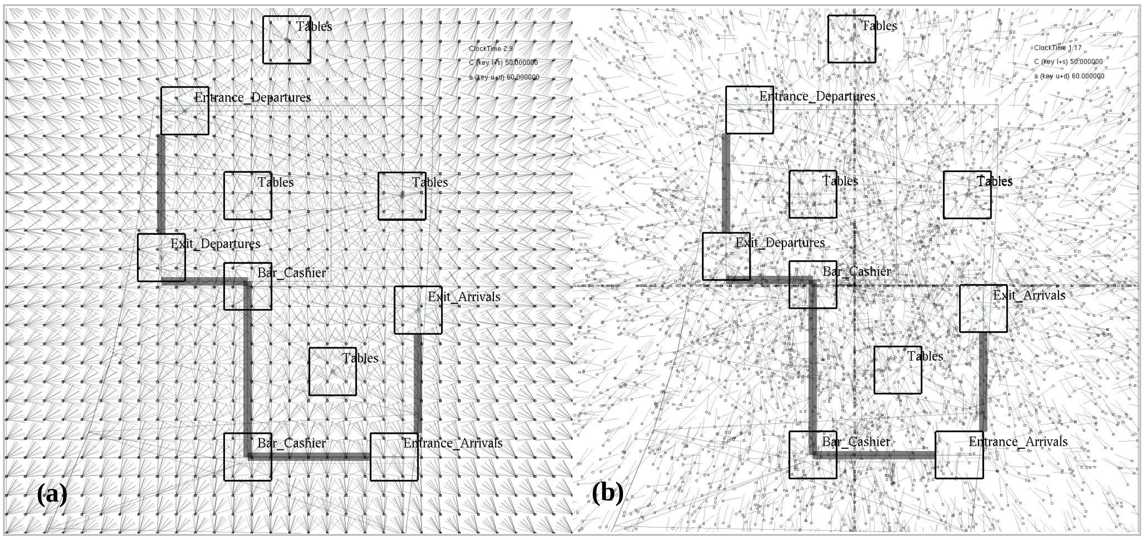


Figure 6. 62: Signs and destinations positioning in the actual restaurant area

The destination positioning takes into consideration the initial arrangement that has been derived from the meso-scale. However additional destination points are applied to serve the current design requirements and the scale of the model. Again the destinations

---

[46] The position of signs has a direct relation with the design outcomes. If the signs are distributed using a random number generator, then different outcomes emerge each time the program runs.

[47] a. The continual movement towards the signs can be achieved with a maximum distance between them. If the distance is larger, then the passengers stack and they cannot move towards their destinations.
b. The curved lines of movement (paths) can be controlled by the sign strength C_{parallel}. A large value of C_{parallel} means that the passengers can move to the sum of surrounded signs' headings. If the C_{parallel} is small then the curved line of movement is intense.
Different values of C_{parallel} can be applied according to the scale of design. At meso-scale the circulation diagrams can be represented as slightly curved or as straight lines since they mainly represent connections between functional areas. At micro-scale where the interior of the space is examined, the circulation diagrams are represented as curved lines with fluctuations due to the large number of obstacles that may influence passenger movement.

are used to represent the functional areas, the passing points, the intersection points, and so on.

6.6.1.3 Boundaries and facilities positioning

The ‘detailed design’ in the restaurant area demands the use of various building elements like space partitions, walls, and openings. These elements shall be called boundaries.

Figure 6.63a shows the positioning of boundaries. They are basically lines that consist of a number of static ‘individuals’[48].

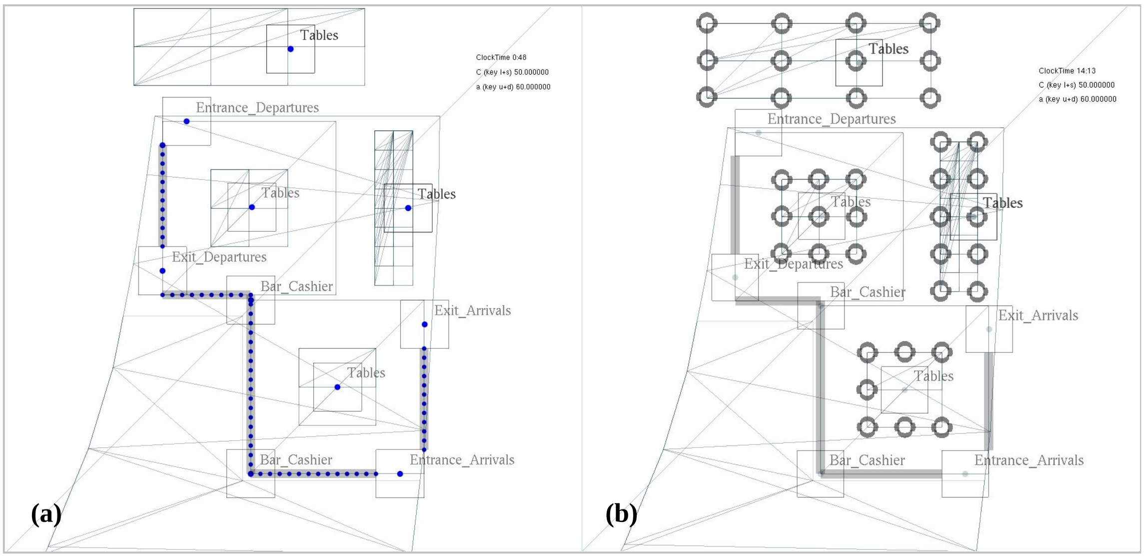


Figure 6. 63: a. Boundaries, b. Facilities positioning

Figure 6.63b shows an example of facilities positioning according to the architect’s judgement and choice[49]. The facilities are groups of special types of ‘individuals’, which in this case shall be called ‘tables’ and they can take the form of stationary or dynamic objects.

6.6.1.4 Passenger movement behaviour

Passenger route choice movement behaviour uses the ‘Experiment type 3: Individual’s list of compulsory and optional destinations’ (see section 5.6.5.2).

---

[48] The boundary length, position and shape can be geometrically defined in different ways like using input .txt files and mouse functions. In this case an input data file is used. The file defines the coordinates of boundary lines and then the computer code calculates the number of ‘individuals’ that need to be placed along each line.

[49] The facilities positioning can be achieved using similar approaches to those that have been described for the boundaries like input data files and the mouse functions. In this case mouse function is used.

Figure 6.64 shows the initial configuration and the numbering of destinations that influence our movement scenario[50]. The basic direction of movement for both flows (departures and arrivals) is taken from the meso-scale. The entry and the destination points are shown with the red (top left hand side) and blue dots (bottom right hand side) of the design scheme.

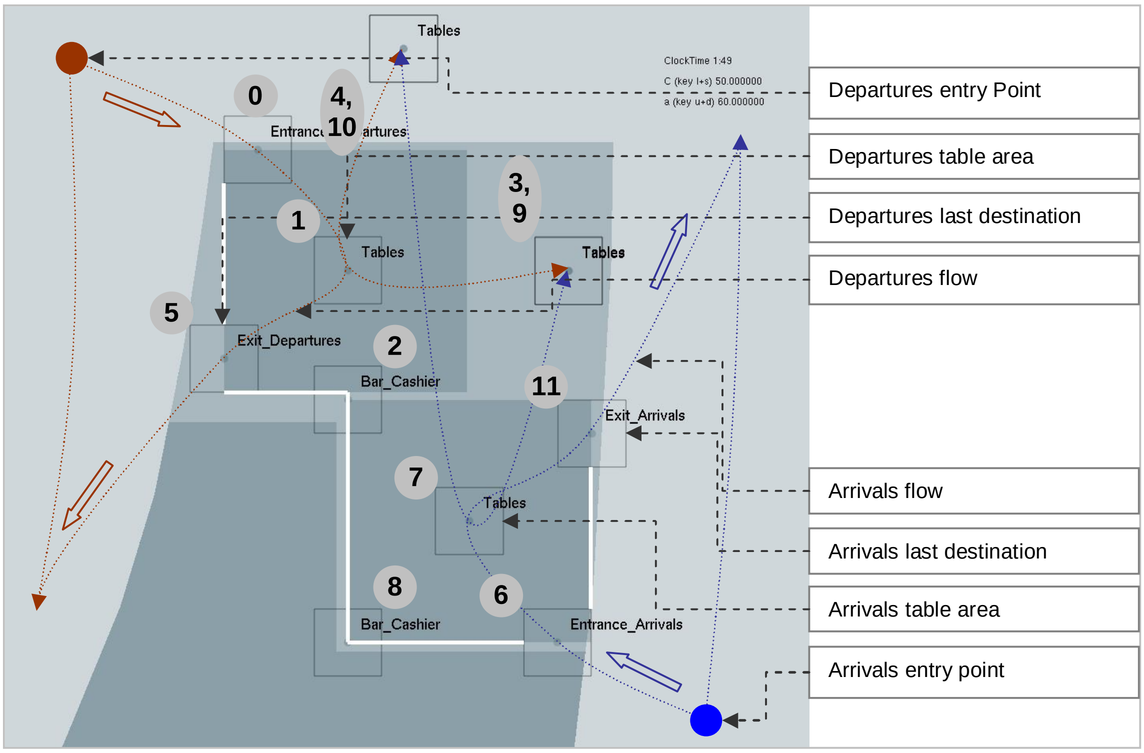


Figure 6. 64: Graphical representation of movement scenario for departures and arrivals

Figure 6.65 shows the graphical representation of the departure flow movement scenario. Destinations are as follows: 0.entrance (departures), 1.table area (departures), 2.bar-cashier, 3.table area, 4.table area, and 5.exit (departures).

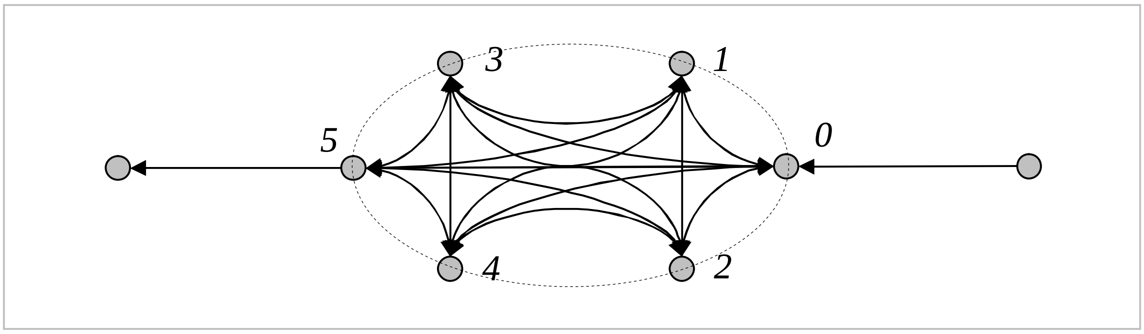


Figure 6. 65: Departure flow destination diagram

---

[50] The current case study investigates one possible movement scenario among the large number of different passenger route choice behaviours. Route choices are infinite and they can only be specified by taking into account different design objectives.

After the passengers[51] enter the restaurant, they can move randomly in the restaurant area for the departures including the two table areas outside. The exit point serves as the last destination point.

Figure 6.66 shows the graphical representation of arrival flow movement scenario. Destinations are as follows: 6.entrance (arrivals), 7.table area (arrivals), 8.bar-cashier, 9.table area, 10.table area, 11.exit (arrivals).

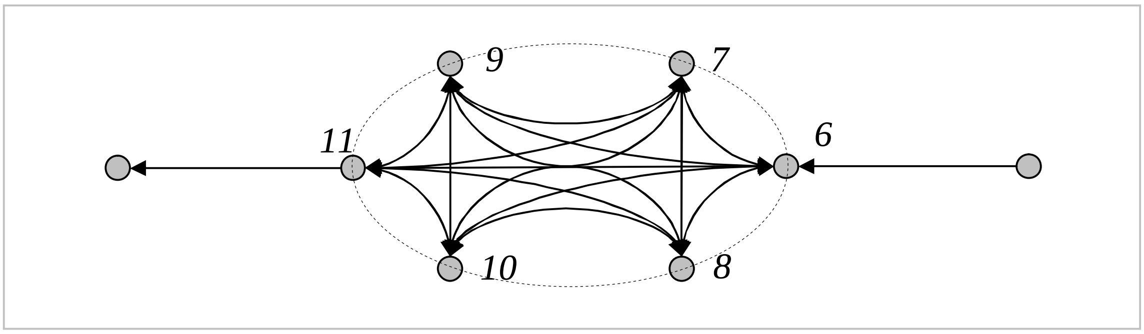


Figure 6. 66: Arrival flow destination diagram

The arrival flow has similar route choice behaviour to that of the departure flow where the entry and the last destination points are fixed, movement inside the restaurant is random, and their movement includes the table areas outside restaurant. Differences can be found on the number of passengers and their time of appearance, which can influence the overall behaviour of the model[52].

6.6.1.5 Individual interaction behaviour

The current case studies use three basic types of 'individuals':

i. The passengers

ii. The tables

iii. The boundaries

---

[51] The departure flow cannot move towards the arrival restaurant area since this may cause intersecting phenomena. Also, it may cause confusion on exit point and it may result the movement of the departure flow towards the area for arrivals.

[52] In case of the departure flow, the passengers can enter the restaurant in groups or individually but at a different time and they tend to stay for a longer time period in the restaurant area. The flexibility in the number of passengers and their time of appearance may provide a low passenger density. In case of arrival flow, the passengers arrive in groups at the same time and they tend to spend less time inside the restaurant area since they move quickly towards the exit. The large number of passengers using the same area at the same time may increase their density.

In order to make the design process and the program as simple as possible, all types are subjected to various virtual repulsive forces[53].

- **Passengers**

The interaction behaviour between the passengers is achieved using virtual social repulsive forces. Additionally, the attractive effect may be applied if the passengers tend to move in groups[54].

- **Tables**

The tables are modelled using the rules from the obstacle avoidance effect. Their position is re-arranged due to the virtual repulsive forces between them and the passengers or the boundaries.

- **Boundaries**

The behaviour of boundaries is based on the boundary effect. The boundaries are used to repel the passengers and the tables inside and outside the restaurant area. In comparison with the other two types, they are static ‘individuals’ and they form boundary lines.

6.6.2 Approaches to modelling and initial simulations

The current study will investigate design possibilities using three different approaches.

- **First approach of modelling**

In the first approach, the passengers move in the restaurant area without any application of tables at the beginning of the simulation. Then, after the circulation diagrams are gradually generated, the tables will be positioned according to the generated diagrams.

Figure 6.67a shows the initial configuration of the model including the entry and the destination points for the departure and the arrival flows. Figure 6.67b shows possible generated circulation diagrams[55].

---

[53] The virtual repulsive force parameters between any combinations of individuals can be modified and can influence the final design outcomes. For instance, the virtual repulsive force – distance between the passengers and the tables may be less than the distance between passengers or passengers and tables.

[54] This phenomenon has been investigated in section 5.6.4.2. However, in this case is difficult to be applied since the passengers once enter the restaurant, they do not move in group but they are separated and move randomly towards different destination goals.

[55] The fluctuations of the passenger movement (curve lines) can be achieved by modifying the sign strength C_{parallel} .In this case small C_{parallel} ($C_{\text{parallel}} = 50$ force units) has been chosen since this value produces intense curved circulation lines.

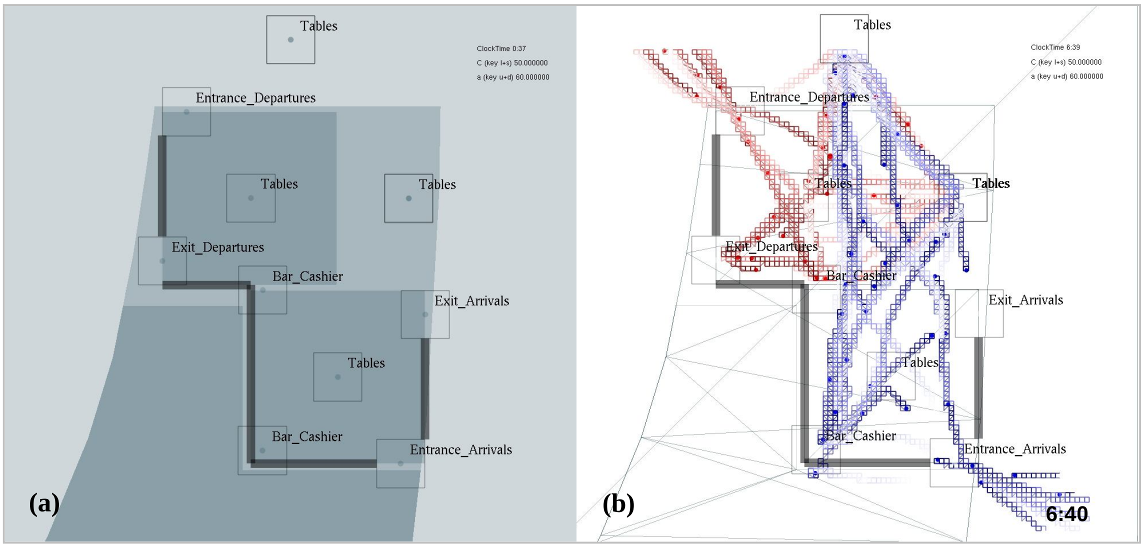
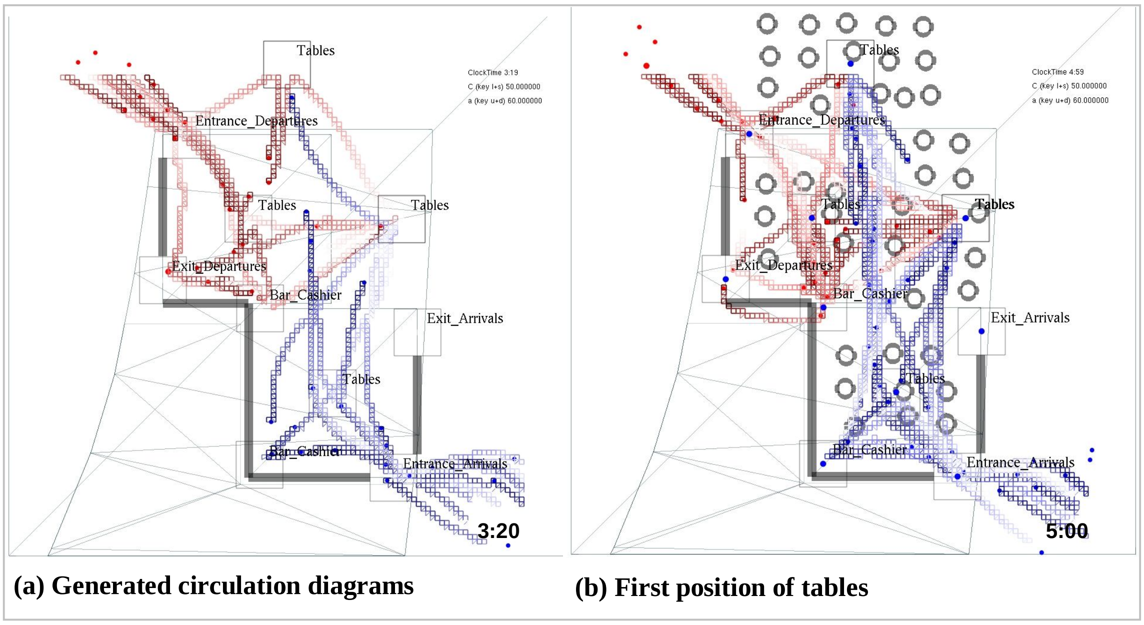


Figure 6. 67: a. Initial configuration of the model, b. Generated circulation diagrams (see Appendix F.1)

The circulation diagrams affect the form of space that can be generated. Here, we assume that the circulation diagrams represent the 'positive' space that is occupied by the passengers. The 'negative' space is the empty space in between[56].

Figure 6.68 shows two steps towards the allocation of tables. Figure 6.68a shows the generated circulation diagrams (virtual clock time 3:20). Figure 6.68b shows the restaurant area after the tables have been positioned[57] (virtual clock time 5:00).



(a) Generated circulation diagrams **(b) First position of tables**

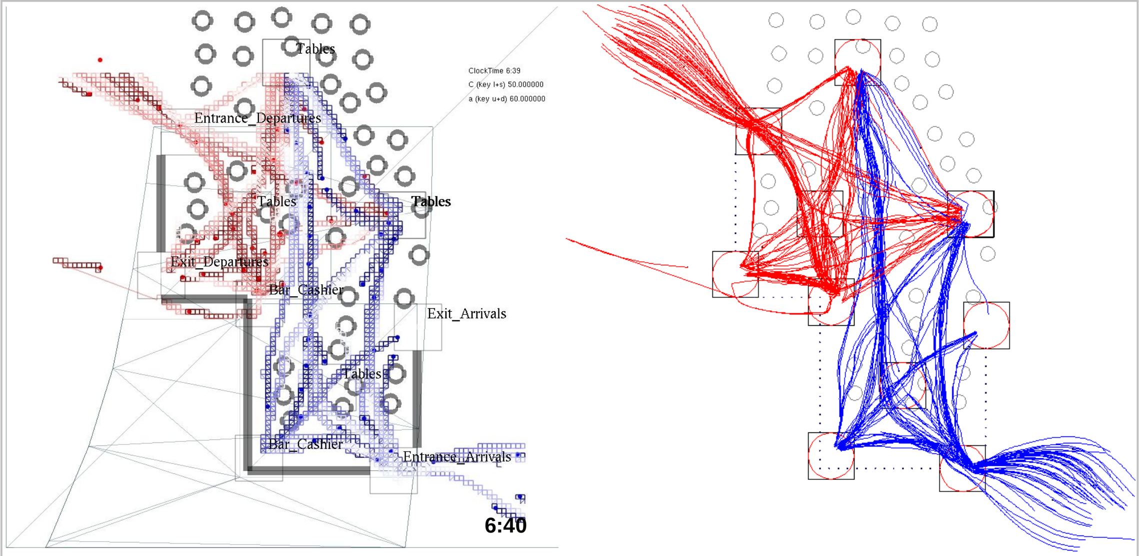
Figure 6. 68: First approach of modeling: first (a) and second (b) steps of the design approach (see Appendix F.2)

---

[56] Models will be developed based on the relation between the circulation diagrams and the empty space that can be filled with passenger facilities.

[57] The tables positioning is based on the assumption that the best places are in the 'negative' or empty spaces. This can be done using the mouse function while the program runs.

Figure 6.69c shows the new positions of tables (virtual clock time 6:40). The re-positioning is due to the virtual repulsive force that allows interaction between the passengers and the tables. Figure 6.69d shows all possible passenger movements. The diagrams with the red colour show the departure flow and with the blue colour the arrival flow.



(c) Re-positioning of tables

(d) Final circulation diagrams and tables

Figure 6. 69: First approach of modeling: third (c) and forth (b) steps of the design approach (see Appendix F.2)

The final positioning of tables shows that there is no large difference between the first positioning (virtual clock time 5:00) as shown in figure 6.68b, and the last automated arrangement of tables (virtual clock time 6:40) as shown in figure 6.69c. This concludes that once the circulation diagrams have been generated, the 'positive' and the 'negative' spaces have also been largely specified. The satisfactory position of tables is the empty remaining space.

This approach shows that:

i. It involves parallel methods of computer-generated circulation diagrams and the architect's judgement.

ii. The design results are influenced by the passenger route choice behaviour at a global level and the passenger interaction behaviour at a local level.

iii. The modelling achieves efficient movement in circulation areas and at the same time the tables are positioned in the areas that do not cause obstruction of passenger movement.

iv. However, useful space is wasted due to the special attention that is given to the circulation and then to the arrangement of tables. This may be acceptable in design tasks where movement is more important than the number of tables.

- **Second approach of modelling**

In the second approach, the tables are positioned from the beginning and remain stationary. The tables affect the movement of passengers but their position is not affected by passenger repulsive forces.

Figure 6.70a shows the initial configuration of tables[58] in the two restaurant areas and the surrounded area of the foyer. Also, it shows the passenger movement behaviour. The image is captured at the same time (virtual clock time 6:40) as with the first approach in order to compare the two results. Figure 6.70b shows the final circulation diagrams as the results of passenger movement and interaction behaviour.

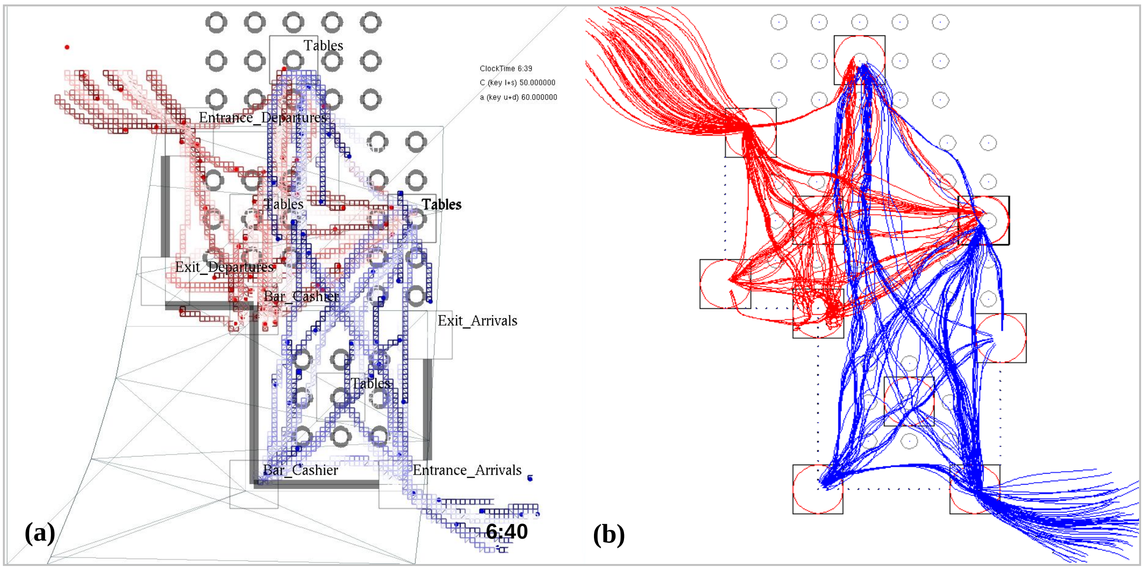


Figure 6. 70: a. Initial configuration and passenger movement behavior, b. Final circulation diagrams (see Appendix F.3)

The current approach shows that:

i. It involves empirical arrangement of the tables that can satisfy the restaurant space requirements. Also, the space is not wasted because a maximum number of tables can be located.

---

[58] The initial configuration is based on the information derived from Neufert [1980] according to the space requirements in restaurant areas. Specifically, the arrangement of tables uses the circular tables, which can be designed with 2.00 meters distance between each other and in rectilinear fashion.

ii. The final circulation diagrams are the results of the passenger movement behaviour at a global level and the interaction between the passengers and the tables at a local level.

iii. The criterion of efficient movement in the restaurant area is not fulfilled since there is not any correlation between the arrangement of tables and passenger route choice. Also, the configuration fails to provide an effective flow because passengers need to pass around the tables to find their destinations. This causes obstructions, blockages, and confusion of passenger movement.

iv. The modelling can be improved by changing the position of the functional areas and hence the passenger route choice behaviour in order to avoid the movement through the tables.

- **Third approach of modelling**

In the third approach, the tables are applied from the beginning in a predetermined grid. Their configuration is achieved through the dynamic interaction between various individuals in real time and it shall be called the ‘automated’ procedure.

Figure 6.71a shows the first positioning of the tables. Figure 6.71b shows the generated circulation diagrams and the final spatial configuration of tables (virtual clock time 6:40).

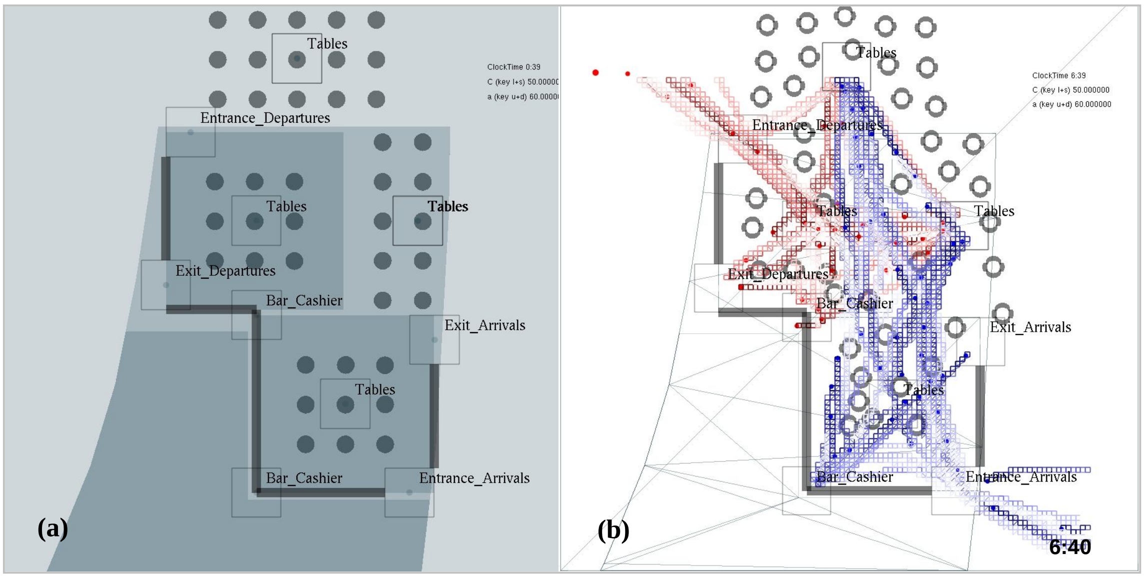


Figure 6. 71: a. Initial configuration, b. Generated circulation diagrams and final configuration of tables (see Appendix F.4)

This approach shows that:

i. It involves dynamic interaction between tables, passengers and boundaries. The position of tables and passengers can be re-adjusted in real time while the program runs.

ii. The generated design is the ‘automated’ outcome of passenger interaction with the tables at a local level.

iii. The changes are not pre-programmed or known in advance and the results are unpredictable.

iv. During the simulation, the arrangement of tables is adapted to the local conditions of passenger movement behaviour.

v. However, the tables are clustering in specific positions such as the departure exit causing obstructions of movement and leaving much unused space as shown in figure 6.71b. This is the result of the interaction with passengers who randomly ‘drift’ around the tables.

6.6.2.1 Evaluation of the initial results

The initial computer-generated results will be evaluated based on:

i. The generated circulation diagrams

ii. The configuration of tables

iii. The initial design and the passenger route choice behaviour

i. Generated circulation diagrams

Generated circulation diagrams are investigated in relation to their ability to provide effective flow and the efficient movement of passengers in the restaurant area.

Figure 6.72 shows the departure (red trails) and arrival (blues trails) circulation diagrams. The distribution of flows seems to be equal in all destination points allowing passengers to use the entire space. However, their movement towards the table areas outside might cause intersecting phenomena (see area A). Also, the departure restaurant area seems to be wasted due to the use of space from both flows.

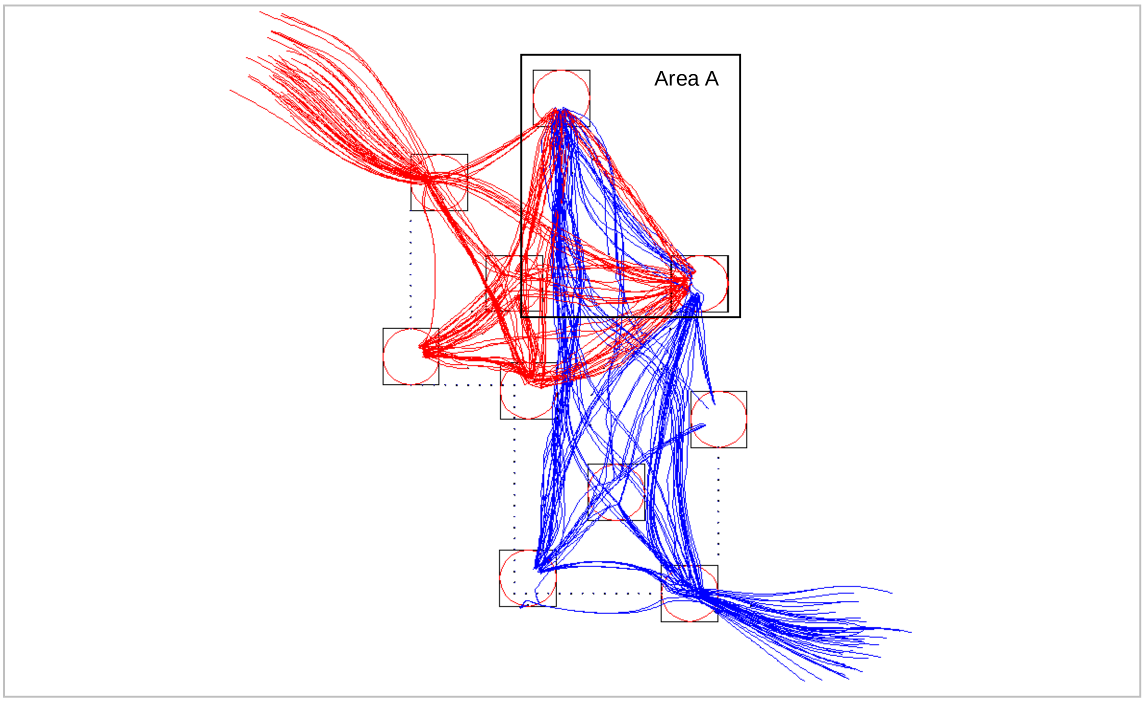


Figure 6. 72: Departures and arrivals passenger flow diagrams (see Appendix F.1)

The brief asked for a design that will provide a common area, although the results showed that the movement is obstructed by intersecting phenomena providing a less satisfactory effective flow. The reasons include the spatial configuration of the functional areas, the specific selection of activity destination points (entrances, exits, boundaries, etc) and the selected route choice behaviour.

ii. Configuration of tables

In this section the relationship between the circulation diagrams, the configuration of tables and the role of architect in the procedure will be examined[59].

a. First and second approach of modelling

Figure 6.73 shows the first and second approach of modelling. In the first approach (figure 6.73a) it was found that passenger movement covers large proportion of restaurant area, although the allocation of tables is found to be ‘good’ since they occupy the empty remaining space.

Figure 6.73b shows the second approach. In this case there is not any correlation between passenger movement and position of tables[60]. As a result, the movement of

---

[59] Common parameters of modelling include: a. Virtual clock time 5:0 min. after the simulation starts, b. Configuration of the environment, the boundaries, the passenger behavioural rules, and the number of people.

[60] In the second approach the position of tables is decided from the beginning and it cannot be changed through the process. It is basically a positioning based on intuitive criteria (proximity with entrances, exits, and so on).

passengers is obstructed by the tables causing blockages and reduction of passenger movement speed[61]. This approach offers a less satisfactory effective flow and less efficient movement towards destinations.

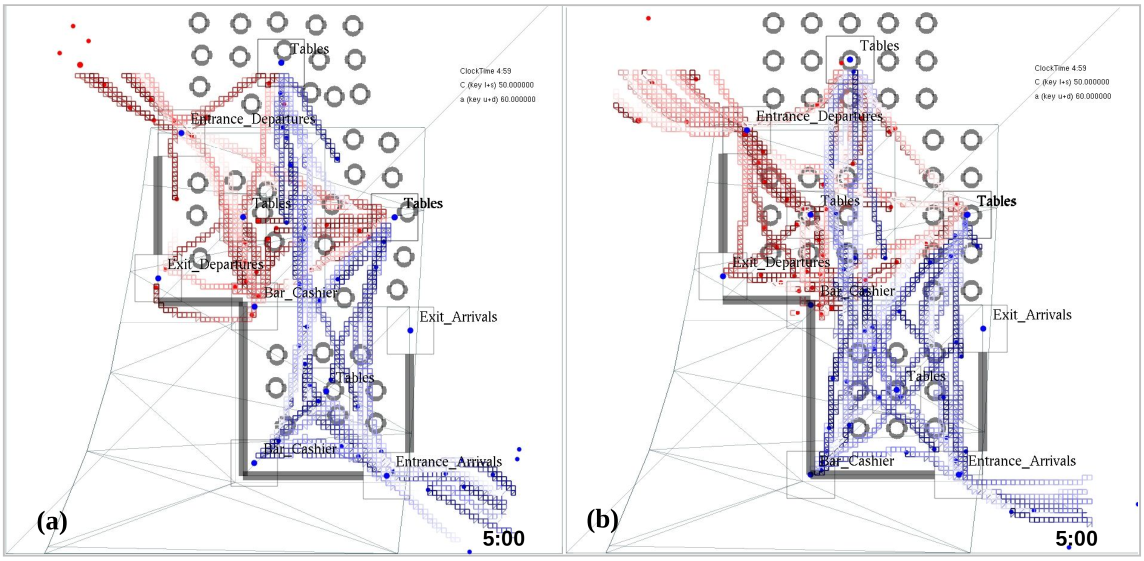


Figure 6. 73: First (a) and second (b) approach of modeling

The first approach provides better results that the second in terms of the configuration of tables, the movement of passengers, and the generated circulation diagrams. However, negatives include the non-uniform distribution of the tables in space.

b. First and third approach of modelling

Figure 6.74 shows the first and third approach of modelling.

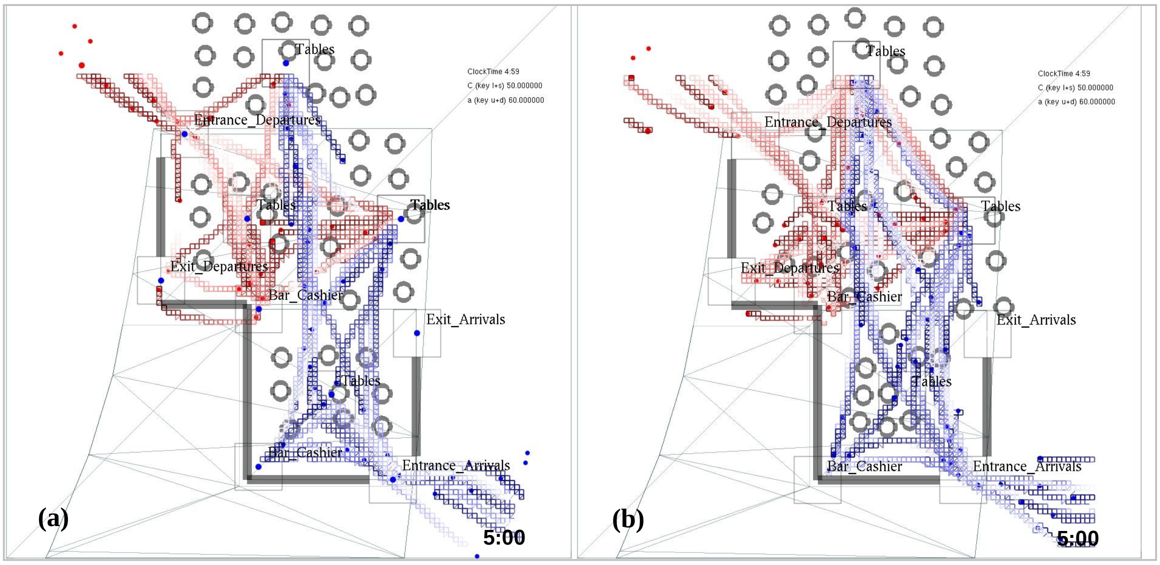


Figure 6. 74: First (a) and third (b) approach of modeling

---

[61] The passengers do not have the ability to completely change their direction of movement apart from their local position in order to avoid obstacles, boundaries, and so on. Thus radical changes on their movement cannot be observed.

Both models have the ability to re-adjust the position of tables since the configuration is based on the empty space. The passengers are free to move according to their route choice destination selection and they can interact with the tables.

However, the comparison shows that the first achieves better configuration of tables than the third. The reason is mainly because the first approach involves a parallel interaction of the model with the architect[62]. On the contrary, in the third approach that is fully 'automated', the program re-adjusts the tables without taking into consideration factors that are not fully described or pre-programmed.

Also, both approaches cannot distribute the tables uniformly in the virtual environment.

iii. Initial design and route choice behaviour

The computer-generated results can be evaluated according to:

i. the initial design of the boundaries, the spatial configuration of different destinations (entrances, exits, etc) and their influence on,

ii. the passenger route choice behaviour.

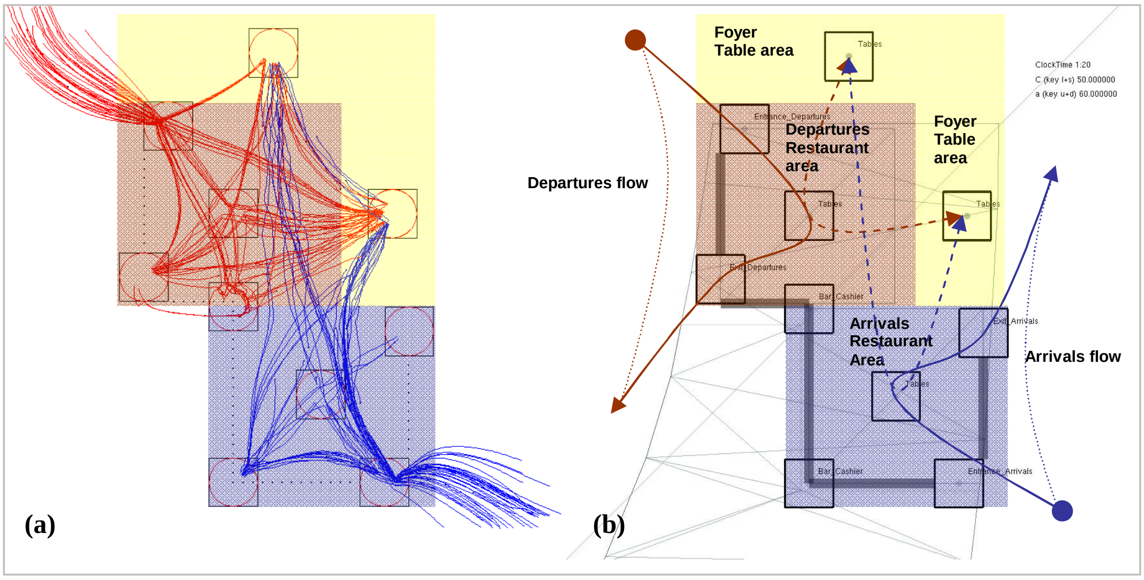


Figure 6. 75: Departures and arrivals passenger flow diagrams (a) and initial design configuration (b)

Figure 6.75 shows circulation diagrams in relation to the initial design configuration. This arrangement was found to cause circulation problems due to the unclear separation

---

[62] The architect is able to analyse the results in real time and take the best design decision.

between the two flows. Intersection phenomena were observed that might cause the blockage of area if the number of passengers is increased[63].

Furthermore, in each restaurant area, the arrangement of the entrances, the exits, and the boundaries influence passenger movement. For instance, in the area for arrivals, the configuration of the entrance and the exit destination points in relation to the movement of passengers is found to be less functional.

One such example is the movement of arrivals towards the foyer table area (see figure 6.75b). In this case, the passengers need to return back to the arrivals exit in order to move towards the main exit of the ferry terminal. It is believed that such route choice behaviour does not provide satisfactory efficient movement.

In general, the passenger movement scenarios and the route choice selection in the restaurant area cannot be described as satisfactory in this initial model.

6.6.2.2 Suggested solutions

In order to improve the initial model, suggestions on a new design configuration and passenger movement scenarios can be made.

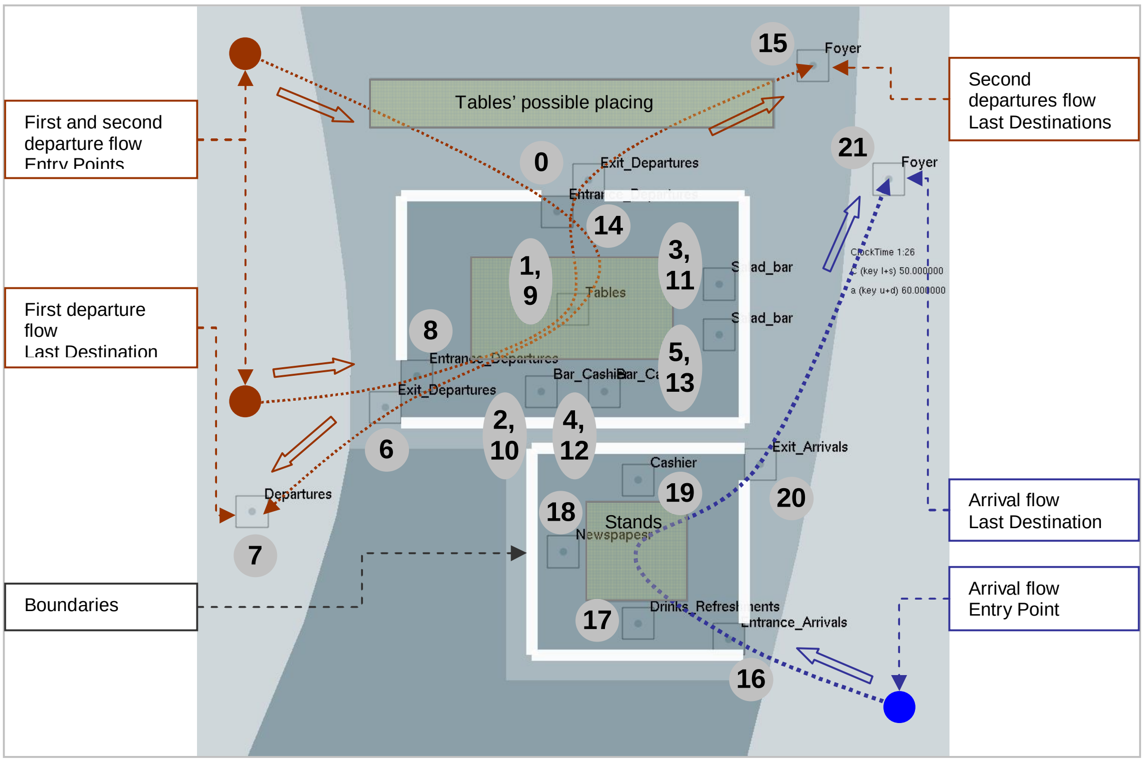


Figure 6. 76: Possible new modified design

---

[63] The intersection phenomena can be accepted and improved if the passengers are in the same flow and with similar route choice behaviour. However, in the current case study where the departure and the arrival flows are clearly distinguished and they include different destination goals, the passenger flows needs to be treated very carefully.

Figure 6.76 provides a possible new design[64] of functional areas, destinations and boundaries. The basic design scheme and its location remains the same. The departure area has been extended and it has been separated from the arrival area. Also, the restaurant area for arrivals has been changed into a ‘convenience store’ (drinks, cold food, newspapers, etc.). The separation has been decided mainly due to the differences that can be found between the movement activities of the two flows. The facilities for restaurant and for convenient shop shall be called tables and stands respectively.

The new model uses two departure flows[65] as shown in figure 6.76. Both flows can enter restaurant area and move randomly towards various activities.

Figure 6.77 shows the destinations for the first departure flow: 0.entrance[66] (departures), 1.table area (departures), 2.bar-cashier, 3.salad bar, 4.bar-cashier, 5.salad bar, 6.exit[67] (departures), and 7.departures (towards).

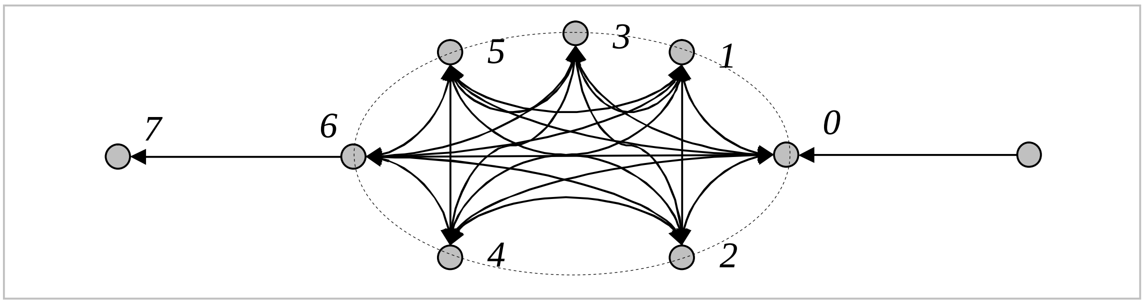


Figure 6. 77: First departure flow destination diagram

Figure 6.78 shows the destinations for the second departure flow[68]: 8.entrance (departures), 9.table area (departures), 10.bar-cashier, 11.salad bar, 12.bar-cashier, 13.salad bar, 14.exit (departures), and 15.foyer.

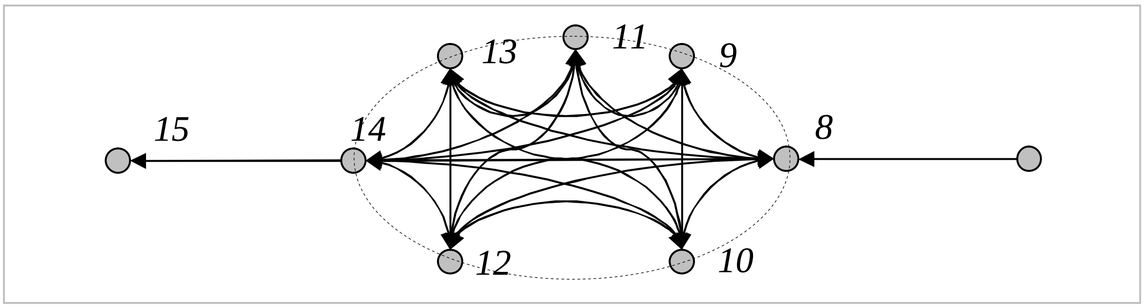


Figure 6. 78: Second departure flow destination diagram

---

[64] The modified design takes into consideration the results derived from the initial model but also it emphasises the architect’s creativity and sensitivity as it has been explained in the Design Framework.

[65] The departures stay longer in the common area and their route choice includes various leisure activities like shopping and sitting in a restaurant.

[66] First departure flow enters from the top side of restaurant area.

[67] Exits and entrances are designed together.

[68] Second departure flow enters from the left hand side of restaurant area that is closer to the shops.

The arrival flow[69] enters the convenience store coming from the arrival hall and they can randomly select various destinations (newspapers, drinks, etc.).

Figure 6.79 shows the destinations for the arrival flow: 16.entrance (arrivals), 17.refreshments, 18.newspapers, 19.cashier, 20.exit (arrivals), and 21.foyer.

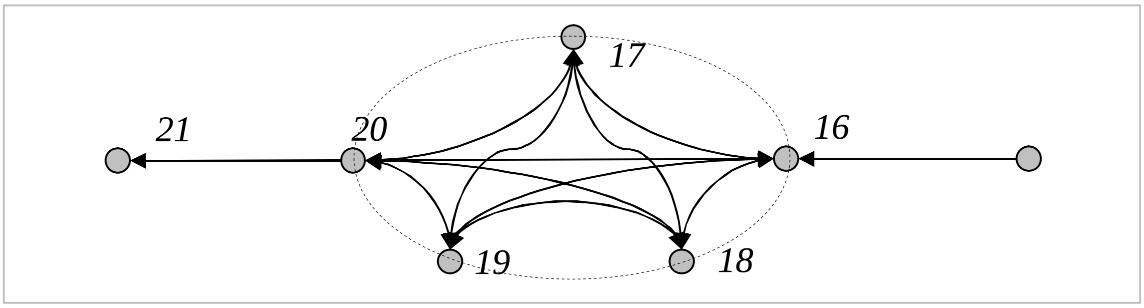


Figure 6. 79: Arrival flow destination diagram

i. Computer-generated circulation diagrams

Figure 6.80 shows the initial results derived from the simulation. In this case, the configuration of facilities is not examined. The investigation is concentrated upon the passenger movement behaviour and the generated circulation diagrams.

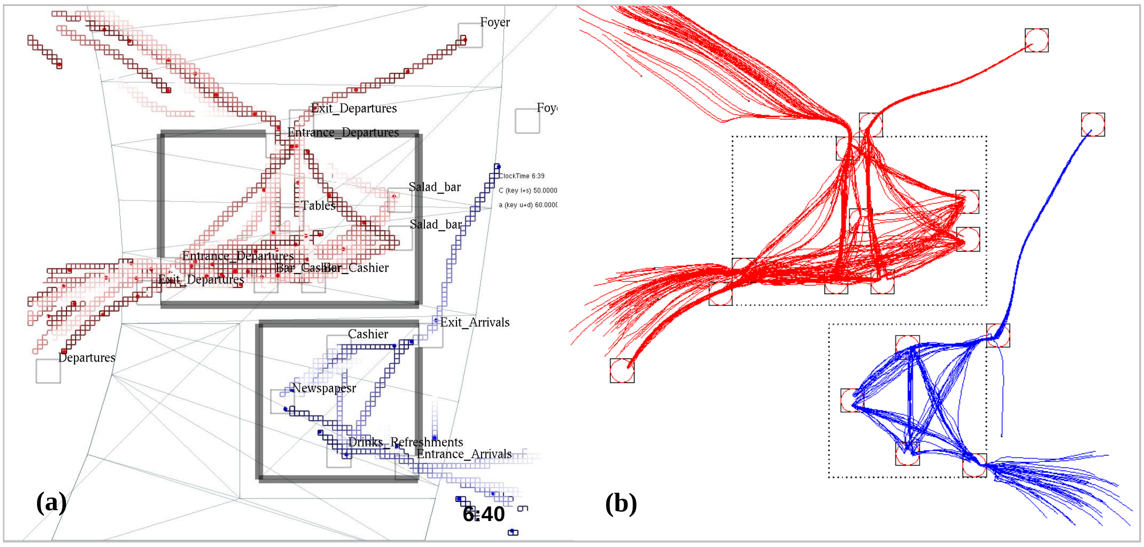


Figure 6. 80: Computer-generated diagrams after the design is modified (see Appendix F.5)

The model achieves the clear separation between the departure and the arrival flows[70]. Also, both flows have the ability to move towards their destination following a smooth

---

[69] The arrivals stay in the building for a short time period and their main destination goal is to be directed towards the building exit. Also, their movement is rush and they do not tend to sit in the restaurant area apart from buying cold foods and drinks.

[70] Such arrangement is found to be important in terminal buildings since departure and arrival flow act separately in most parts of terminals without any contact mainly because passengers are motivated to move towards different destinations and goals, at movement speed, etc.

flow[71] and without any blockage, intersection of flows from the opposite directions, and so on.

ii. Allocation of passenger facilities

Figure 6.81 shows the process of allocating the facilities based on the third approach of modelling. Figure 6.81a shows the initial configuration where the tables are positioned in the departure restaurant area and in the common foyer area. The stands are positioned in the arrival shop area (convenient shop). Figure 6.81b shows the final arrangement of facilities and the computer-generated circulation diagrams.

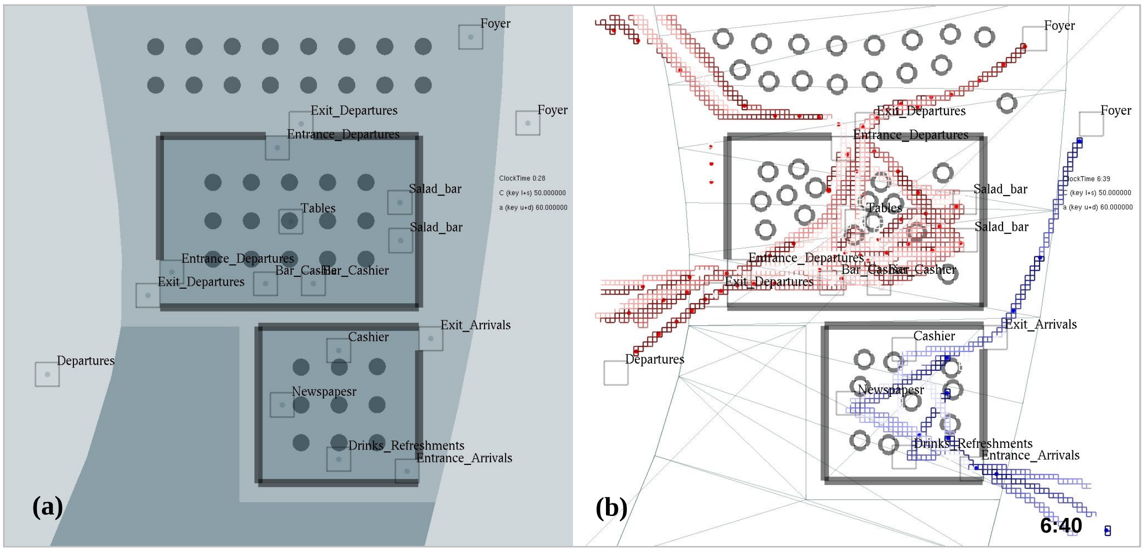


Figure 6. 81: a. Initial configuration of the model, b. Final arrangement of the facilities (see Appendix F.7)

The configuration of the facilities is found to be satisfactory and to be adapted to passenger movement. However, the detailed analysis of the results shows that a number of tables are arranged in the four corners of the space allowing a large free central space for movement. Also, in some areas the clustering of tables may not provide comfortable sitting.

In general, this approach may be useful for the arrangement of facilities, though the space given for the facilities is larger than the space for movement.

6.6.3 Other possible solutions

The current research examines some representative designs since the number of possible solutions is tremendous and their investigation is beyond our research scope. The

---

[71] Possible obstructions may cause confusion, intersection, and in case of emergency situation possible panic. However, intersecting phenomena in the same flow are inevitable due to the complexity of the movement and the random selection of the activity destinations.

following example demonstrates one possible elaborated design where small changes may influence the design philosophy and may provide different design results.

i. Integration between departure and arrival flows

Although the competition brief asks for a specific design arrangement[72] it is well known that a foyer area can be experienced in different ways. The diversity of the passenger movement behaviour and the functional arrangement allows the investigation and the generation of many different movement scenarios.

Figure 6.82 shows an additional movement scenario for arrival flow. Passengers are given the possibility of using the restaurant area that has been previously used only by the departure flow[73]. The destination selection is as follows: 22.entrance (arrivals), 23.tables (restaurant), 24.exit (departures), 25.foyer.

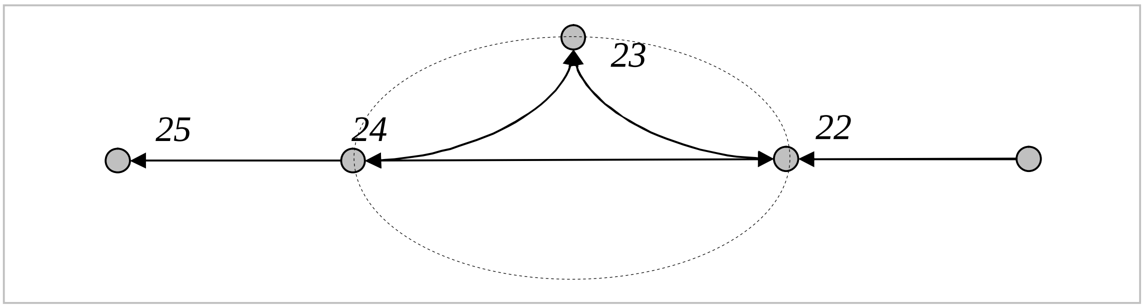


Figure 6. 82: Arrival flow destination diagram

The arrival flow can enter the restaurant, use the table facilities and move to the exit on the top side of restaurant[74].

Figure 6.83 shows the suggested design configuration and results of the simulation. Figure 6.83b shows the departure (red trails) and arrival (blue trails) circulation diagrams. Intersection behaviour can be observed although this is only in the restaurant area where the complexity of the movement allows such phenomena even between the same flows. It is believed that this solution may provide satisfactory results since the two flows can use the common area but their movement cannot influence the smooth flow towards departure and arrival destinations.

---

[72] The area of foyer can be used for the departure and the arrival flows taking into consideration the effective movement of passengers towards different destinations as well as the creation of a common area. It is believed that the design suggestions have achieved the provision of a clear separation between those two movements.

[73] Passengers from arrivals may select to stay in the foyer area, possibly to wait for a bus, for a friend, and so on. In this case, places that can provide them seating or leisure activities.

[74] The arrival flow is not allowed to use the exit on the left hand side because this may cause obstruction of the movement towards the departures.

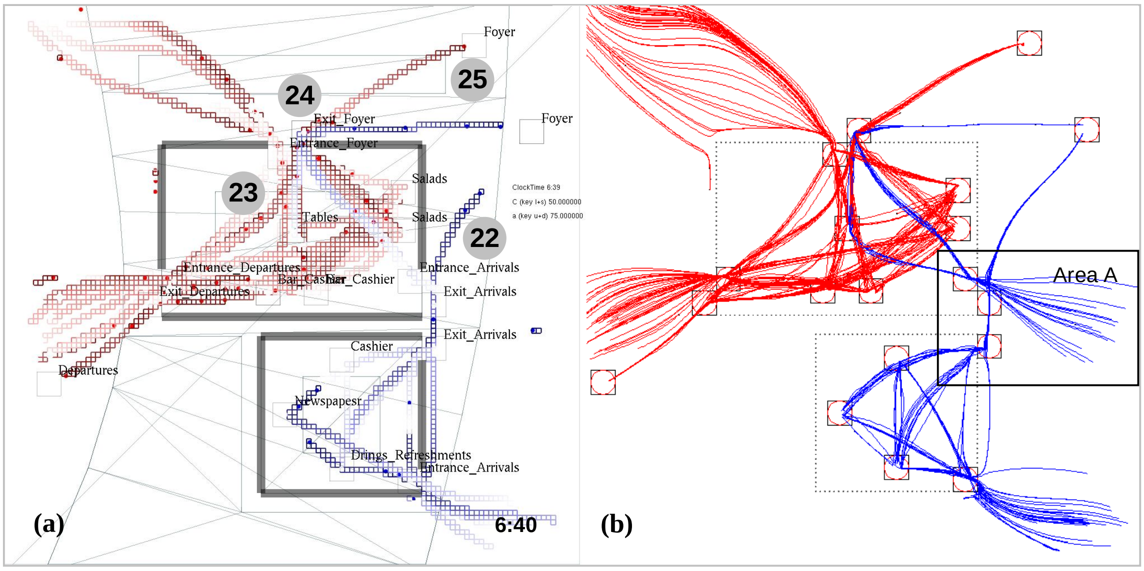


Figure 6. 83: Elaborated design and computer-generated circulation diagrams (see Appendix F.8)

The area A (see figure 6.83b) contains a possible intersecting phenomenon between the two arrival flows due to the proximity of the new restaurant entrance and the shop exit. A possible solution might be the relocation of the entrance in an upper part of the same side in order to provide enough space for passenger reaction to avoid collision. Another suggestion is to use only one entrance on the top side of restaurant area for both the departure and the arrival flows.

ii. Configuration of facilities

Figure 6.84 shows the configuration of facilities using the third approach of modelling.

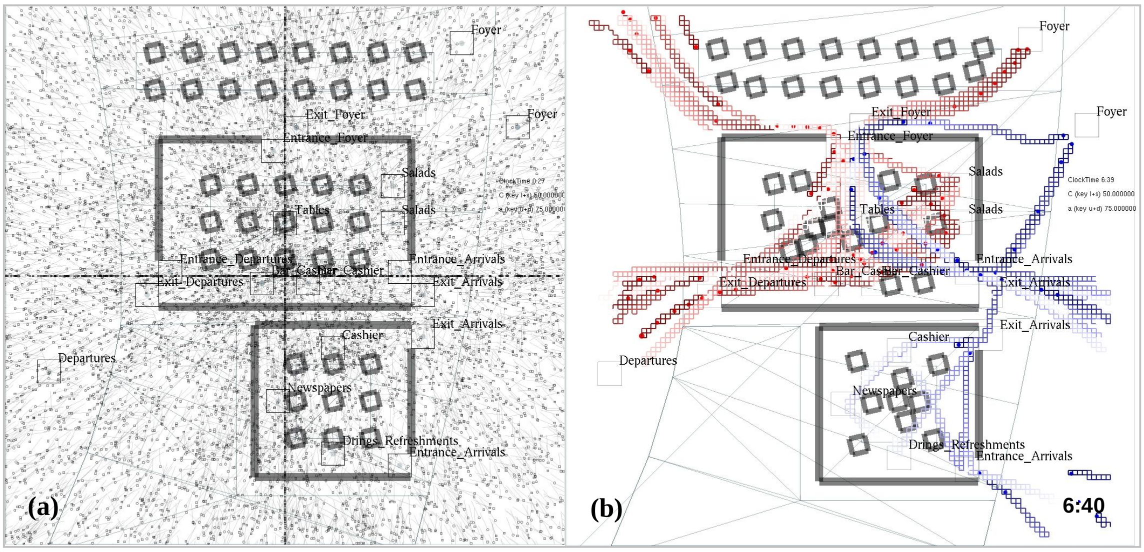


Figure 6. 84: a. Initial positioning of facilities and signs, b. Final simulated results (see Appendix F.10)

Figure 6.84a shows the initial positioning of the facilities according to the modified design scheme. Also, the image shows the random distribution of signs.

Figure 6.84b shows the final configuration of the facilities that has been derived from the interaction between different 'individuals' in real time. The arrangement of facilities can vary, depending upon a large number of factors. These factors include the position of passengers and their behaviour, the time of interaction with tables and boundaries, the initial configuration of facilities and so on.

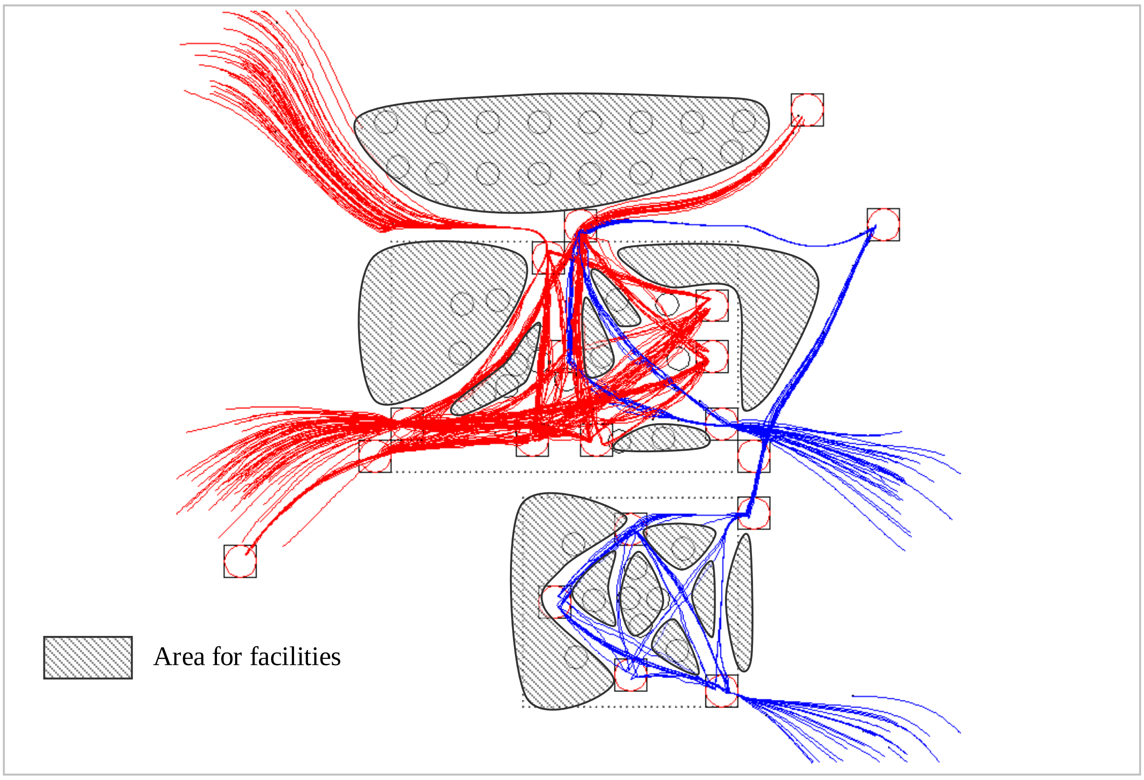


Figure 6. 85: Proposed design outline in restaurant area

Figure 6.85 shows an attempt to propose new interior outline based on the computer-generated circulation diagrams and the configuration of facilities.

6.7 Discussion of results

The case study of ferry terminal has been examined at three different scales. In the macro-scale, path systems of the entire design scheme have been generated. In the meso-scale, 'localized models' have been used for the detailed configuration of functions in the foyer area. Finally, in the micro-scale again 'localized models' have been examined for the spatial configuration of facilities in the restaurant zone of foyer.

6.7.1 Passenger movement behaviour modelling and design

The case studies showed that any attempt to model human movement behaviour precisely would demand the use of a large number of rules with tremendous complexity. The current modelling used simple rules based on a large number of human movement assumptions because the model is mainly used as a tool to assist design and not to mimic exactly the behaviour of real people.

Rules were categorized in two basic parts:

i. The first part investigated the interaction between the passengers by applying the repulsive effect.

ii. The second part investigated rules of navigation such as the sign effect or route choice mechanisms such as individual's list of destinations

In the first part, the rules were based on local interactions without the overall knowledge of the behaviour in advance. In the second part, the rules of navigation were based on the globally pre-determined and random organization of signs and destination lists.

The generation of circulation patterns was influenced by both approaches to a greater or lesser extent. It is believed that the combination of the rules at a local and global level distinguishes the current process from other applications. Other approaches take into account either only the local interactions between individuals or they are based on the global spatial configuration approaches.

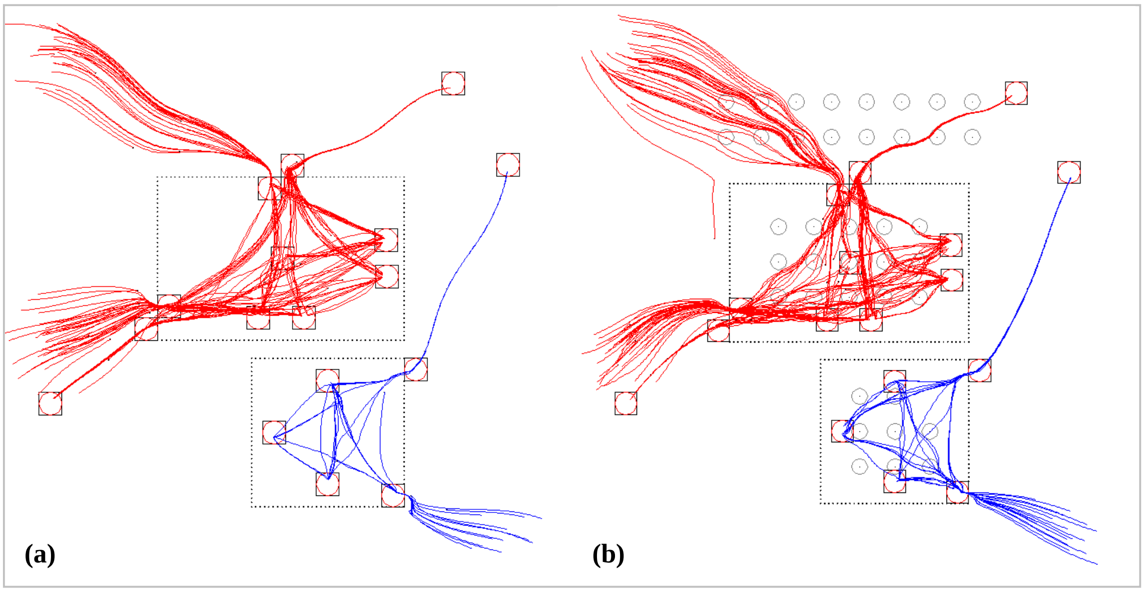


Figure 6. 86: Generated patterns based on global and local levels of interactions

Figure 6.86 shows two examples from the micro-scale where the parallel application of local and global approaches to modelling is examined. Figure 6.86a shows circulation diagrams that were generated using the passenger route choice movement behaviour between different destinations and the interaction with the signs.

Figure 6.86b shows circulation diagrams that were generated using passenger movement (see figure 6.86a) and interaction behaviour between passengers and facilities (stationary objects). In this case, passengers try to avoid collision with facilities, thus their movement is influenced by the location of facilities at a local level. After avoiding collision with the stationary objects the passengers do not re-adjust their movement line at a global level but they remain on the same track. Any changes do not affect the overall structure of path systems. This shows that the passengers are ‘automata’ without any intelligence. The degree of intelligence depends entirely on the rules applied and the sophistication of the algorithm.

Whether or not such behaviour can be characterized as ‘emergent’ is open to debate. Clearly, every time the program runs different unpredictable patterns can be generated since the initial positioning of elements such as signs are based on a random number generator. However, the application of route choice rules has shown that the global control of movement behaviour in the form of compulsory or optional list of destinations presupposes an overall knowledge of the environment.

Overall, our attempt was to apply as simple as possible behaviour rules that would achieve a certain degree of complexity. However, as the degree of complexity in each case study has been increased, the computer programs have become large because additional rules have been added. This, in turn, slows down the running of the programs in real time, particularly as the number of people or agents is increased.

Therefore, the rules have to remain as simple as possible. It is better for a short program to be used addressing a specific aspect of design, instead of creating a large program that can be used to solve a range of design tasks but to be less effective for a specific design purpose.

6.7.2 Design process

Here we have examined the use of computers to analyse circulation at three scales, macro, meso and micro. There may be times when the results of a micro-scale analysis will feed back to the overall design of a building, but that is not the way that people work. Instead they work at the macro scale and only when decisions have been frozen at this scale do they proceed to the meso and then the micro-scale.

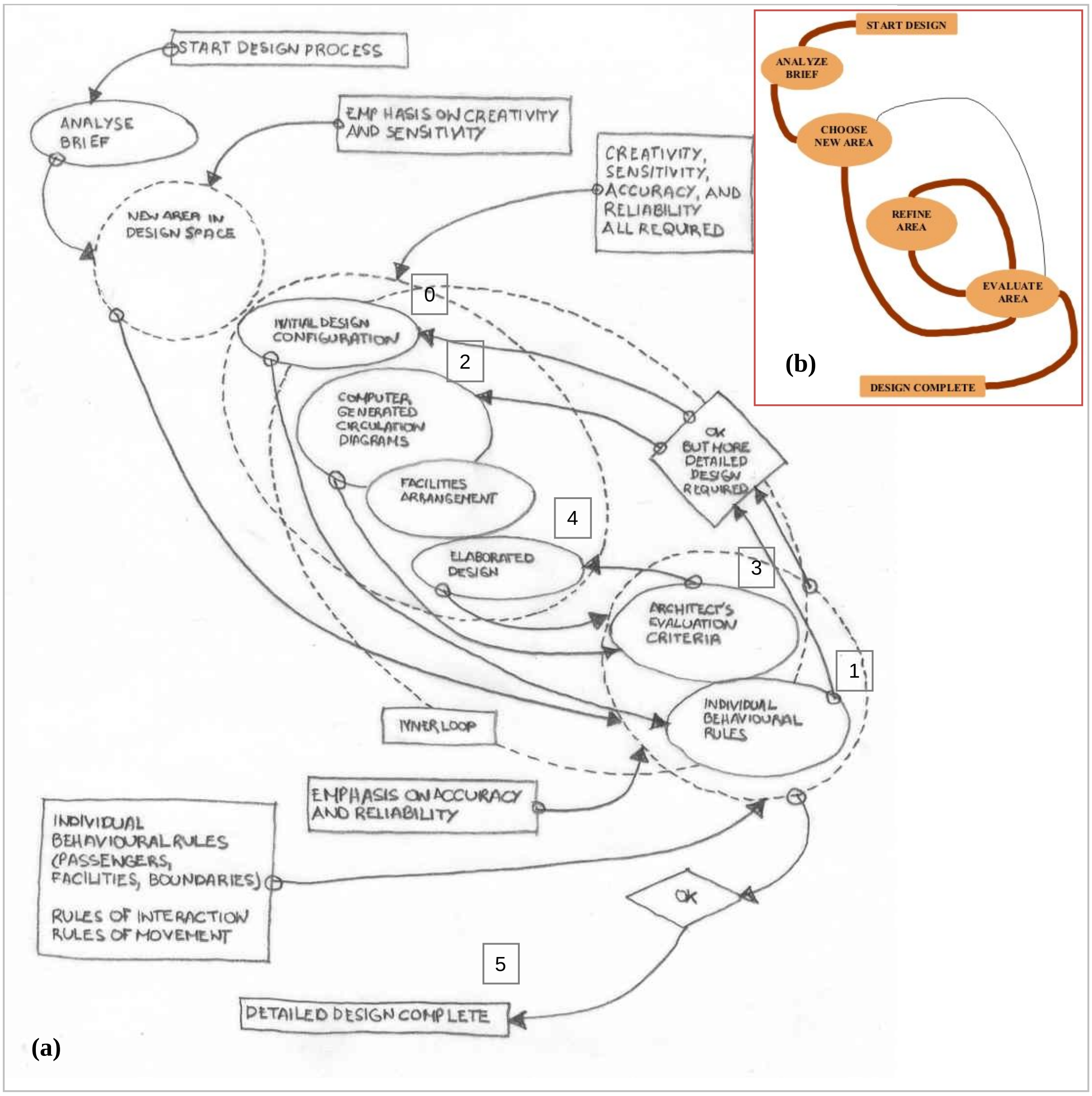


Figure 6. 87: a. Analytical inner loop mechanism of design framework, b. Design Framework

Figure 6.87 shows the inner loop mechanism of the ‘detailed design’ where the evaluation part consists of the computer behaviour rules (step 1) and the architect’s evaluation criteria (step 3) that are influenced by creativity.

Chapter 7: Conclusions

7.1 Computer programs in the design process

It was initially hypothesized that:

> 'Architects will at times need to be actively involved in computer programming by writing or modifying software'

The hypothesis was based on the assumption that in order for architects to have complete control over their design that identify their personal aesthetic language, they should have the same control over the design process including the way computer software tools are used.

A Design Framework was constructed at the beginning of this work as an attempt to understand the design process (see figure 7.1). The research used this framework to locate different applications of computer programs and see their effect on the overall design process. It was found that computer programs cannot be used independently in design but need to be parts of a wider design process because their ability to solve entirely a design problem is severely limited.

The architect is the person responsible for addressing design problems by defining them explicitly and any decision to be taken in the 'scheme' or in the 'detailed' design stage cannot be taken without his or her overall control. Also, he or she is the one who should select and apply computer programs and decide their role in the design process[75]. Thus, computational methods may occupy different positions in this framework.

Programs can be used for:

i. Representation – specifying an area in design space
ii. Evaluation of the suitability of the area in design space
iii. Emergence – the generation of new, 'unexpected' forms which, in turn need Representation and Evaluation

---

[75] Although architects have passed (abdicated) responsibility for the selection of technical software to engineers.

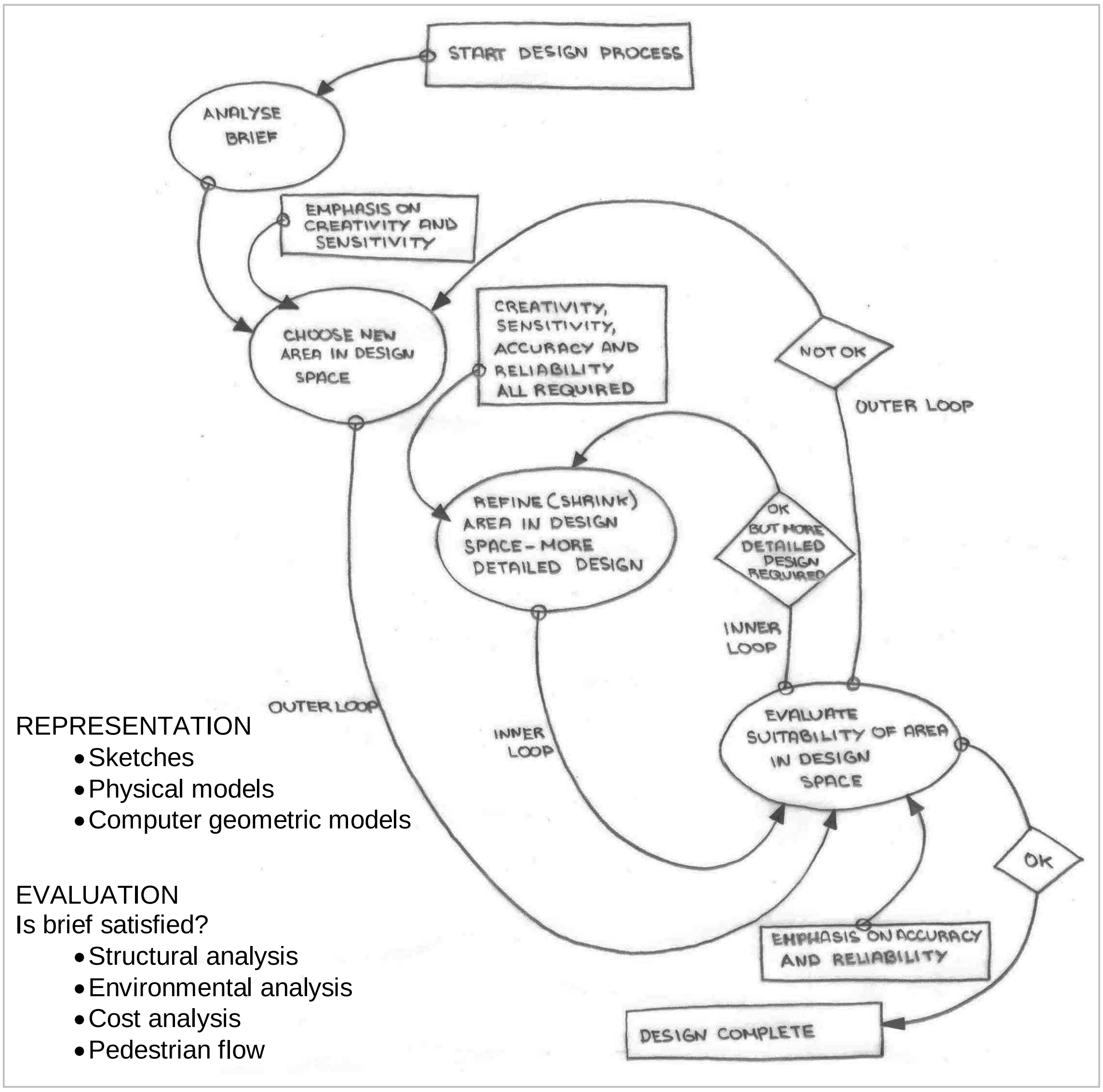


Figure 7. 1: Design Framework

Representation is the part of Design Framework where most architects are actively involved. Commercial CAD software packages are used as digital drawing boards. Programs for geometric modeling have the ability to incorporate parametric descriptions and provide a scripting environment. They allow architects to create and modify their ‘scheme’ or ‘detailed’ design and to establish their own geometrical rules.

Evaluation is the least controversial part of the application of computers to design because is the stage where the creativity is over and the computer can be used for the rational appraisal of the design proposals. Structural, building physics and cost analysis, pedestrian simulation and so on can be used in the evaluation of a proposed design.

The role of architects and engineers in the Design Framework is different. If something needs to be decided purely by evaluation (hand or computer calculations) then it might be better to be done by an engineer rather than an architect. However, architects are gradually becoming more involved in the evaluation stage by, for example, using structural analysis programs. Such programs are easier to learn than ‘traditional’ hand calculation methods and they presuppose advance geometric modeling skills that are mainly learnt by architects. Hence the person who produces geometric models will effectively also control structural, environmental or other models as well.

Recently programs used by architects for representation are becoming integrated with evaluation tools used for the structural, light and wind analysis of the performance of buildings. Such programs allow three dimensional modeling and design modification. However, this may decrease the architect’s understanding of issues like structural and wind analysis since the evaluation is done automatically by the program itself.

Architects and students need to be actively involved in parametric programming, at least to the level of understanding its methods, but also they need to be aware of its pitfalls and limitations. What parametric tools can do is to help architects to increase their geometrical awareness by offering them the possibility of constructing computer models based on their own geometrical rules.

Our research is particularly interested in the third part, Emergence. In this part, computer programs can be used as feedback loop mechanisms that alternate between creativity and evaluation stages repeatedly in order to generate unexpected and tentative design results that can be further elaborated. Such procedures are found essential in the design process since allow architects to navigate (scheme design) and refine (detailed design) solutions in design space.

7.2 Architects and programming

If the solution of a design problem is too demanding to be carried out by ‘classical’ methods, the use of computer programs becomes necessary. Architects may have better control over their design by writing their own programs as part of an emergent design procedure. This will allow them to investigate different design possibilities and provide them with the freedom of selecting their own rules.

The rule-making process is an important computational activity that needs the knowledge of computer programming. One possibility is for the architects to write their own complete programs but this means that programming will depend entirely on their ability to select the appropriate rules and to be involved in the algorithmic design procedure. Another possibility is for the architects to have a close collaboration with the person responsible to write those algorithms, who might have been trained as an architect, engineer, mathematician, physicist or mathematician.

At a recent conference [76] there was a session with a speaker from Walt Disney Animation Studios and a speaker from Électricité de France (EDF) who described the application of computers for the modeling and simulation of nuclear power stations. These two industries represent two extremes. In the first appearance is all and accuracy and reliability are unimportant, in the second this situation is reversed. Architecture is somewhere between these two extremes. The two extremes do learn from each other, so that animators are using the finite element method to make their simulations look ‘realistic’, while methods developed in the animation studios are being adapted for real engineering problems.

One of the problems associated with the use of computers for analysis is that it removes the need for a logical mental model of a design. This has already happened for structural design of buildings of the willful school of architecture and may happen for circulation design.

7.3 The computer program for circulation design

The current research investigates such computational procedures. It uses the Design Framework to locate various programs that were written for the design of circulation diagrams in a ferry terminal building and examine their contribution to the overall design process.

The results show that the role of computer programs in circulation design varies according to the scale of investigation and the design tasks. The programs can be used to emphasize the creative or the evaluation part of the design process. They can be used at the scheme design stage when architects navigate design space looking for a satisfactory design solution. At this stage, the emphasis is on creativity and the design emerges out of the process. Later, at the detailed design stage where the areas of investigation in design space are refined or are shrunk in order to reach ‘better’ design solutions, the emphasis is on creativity, accuracy, and reliability.

The contribution of programs as purely creative design tool was found to be limited, but they are more effective as part of the circulation design decision-making mechanism. Its limitation as a design tool is mainly due to the nature of circulation diagrams and their

---

[76] SGP 2008, the Sixth Symposium on Geometry Processing, July 2 to 4, 2008, Copenhagen, Denmark.

application in layout design. Circulation diagrams are used as mediums for further plan development by organizing and linking functional areas and spaces. They provide networks of relations and connections and they are basically used as conceptual diagrams where design ideas can be improved.

The simulation process produces circulation patterns that people might use within a building. It is not possible or desirable to specify exactly how people will use a building. The only exception to this is under extreme emergency situations, and even then it should be acknowledged that people may behave unpredictably.

The programs are used as generative and evaluation tool and do not attempt to replace commercial pedestrian simulation software but to give indications of the movement conditions that may cause major (ultimate) or minor (serviceability) circulation problems from the beginning of the design process.

The programs can provide architects with an awareness of circulation design possibilities to improve the design of public and especially terminal buildings in order to fulfil aesthetic and functional criteria.

Design decision-making and aesthetic judgement cannot be undertaken solely by computers. There is no such thing as artificial intelligence, if intelligence is taken to mean understanding of the real world. Thus, architects must continue to have overall control of the design process including the computational approaches used at different design decision-making levels according to the problem under investigation.

Overall, the architect is the person to make decisions at all levels of the design process. Each stage of this process develops according to the architect’s sensitivity, creativity, knowledge and experience and that of other architects and engineers working on the project. Each of the members of the design team will use computer software in their work, and it is important that the architect understands the limitations and possibilities of the software.

If the aim of this research had been to simply produce pedestrian simulation software, there is no doubt that it would have been best done by someone with an engineering, physics, mathematics or computer science background. However, the aim has been wider than simply produce software, it has been to examine the way in which software and the way it is written might influence architecture in the future.

References

Alexander, C., 1964. *Notes on the synthesis of form.* Cambridge: Harvard University Press.

Alexander, C., 1977. *A pattern language towns buildings, construction.* New York: Oxford University Press.

Anfam, D., 1990. *Abstract Expressionism*. London: Thames and Hudson

Arthur, P. and Passini, R., 1992. *Wayfinding: People, Signs, and Architecture.* New York: McGraw-Hill.

Bachelard, C., 1958.*The poetics of space*. 1994 ed. Boston, Massachusetts: Beacon Press.

Ballantyne, A., 2007.*Deleuze and Guattari for Architects.* London: Routledge.

Barkhardt, R. W., 1977. *The spirit of system: Lamarck and evolutionary biology.* Cambridge, Massachusetts; London: Harvard University Press.

Bates, M. J., 2006. 'Fundamental Forms of Information', *Journal of the American Society for Information Science and Technology, 57*(8), 1033-1045.

Batty, M, 2003. Agent-Based Pedestrian Modelling. *Centre for Advance Spatial Analysis: Working paper series.* London: UCL.

Batty, M., 2005. *Cities and Complexity: Understanding Cities with Cellular Automata, Agent-Based Models, and Fractals*. Cambridge, Massachusetts; London: MIT Press.

Bernard, C., 1995. *Earth Moves: The furnishing of territories.* Cambridge, Massachusetts; London: MIT Press.

Berkel, B. and Bos, C., 1999. *UN Studio – Move: 3 Volume Set: Imagination/ Techniques/Effects*. Netherlands: Goose Press.

Blow, C., 1996. *Airport Terminals.* Oxford: Butterworth-Heinemann.

Bonabeau, E., Dorigo, M., Theraulaz, G., 1999. *Swarm Intelligence: From natural to artificial systems.* New York; Oxford: Oxford University Press.

Borgers, A. and Timmermans, H., 1986. A model of pedestrian route choice and demand for retail facilities within inner-city shopping area. *Geographical Analysis.* Volume 18. pp. 115-128.

Bovill, C., 1996. *Fractal geometry in architecture and design.* Boston: Birkhauser.

Braha, D. and Maimon, O., 1997. The Design Process: Properties. Paradigms, and structure. *IEEE Transactions on systems, man, and cybernetics, Part A: Systems and human.* Volume 27, No 2, pp. 146-166.

Brawne, M., 1981. *Some views on design and science.* Stockholm: Royal Institute of Technology Library.

Carranza, P. M. and Coates, P., 2000. Swarm modelling: The use of swarm intelligence to generate architectural forms. *Proceedings of the Generative Art Conference, 2000,* Milan.

Available from:

http://www.generativeart.com/2000/CARRANZA_COATES.HTM

Chandrasekaran, B., 1990. Design Problem Solving: A Task Analysis. *AI Magazine.* Volume 11, Issue 4 (Winter 1990), pp. 59-71.

Chouchoulas, O., 2003. *Shape evolution: an algorithmic method for conceptual architectural design combining shape grammars and genetic algorithms.* Bath: University of Bath.

Coates, P. and Schmid, C., 1999. Agent Based modelling. In: *Architectural Computing from Turing to 2000.* Liverpool, pp. 652-661.

Coates, P., 2000. Self organizing spatial structures. London: East London University. Available from: http://www.uel.ac.uk/ceca/cad/cad_homepage/brochure.html

Coates, P., 2004. Review paper: some experiments using agent modelling at CECA. *Proceedings of the Generative Art Conference, 2004.* Milan.

Cornford, F., M**.,** 1941. *The republic of Plato Plato translated with introduction and notes by Francis MacDonald Cornford.* Oxford: Clarendon Press.

Cross, N., 1977. *The automated architect.* London: Pion.

Cyprus Ports Authority, 2003. *Architectural design competition for the new port terminal building in Limassol Port: Competition brief.* Nicosia: Cyprus Ports Authority.

Cyprus Ports Authority, 2005. *Annual report 2005.* Nicosia: Cyprus Ports Authority.

http://www.crowddynamics.com/ (sampled Juy 2008)

Darwin, C. R., 1968. *The origin of species.* London: Penguin

Dawkins, C. R., 1986 *The Blind Watchmaker*. New York: W. W. Norton & Company, Inc.

De Beer, G., 1963. *Charles Darwin: Evolution by natural selection*. London: Nelson.

DeLanda, M., 2002. Materiality: Anexact and intense. In: L. Spuybroek, ed. *NOX: Machining Architecture* . London: Thames & Hudson, pp. 370-377.

DeLanda, M., 2002. Deleuze and the use of the genetic algorithm in architecture. In: N. Leach, ed. *Design for a digital world*. Chichester: Wiley – Academy, pp. 117-120.

Deleuze, G., 1981. *Francis Bacon: The logic of sensation*. 2003 ed. London; New York: Continuum.

Deleuze, G. and Guattari, F., 1987. *A thousand plateaus: Capitalism and Schizophrenia.*2004 ed. London; New York: Continuum.

Deleuze, G., 1988.*Foucault.*London: Athlone.

Deleuze, G., 1993. *The Fold: Leibniz and the Baroque*. London: Athlone.

Dept. of Transport, Great Britain, 1978. *The Highway Code*. London: H.M.S.O.

Dewrstyne, H., 1968. *Inside the Bauhaus*. London: Architectural Press.

Dover Harbour Board, 2007. *Planning for the next generation – Second round consultation document*. Dover: Dover Harbour Board.

Available from: www.doverport.co.uk

Dreyfus, H. and Dreyfus, S., 1989. Why computers may never think like people. *In:* T. Forester, ed. *Computers in the human context: information technology, productivity and people*. Oxford: Basil Blackwell, pp. 125-143.

Do, E.Y.L. and Gross, M.D., 2001. Thinking with diagrams in architectural design. *Artificial Intelligence Review*. Volume 15, pp. 135-149.

Düchting, H., 2000. *Wassily Kandinsky, 1866-1944: A revolution in painting*. Köln; London: Taschen.

Edwards, B., 1998. *The Modern Terminal: New approach to airport architecture*. London: E & FN Spon.

Fagerstrom, G., 2007. *Tensegrity tower: A parametric/associative approach to structure, program and navigation in high rise buildings*. AAH 920 degree project in architecture. LTH-Lund Institute of Technology, Lund.

Feireiss, K., 1993. *Ben van Berkel mobile forces = mobile Kräfte.* Berlin: Ernst.

Flake, G. W., 1998. *The computational beauty of nature: Computer explorations of fractals, chaos, complex systems, and adaptation.* Cambridge; Massachusetts: MIT Press.

Flemming, U., 1986. The role of shape grammars in the analysis and creation of design. *Proceedings of Symposium on Computability of Design.* December 1986. SUNY Buffalo, pp. 245-272.

Frazer, J., 1995. *An evolutionary architecture.* London: Architecture Association.

Galea,E.R., Owen,M., Lawrence,P.J., 1996. Computer modelling of human behaviour in aircraft fire accidents. *Toxiocology.* Volume 115. pp. 63-78.

Garling, T. and Garling, E., 1988. Distance minimization in downtown pedestrian shopping. *Environment and Planning A.* Volume 20, pp. 547-554.

Gero, J., S., 2002.Advances in it for building design. In: M. Anson, J. Ko, and E. Lam, ed. *Advances in Building Technology*. Elsevier, Amsterdam. pp. 47-54.

Gibson, James J., 1986.*The ecological approach to visual perception James J. Gibson.* Hillsdale, N.J.: Erlbaum.

Goldstein, J., 1999. Emergence as a Construct: History and Issues. *Emergence*, Vol. 1, Issues 1, pp. 49-72.

Golledge, R., 1995. Path selection and route preference in human navigation: a progress report. *Spatial Information Theory-Lecture Notes in Computer Science.* Volume 988, pp. 207-222.

Goonatilake, S., 1991. *The evolution of information: Lineages in gene, culture and artefact.* London: Pinter

Grassé, P.-P., 1959. in Insectes Sociaux VI. 79

Harley, J., 2004. *Xenakis: his life in music.* New York; London: Routledge.

Helbing,D., 1992. A fluid-dynamic model for the movement of pedestrians. *Complex Systems.* 6. pp. 391-415.

Helbing, D., 1994. A mathematical model for the behaviour of individuals in social field. *Journal of Mathematical Sociology.* Volume 19, pp. 189-219.

Helbing, D., and Molnar, P., 1995. Social force model for pedestrian dynamics. *Physical Review E*, Volume 51, pp. 4282-4286.

Available from: http://arxiv.org/PS_cache/cond-mat/pdf/9805/9805244.pdf

Helbing, D., and Molnar, P., 1997. Self-organization phenomena in pedestrian crowds. F. Schweitzer, ed. *Self-organization of Complex Structures: From Individual to Collective Dynamics*, Gordon and Breach, London, U.K., pp. 569-577.

Available from: http://www.citebase.org/fulltext?format=application%2Fpdf&identifier=oai%3AarXiv.org%3Acond-mat%2F9806152

Helbing, D., Keltsch, J. and Molnar, P., 1997a Modelling the evolution of human trail system. *Nature.* July 1997. Volume 388, pp. 47-50.

Helbing, D., Schweitzer, F., Keltsch, J. and Molnar, P., 1997b Active walker model for the formation of human and animal trail systems. *Physical Review E.* September 1997. Volume 56, Number 3, pp. 2527-2539.

Helbing,D., Molnar,P., Farkas,I.,Bolay,K., 2001. Self-organizing pedestrian movement. *Environment and Planning B: Planning and Design.* Volume 28, pp. 361-383.

Helbing, D., Buzna, L., Johansson, A., Werner, T., 2005. Self-Organized Pedestrian Crowd Dynamics: Experiments, Simulations, and Design Solutions. *Transportation Science*, Vol. 39, No. 1, pp. 1-24.

Hillier, B., Penn, A., Hanson, J., Grajewski, T. and Xu, J., 1993. Natural movement: or, configuration and attraction in urban pedestrian movement. *Environment and Planning B: Planning and Design.* Volume 20, pp. 29-66.

Hillier, B., Major, M. D., Desyllas, J., Karimi, K., Campos, B., Stoner, T., 1996. Tate Gallery, Milbank: a study of the existing layout and a new masterplan proposal. Technical report, Bartlett School of Graduate Studies, University College London, London.

Holland, John H. 1975. *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor

Holland, J. H., 1992. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control and artificial intelligence*. Cambridge, Massachusetts; London: MIT Press.

Hoogendoorn, S. P., Bovy, P. H. L., Daamen, W., 2001. Microscopic Pedestrian Wayfinding and Dynamics Modelling. In: M. Schreckenberg, S. Sharma, eds. *Pedestrian and Evacuation dynamics.* Springer, Heidelberg. pp. 123-154.

Hubbard, J., 2000. *Shaum's outline of theory and problems of programming with C++.* New York; London: McGraw-Hill.

Jacobs, J., 2000. *The death and life of the great American cities.* London: Pirnlico.

Jencks, C., 1971. *Architecture 2000: Predictions and methods.* London: Studio Vista.

Johnson, S., 2001. *Emergence*. London: Penguin.

Jormakka, K., 2002. *Flying Dutchmen: motion in architecture.* Basel; Boston; Berlin: Birkhäuser.

Kandinsky, W., 1977.*Concerning the spiritual in art*. New York: Dover Publications.

Kandinsky, W., 1979. *Point and line to plane*. New York: Dover Publications.

Kaklamani, M., Kontovourkis, O., Pavlou, C., 1999. *Laboratory centre for the conservation of ancient ship wrecks.* Diploma Thesis (Dip. Arch.). National Technical University of Athens, Athens.

Klee, P., 1953. *Pedagogical sketches*. London: Faber and Faber.

Kontovourkis, O., 2006. *A conceptual design methodology using computers.* Report submitted for transfer from MPhil to PhD. University of Bath, Bath.

Krause, J., 1997. Agent Generated Architecture. *ACADIA '97 Conference Proceedings.* Cincinnati, Ohio. pp. 63-70.

Krawczyk, R., 2002. Architectural interpretation of Cellular Automata. *Proceedings of the Generative Art Conference*, December 2002, Milan.

Kurose, S., Borgers, A., Timmermans, H.J.P., 2001. Classifying pedestrian shopping behaviour according to implied heuristic choice rules. *Environment and Planning B: Planning and Design,* Volume 28, pp. 405-418.

Landau, E., 1989. *Jackson Pollock*. London: Thames and Hudson.

Latour, A., 1991. *Louis I. Kahn: Writing, Lectures, Interviews.* New York: Rizzoli.

Le Corbusier, 1980. *The Modulor: A harmonious Measure to the human Scale Universally applicable to Architecture and Mechanics.* Translated by Peter de Francia and Anna Bostock. Cambridge, Massachusetts: Harvard University Press.

Le Corbusier, 1986. *Towards a new architecture*. New York: Dover Publications.

Lee, D., 2003. *Plato: The republic*. London: Penguin.

Leach, N., 2004. Swarm tectonics. In: Leach, N., Turnbull, D., Williams, C. *Digital Tectonics*. Chichester: Wiley, pp. 70-77.

Lewin, K., 1952.*Field Theory in Social Sciences.* London: Tavistock.

Lynch, K., 1960. *The image of the city*. Cambridge, Massachusetts; London: MIT Press.

Lynn, G., 1999. *Animate form.* New York: Princeton Architectural Press.

Lynn, G., 2004.*Folding in Architecture.* London: John Wiley.

March, L. & Steadman, P., 1971. *The geometry of environment: An introduction to spatial organization in design.* London: RIBA Publications.

Mitchell, W. J., 1996. *The logic of architecture design, computation, and cognition.* Cambridge, Massachusetts; London: MIT Press.

Monaghan, J. J., 2005. Smoothed partcile hydrodynamics. *Report in Progress in Physics.*

Moore, G. E., 1965. Cramming more components onto integrated circuits. *Electronics.* Volume 38, No 8.

Moussavi, F. and Zaera-Polo, A., 2002. *The Yokohama project: Foreign Office Architects.* Barcelona: Actar.

Neufert, E., 1980. *Architects' Data.* Berlin: Blackwell science.

Negroponte, N. and Groisser, L., 1970. URBAN5: A machine that discusses design. In: G. Moore. *Emerging methods in environmental design and planning.* Cambridge: MIT Press, pp. 105-114.

Okazaki, S., Matsushita, S., 1993. A study of Simulation Model for Pedestrian movement with Evacuation and Queuing. In: Roderick A. Smith and Jim F. Dickie, eds. *Proceedings of the International Conference on Engineering for crowd safety.* London: Elsevier.

Available from: http://www.anc-d.fukuiu.ac.jp/~sat/ECS93.pdf

Otto,F. & Rasch,B.,1995. *Finding Form: Towards an Architecture of the minimal.* Axel Menges.

Penn, A. and Dalton, N., 1994. The architecture of society: stochastic simulation of urban movement. In: N. Gilbert, J. Doran, eds. S*imulating societies: The computer simulation of social phenomena.* London: UCL Press, pp. 85-125.

Peponis, J., Zimring, C., Choi, Y.K., 1990. Finding the building in wayfinding. *Environment and behaviour.* Volume 22. pp. 555-590.

Popper, K., 1972. *Objective knowledge: an evolutionary approach.* Oxford: Clarendon Press.

Proust, M., 1964. *Swann's Way.* London: Chatto and Windus.

Resnick, M., 1994. *Turtles, termites and traffic jams.* Cambridge, Massachusetts; London: MIT Press.

Reynolds, C.W., 1987. Flocks, Herds, and Schools: A Distributed Behavioral Model. *Computer Graphics,* 21(4), pp.25-34.

Reynolds, C. W., 1999. Steering Behaviours For Autonomous Characters. *Proceedings of Game Developers Conference in San Jose, California. Miller Freeman Game Group.* San Francisco, California, pp. 763-782.

Rocker, I., 2006. When code matters. In: H. Castle ed. *Programming Cultures: Art and Architecture in the Age of Software.* London: Wiley-Academy. pp. 16-25.

Rowe, P.R., 1987. *Design thinking.* Cambridge, Massachusetts; London: MIT Press.

Scheurer, F., 2005. Turning the design process downside-up: Self-organization in real-world architecture. In: B. Martens, and A. Brown, ed. *Computer Aided Architecture Design Futures 2005.* Vienna: Springer. pp. 269-278.

Shea, K., 2004.Directed Randomness. In: N. Leach, D. Turnbull, and C. Williams. *Digital Tectonics*. Chichester: Wiley, pp. 89-101.

Simon, H. A., 1996. *The sciences of the artificial.* Cambridge; Massachusetts: MIT Press.

Spiller, J., 1961. *Paul Klee: the thinking eye*. London: Lund Humphries

Spuybroek, L., 2004. *NOX: Machining Architecture* . London: Thames & Hudson.

Steadman, P., 1979. *The evolution of designs: Biological analogy in architecture and the applied arts.* Cambridge: Cambridge University Press.

Stiny, G. and Gips, J., 1972. Shape Grammars and the Generative Specification of Painting and Sculpture. In: C.V. Freiman, ed. *Information Processing 71*, Amsterdam: North-Holland, pp. 1460-1465.

Teknomo, K., Yasushi, T., Hajime, I., 2000. Review on Microscopic Pedestrian Simulation Model. *Proceedings Japan Society of Civil Engineering Conference.* March 2000. Morioka, Japan.

Terzidis, K., 2006.*Algorithmic Architecture.* Oxford: Elsevier.

Terzopoulos, D., 1999. Visual modelling for computer animation: graphics with a vision. *Computer Graphics.* Volume 33. pp. 42-45.

Treib, M., 1996. *Space calculated in seconds: The Philip Pavilion, Le Corbusier, Edgard Varese.* Princeton, New Jersey; Chichester: Princeton University Press.

Turner, A., Penn, A., 2002. Encoding natural movement as an agent-based system an investigation into human pedestrian behaviour in the built environment. *Environment and Planning B: Planning and Design*, Volume 29, pp. 473-490.

Available from: http://eprints.ucl.ac.uk/archive/00000073/01/turner-penn-2002_Encoding_natural_movement.pdf

Turner, A., Mottran, A. and Penn, A., 2004. An ecological approach to generative design. In: J.S. Gero, ed. *Design Computing and Cognition '04.* Dordrecht, NL: Kluwer Academic Publishers, pp. 259-274.

Whitehead, B., and Eldars. M. Z., 1964. An approach to the optimum layout of single-storey buildings. *The Architects' journal.* June 1964. pp. 1373-1380.

Williams, C.J.K., and Kontovourkis, O., 2008. Practical Emergence. *In:* D. Littlefield, ed. *Space Craft: Developments in architectural computing*. London: RIBA Publishing, pp. 68-81.

Whitford, F., 1967. *Kandinsky*. London: Paul Hamlyn.

Woo, M., Neider, M., Davis, T., Shreiner, D., 1999. *OpenGL Programming Guid:. Third Edition: The official guide to learning OpenGL, Version 1.2.* Boston; San Francisco; Ney York: Addison-Wesley.

Wood, J., 1916. *Nuttall's Standard Dictionary.* London; New York: Frederick Warne and Co.

Xenakis, I., 2001. *Xenakis: Metastasis.* [Audio CD]. Paris: Le Chant du Monde.

Xenakis, I., 2001. *Formalized music: thoughts and mathematics in composition.* Hillsdale, New York: Pendragon Press.

Yessios, C., 1975. Formal Languages for Site Planning. In: C.M. Eastman, ed. *Spatial Synthesis in Computer-Aided Building Design.* New York: Wiley, pp. 147-168.

Yoon, J. S., Maher, M. L., 2005. A Swarm Algorithm for Wayfinding in Dynamic Virtual Worlds. *Proceedings of the ACM symposium on Virtual reality software and technology*. Monterey, CA, USA. pp.113-116

Available from:

http://delivery.acm.org/10.1145/1110000/1101639/p113-yoon.pdf?key1=1101639&key2=9061022511&coll=GUIDE&dl=GUIDE&CFID=695409&CFTOKEN=74950358

Appendix A

Appendix A contains the code of a simple Genetic Algorithm that is described in section 3.7.3. The program maximizes the area of a triangle with given perimeter and includes only the ‘mutation’ process. It is written in C++ and the output results are produced in .txt file. Comments can be found in various parts of the algorithm.

C++

Simple Genetic Algorithm

```
#include <iostream>
#include <cstdlib>
#include <cmath>
#include <fstream>

using namespace std;

#define m 50
#define n 10
#define MYPERIMETER 45

void CopyOffspring(void);
void MutateOffspring(void);
void FindFitness(void);

void Save(void);

int    i,j,Offspring[m][n],Population[m][n],cycle;
double multiplyer,x[m],y[m],z[m],OffspringFitness[m],PopulationFitness[m];

ofstream Values(‘Values.txt’);
ofstream Array(‘Array.txt’);

int main()
{
    srand( (unsigned)time( NULL ) );

    /*This part of algorithm creates the initial population into binary form of 1’s and 0’s. This is the search space
    because consists of coded solutions which are called genotypes. Array of solutions takes the name Offspring*/

    for ( i=0; i<m; i++ )
    {
        for ( j=0; j<n; j++ )
        Offspring[i][j]=rand()%2;  /*Array (Offspring) is populated with 1’s and 0’s in a random process*/
    }

    //This part of the algorithm calculates the fitness value of the initial population

    FindFitness();  /*This line is calling the function FindFitness(). The instructions of
                      this function are coming after the end of the int main() function*/


    /*This part of the algorithm repeats functions CopyOffspring(), MutateOffspring(), and FindFitness() a number of
    generations*/

    for(cycle=0;cycle<=10000;cycle++)  /*Repeat the same process for 10000 generations*/
    {
        CopyOffspring();  /*This line is calling the function CopyOffspring(). The instructions of this function are
                            coming after the end of the int main() function*/

        MutateOffspring();  //This line is calling the function MutateOffspring()
        FindFitness();  //This line is calling the function FindFitness()

        for ( i=0; i<m; i++ )
        {
            if(OffspringFitness[i]<PopulationFitness[i])
            {
                for (j=0; j<n; j++)
                Offspring[i][j]=Population[i][j];
                OffspringFitness[i]=PopulationFitness[i];
            }
        }
    }

    FindFitness();  //This line is calling the function FindFitness()
    Save();  //This line is calling the function Save()
    cout<<‘Finished\n’;
    cin.get();
}

void CopyOffspring(void)
{
    //This function copies offspring array and creates the population array

    for (i=0; i<m; i++)
    {
        for (j=0; j<n; j++)
        Population[i][j]=Offspring[i][j];
        PopulationFitness[i]=OffspringFitness[i];
    }
}

void MutateOffspring(void)
{
    //This function mutate offspring

    for (i=0; i<m; i++)
    {
        for (j=0; j<n; j++)
        if(rand()%30==1)
        {
            if(Offspring[i][j]==1)Offspring[i][j]=0;else Offspring[i][j]=1;
        }
    }
}
```

```
void FindFitness(void)
{
    //This function maps coded solutions into actual solutions and evaluates the offspring array according to fitness
    score

    /*First coded solutions or genotypes are mapped into actual solutions or phenotypes. In this case binary numbers
    of 1’s and 0’s are translated into decimal numbers*/

    for (i=0; i<m; i++)
    {
        x[i]=0.0;
        y[i]=0.0;
        multiplyer=1.0;
        for (j=n-6; j>=0; j-=1)
        {
            x[i]+=multiplyer*Offspring[i][j];  //Binary numbers are translated into decimal
            y[i]+=multiplyer*Offspring[i][j+5];
            multiplyer=2.0*multiplyer;
        }
    }

    /*This part of the function evaluates the offspring according to fitness score. It tries to maximize the area of
      a triangle with a given perimeter*/

    for ( i=0; i<m; i++ )
    {
        z[i]=MYPERIMETER-x[i]-y[i];

        if(x[i]<0.0||y[i]<0.0||z[i]<0.0||x[i]+y[i]<z[i]||y[i]+z[i]<x[i]||z[i]+x[i]<y[i])
        OffspringFitness[i]=0.0;
        else
        {
            OffspringFitness[i]=2.0*(x[i]*x[i]*y[i]*y[i]+y[i]*y[i]*z[i]*z[i]+z[i]*z[i]*x[i]*x[i])-
                                    (x[i]*x[i]*x[i]*x[i]+y[i]*y[i]*y[i]*y[i]+z[i]*z[i]*z[i]*z[i]);

            if(OffspringFitness[i]<=0.0)OffspringFitness[i]=0.0;
            else OffspringFitness[i] = (1.0/4.0)*sqrt(OffspringFitness[i]);
        }
    }
}

void Save(void)
{
    /*This function creates two output files. Array.txt consists the population of coded solutions. Values.txt
    consists actual solutions and the fitness value*/

    for ( i=0; i<m; i++ )
    {
        for ( j=0; j<n; j++ )
        Array<<‘[‘<<i<<‘]’<< Offspring[i][j] <<‘ ‘;
        Array<<‘\n’;
    }

    for ( i=0; i<m; i++ )
    {
        Values<<‘[‘<<i<<‘]’<<x[i]<<‘ ‘<<y[i]<<‘  ‘<<z[i]<<‘ ‘<<OffspringFitness[i]<<‘ ‘;
        Values<<‘\n’;
    }
}
```

Appendix B

Appendix B contains two simple computer codes, the first using C++ with OpenGL and the second the Processing programming language. The codes include their basic functions and their overall syntax. The flowchart of the computer code consists of two main functions, the *main()* and *Draw()* in C++ with OpenGL and *setup()* and *draw()* in Processing. The organizational diagram of main functions is illustrated analytically in section 5.5.

All other functions are used for OpenGL graphics, which include mouse click and drag, key functions, rotate, zoom and shift view. The use of OpenGL in Processing is possibly easier than in C++. Large part of the program that deals with OpenGL graphics in C++ such as *Graphics(R,G,B)* function can be replaced in Processing only by *import processing.opengl.** command.

Outputs can be illustrated either while the program runs or when the program is terminated. The outputs can be drawing files (.dxf) or image files (.tga, .jpg, etc) and they can be used in AutoCAD or other CAD software package for further elaboration.

C++

Simple computer program

```
#include <iostream>
#include <fstream>
#include <cmath>
#include <cstdlib>
using namespace std;
#include <GL/glut.h>    // For PC or Sun
#define   Macintosh 0   //1 for Macintosh any other number for PC or Sun
#define AbsoluteLastIndividual 5000
#define HalfWidth     360
#define HalfHeight    360

void Graphics(float BackGroundRed,float BackGroundBlue,float BackGroundGreen);
void DoTheDrawing(void);
static void Draw(void);
void CalculateMotion(void);
void CreateData(void);

float  Coord[2][AbsoluteLastIndividual+1],Radius[AbsoluteLastIndividual+1],
       Vely[2][AbsoluteLastIndividual+1],deltat,Colour[3][AbsoluteLastIndividual+1],Boundary[2];
int    Individual,LastIndividual,xyz,i,j,m,n;
int main(void)
{
    deltat=1.0;
    CreateData();
    cout<<"This program runs forever, until 'q' is pressed";
    Graphics(1.0,1.0,1.0);
    return 0;
}
static void Draw(void)
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    CalculateMotion();
    DoTheDrawing();
}
void Graphics(float BackGroundRed,float BackGroundGreen,float BackGroundBlue)
{
    glutInitWindowSize(2*HalfWidth,2*HalfHeight);
    glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE | GLUT_DEPTH);
    glutCreateWindow("Simple Template");
    glClearColor(BackGroundRed,BackGroundGreen,BackGroundBlue,0);
    gluOrtho2D(-HalfWidth,HalfWidth,-HalfWidth,HalfWidth);
    glClear(GL_COLOR_BUFFER_BIT);
    glutIdleFunc(Draw);
    glutDisplayFunc(Draw);
    glEnable(GL_POINT_SMOOTH);
    glutMainLoop();
}
void DoTheDrawing(void)
{
    for(Individual=0;Individual<=LastIndividual;Individual++)
    {
        glColor3f(Colour[0][Individual],Colour[1][Individual],Colour[2][Individual]);
        glPointSize(Radius[Individual]);
        glBegin(GL_POINTS);
        glVertex2f(Coord[0][Individual],Coord[1][Individual]);
        glEnd();
    }
    glutSwapBuffers();
}
void CalculateMotion(void)
{
    for(Individual=0;Individual<=LastIndividual;Individual++)
    {
        for(xyz=0;xyz<=1;xyz++)
        {
            Coord[xyz][Individual]+=Vely[xyz][Individual]*deltat;
        }
    }
}
void CreateData(void)
{
    Boundary[0]=HalfWidth;
    Boundary[1]=HalfWidth;
    Individual=-1;
    m=30,n=30;
    for(i=0;i<=m;i++)
    {
        for(j=0;j<=n;j++)
        {
            Individual++;
            Vely[0][Individual]=0.0;Vely[1][Individual]=0.0;
            for(xyz=0;xyz<=1;xyz++)
            {
                Coord[xyz][Individual]=Boundary[xyz]*((2.0*rand())/(1.0*RAND_MAX)-1.0);
            }
            Radius[Individual]=(10.0*rand())/(1.0*RAND_MAX);
            Colour[0][Individual]=0.0;Colour[1][Individual]=0.0;Colour[2][Individual]=1.0;
        }
    }
    LastIndividual=Individual;
}
```

Processing

Simple computer program

```
//http://www.processing.org/
//SimpleTemplate_070719a
import processing.opengl.*;
int AbsoluteLastIndividual=5000;
int Individual,LastIndividual,xyz,i,j,m,n,otherIndividual,NewLastIndividual;
float [][] Coord,Vely,Colour;
float [] Radius,Boundary;
float deltat;
void setup()
{
    size(720,720,OPENGL);
    ellipseMode(CENTER_RADIUS);
    noStroke();
    smooth();
    Coord=new float[2][AbsoluteLastIndividual+1];
    Radius=new float[AbsoluteLastIndividual+1];
    Vely=new float[2][AbsoluteLastIndividual+1];
    Colour=new float[3][AbsoluteLastIndividual+1];
    Boundary=new float[2];
    InitialValues();
    frameRate(60);
}
void InitialValues()
{
    println("This program runs forever until 'esc' is pressed");
    deltat=1.0;
    Boundary[0]=float(width)/2;
    Boundary[1]=float(height)/2;
    Individual=-1;
    int m=30,n=30;
    for(i=0;i<=m;i++)
    {
        for(j=0;j<=n;j++)
        {
            Individual++;
            Vely[0][Individual]=0.0;Vely[1][Individual]=0.0;
            for(xyz=0;xyz<=1;xyz++)
            {
                Coord[xyz][Individual]=Boundary[xyz]*(random(0,2)-1);
            }
            Radius[Individual]=random(0,5);
            Colour[0][Individual]=0;Colour[1][Individual]=0;Colour[2][Individual]=255;
        }
    }
    LastIndividual=Individual;
}
void draw()
{
    background(255,255,255);
    Calculation();
    stroke(255,255,255);
    for(Individual=0;Individual<=LastIndividual;Individual++)
    {
        fill(Colour[0][Individual],Colour[1][Individual],Colour[2][Individual]);
        ellipse(float(width)/2.0+Coord[0][Individual],float(height)/2.0-
Coord[1][Individual],Radius[Individual],Radius[Individual]);
    }
}
void Calculation()
{
    for(Individual=0;Individual<=LastIndividual;Individual++)
    {
        for(xyz=0;xyz<=1;xyz++)
        {
            Coord[xyz][Individual]+=Vely[xyz][Individual]*deltat;
        }
    }
}
```

Appendix C

Appendix C contains the computer code of the string effect using C++ with OpenGL. The string effect is explained analytically in section 5.6.2.5. Comments on the calculation of motion are given in different parts of the code. Also, in this case the individuals are described as nodes.

C++ with OpenGL

String effect

```
//#include <GLUT/glut.h>  // For Macintosh
#include <GL/glut.h>    // For PC or Sun
#define   Macintosh 1   //1 for Macintosh any other number for PC or Sun

#include <iostream>
#include <fstream>
#include <cmath>
#include <cstdlib>
#include <time.h>
using namespace std;

#define   halfW          508
#define   halfH          360
#define   MaxLastString 1000
#define   MaxLastNode    999
#define   MaxLastPoint 99999
#define   Lastvertex      12
#define   MaxLastZoneH    50
#define   MaxLastZoneV    36
#define   MaxLastPointInZone   500

void CalculationMotion(void);
void Graphics(float BackGroundRed,float BackGroundBlue,float BackGroundGreen);
static void Draw(void);
static void Key(unsigned char key, int x, int y);
void WriteDXF(void);
void Boundary(void);
void EdgeCentre(void);
void EdgeRadius(void);
void LakeCentre(void);
void LakeRadius(void);

int argc;
char **argv;

GLUquadricObj *quadObj;

int   LastNode[MaxLastString+1],
      NodePoint[MaxLastPoint+1],
      StringPoint[MaxLastPoint+1],
      InZone[MaxLastZoneH+1][MaxLastZoneV+1][MaxLastPointInZone+1],
      LastInZone[MaxLastZoneH+1][MaxLastZoneV+1],
      HalfDimension[2],
      Zone[2],
      LastZone[2],
      otherZone[2],
      StartZone[2],
      StopZone[2],
      ZoneSize,
      Point,
      LastPoint,
      otherPoint,
      LastString,
      String,
      Node,
      otherString,
      otherNode,
      vertex,
      i,
      dxfColour,
      EdgeOrLake,
      NumberOfBoundaryLines,
      m,n;

float x[2][MaxLastString+1][MaxLastNode+1],
      Force[2][MaxLastString+1][MaxLastNode+1],
      Movement[2][MaxLastString+1][MaxLastNode+1],
      rcosorsin[2][Lastvertex+1],
      drawx[2],
      xstart[2],
      xfinish[2],
      xBoundary[2],
      PI,
      CircleRadius,
      LSq,
      L,
      r,
      ColourControl,
      mydistance,
      BoundaryRadius,
      theta,
      angle,
      lengthSq,
      tensionCoefficient,
      thisforce,
      ZoneSizeSq,
      gridx[MaxLastZoneH][MaxLastZoneV],
      gridy[MaxLastZoneH][MaxLastZoneV];

ofstream StringEffect("String.dxf");

int main(void)
{
    srand(time(NULL));
    PI=4.0*atan(1.0);

    HalfDimension[0]=halfW;
    HalfDimension[1]=halfH;
    ZoneSize=30;
```

```
    ZoneSizeSq=1.0*ZoneSize*ZoneSize;

    for(i=0;i<=1;i++)
    {
        LastZone[i]=(2*HalfDimension[i])/ZoneSize;
        cout<<"Last zone["<<i<<"]"<<" "<<LastZone[i]<<"\n";
    }
    if(LastZone[0]>MaxLastZoneH){cout<<"Too many zones horizontally\n";return 0;}
    if(LastZone[1]>MaxLastZoneV){cout<<"Too many zones vertically\n";return 0;}

    LastString=100;
    CircleRadius=450.0;
    L=0.25*ZoneSize;
    LSq=L*L;
    r=L/10.0;

    Point=-1;
    for(String=0;String<=LastString;String++)
    {
        for(;;)
        {
            angle=(2.0*PI*rand())/(1.0*RAND_MAX);
            xstart[0]=2.0*CircleRadius*(cos(angle));
            xstart[1]=2.0*CircleRadius*(sin(angle));
            angle=(2.0*PI*rand())/(1.0*RAND_MAX);
            xfinish[0]=2.0*CircleRadius*(cos(angle));
            xfinish[1]=2.0*CircleRadius*(sin(angle));

            Node=0;
            x[0][String][Node]=xstart[0];
            x[1][String][Node]=xstart[1];
            for(EdgeOrLake=-1;EdgeOrLake<=1;EdgeOrLake+=2)
            Boundary();

            xstart[0]=x[0][String][Node];
            xstart[1]=x[1][String][Node];

            Node=0;
            x[0][String][Node]=xfinish[0];
            x[1][String][Node]=xfinish[1];

            for(EdgeOrLake=-1;EdgeOrLake<=1;EdgeOrLake+=2)
            Boundary();

            xfinish[0]=x[0][String][Node];
            xfinish[1]=x[1][String][Node];

            mydistance=sqrt((xfinish[0]-xstart[0])*(xfinish[0]-xstart[0])
                       +(xfinish[1]-xstart[1])*(xfinish[1]-xstart[1]));
            if(mydistance>1.5*L)break;
        }
        LastNode[String]=int(mydistance*1.3/L);
        if(LastNode[String]>MaxLastNode){cout<<"Too many nodes on string\n";return 0;}
        for(Node=0;Node<=LastNode[String];Node++)
        {
            Point++;
            if(Point>MaxLastPoint){cout<<"Too many points\n";return 0;}
            NodePoint[Point]=Node;
            StringPoint[Point]=String;
            for(i=0;i<=1;i++)
            x[i][String][Node]=xstart[i]+1.0*Node*(xfinish[i]-xstart[i])/(1.0*LastNode[String]);
        }
    }
    LastPoint=Point;
    for(vertex=0;vertex<=Lastvertex;vertex++)
    {
        rcosorsin[0][vertex]=r*cos((2.0*PI*vertex)/(1.0*Lastvertex));
        rcosorsin[1][vertex]=r*sin((2.0*PI*vertex)/(1.0*Lastvertex));
    }

    Graphics(0.0,0.0,0.0);
    return 0;
}

static void Draw(void)
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    CalculationMotion();

    glColor3f(0.75,0.75,0.75);

    NumberOfBoundaryLines=400;

    for(EdgeOrLake=-1;EdgeOrLake<=1;EdgeOrLake+=2)
    {
        if(EdgeOrLake==1)
        EdgeCentre();
        else LakeCentre();

        glBegin(GL_LINE_STRIP);

        for(i=0;i<=NumberOfBoundaryLines;i++)
        {
            theta=(2.0*PI*i)/(1.0*NumberOfBoundaryLines+1.0);
            if(EdgeOrLake==1)
            EdgeRadius();
            else LakeRadius();
            drawx[0]=xBoundary[0]+BoundaryRadius*cos(2.0*PI*i/(1.0*NumberOfBoundaryLines));
            drawx[1]=xBoundary[1]+BoundaryRadius*sin(2.0*PI*i/(1.0*NumberOfBoundaryLines));
            glVertex2fv(drawx);
        }
        glEnd();
    }
    for(String=0;String<=LastString;String++)
    {
        for(Node=0;Node<=LastNode[String];Node++)
```

```
        {
            if(Node!=LastNode[String])
            lengthSq=(x[0][String][Node+1]-x[0][String][Node])*(x[0][String][Node+1]-x[0][String][Node])
                     +(x[1][String][Node+1]-x[1][String][Node])*(x[1][String][Node+1]-x[1][String][Node]);

            ColourControl=0.7*lengthSq/LSq;
            if(ColourControl>1.0)ColourControl=1.0;
            glColor3f(ColourControl,0.8,1.0-ColourControl);
            glBegin(GL_LINE_STRIP);
            for(i=0;i<=1;i++)drawx[i]=x[i][String][Node];
            glVertex2fv(drawx);
        }
        glEnd();

        glColor3f(1.0,1.0,1.0);
        for(Node=0;Node<=LastNode[String];Node++)
        {
            glBegin(GL_LINE_STRIP);
            for(vertex=0;vertex<=Lastvertex;vertex++)
            {
                for(i=0;i<=1;i++)
                drawx[i]=x[i][String][Node]+rcosorsin[i][vertex];
                glVertex2fv(drawx);
            }
            glEnd();
        }
    }
    glutSwapBuffers();
}

void CalculationMotion(void)
{
    for(Zone[0]=0;Zone[0]<=LastZone[0];Zone[0]++)
    {
        for(Zone[1]=0;Zone[1]<=LastZone[1];Zone[1]++)
        LastInZone[Zone[0]][Zone[1]]=-1;
    }

    for(Point=0;Point<=LastPoint;Point++)
    {
        for(i=0;i<=1;i++)
        {
            Zone[i]=int((x[i][StringPoint[Point]][NodePoint[Point]]+HalfDimension[i])/ZoneSize);
            if(Zone[i]<0)
            cout<<"Negative zone, i = "<<i<<" x = "<<x[i][StringPoint[Point]][NodePoint[Point]]<<"\n";
            if(Zone[i]>LastZone[i])
            cout<<"Zone number too big, i = "<<i<<" x = "<<x[i][StringPoint[Point]][NodePoint[Point]]<<"\n";
        }
        LastInZone[Zone[0]][Zone[1]]++;
        if(LastInZone[Zone[0]][Zone[1]]<=MaxLastPointInZone)
        InZone[Zone[0]][Zone[1]][LastInZone[Zone[0]][Zone[1]]]=Point;
        else cout<<"Too many points in zone\n";
    }

    for(String=0;String<=LastString;String++)
    {
        for(Node=0;Node<=LastNode[String];Node++)
        {
            for(i=0;i<=1;i++)Force[i][String][Node]=0.0;
        }
    }

    for(String=0;String<=LastString;String++)
    {
        for(Node=0;Node<=LastNode[String]-1;Node++)
        {
            lengthSq=(x[0][String][Node+1]-x[0][String][Node])*
                     (x[0][String][Node+1]-x[0][String][Node])+
                     (x[1][String][Node+1]-x[1][String][Node])*
                     (x[1][String][Node+1]-x[1][String][Node]);

            /*This part of the program calculates the virtual attractive force (tension)
            between two nodes on the same string*/
            if(lengthSq>LSq)
            {
                tensionCoefficient=lengthSq/LSq-1.0;
                for(i=0;i<=1;i++)
                {
                    thisforce=tensionCoefficient*(x[i][String][Node+1]-x[i][String][Node]);
                    Force[i][String][Node]+=thisforce;
                    Force[i][String][Node+1]-=thisforce;
                }
            }
        }
    }
    /*This part of the program calculates the virtual attractive force between two nodes
    that are not in the same string. Nodes attract each other according to given distance
    between them*/
    for(Zone[0]=0;Zone[0]<=LastZone[0];Zone[0]++)
    {
        for(Zone[1]=0;Zone[1]<=LastZone[1];Zone[1]++)
        {
            if(LastInZone[Zone[0]][Zone[1]]>=0)
            for(Point=0;Point<=LastInZone[Zone[0]][Zone[1]];Point++)
            {
                Node=NodePoint[InZone[Zone[0]][Zone[1]][Point]];
                String=StringPoint[InZone[Zone[0]][Zone[1]][Point]];
                for(i=0;i<=1;i++)
                {
                    StartZone[i]=Zone[i]-1;if(StartZone[i]<0)StartZone[i]=0;
                    StopZone[i]=Zone[i]+1;if(StopZone[i]>LastZone[i])StopZone[i]=LastZone[i];
                    for(otherZone[0]=StartZone[0];otherZone[0]<=StopZone[0];otherZone[0]++)
                    {
                        for(otherZone[1]=StartZone[1];otherZone[1]<=StopZone[1];otherZone[1]++)
                        {
                            for(otherPoint=0;otherPoint<=LastInZone[otherZone[0]][otherZone[1]];otherPoint++)
                            {
```

```
                                otherNode=NodePoint[InZone[otherZone[0]][otherZone[1]][otherPoint]];
                                otherString=StringPoint[InZone[otherZone[0]][otherZone[1]][otherPoint]];
                                lengthSq=(x[0][otherString][otherNode]-x[0][String][Node])*
                                         (x[0][otherString][otherNode]-x[0][String][Node])+
                                         (x[1][otherString][otherNode]-x[1][String][Node])*
                                         (x[1][otherString][otherNode]-x[1][String][Node]);
                                if(lengthSq<ZoneSizeSq)
                                {
                                    for(i=0;i<=1;i++)
                                    {
                                        thisforce=(x[i][otherString][otherNode]-x[i][String][Node])/100000.0;

                                        Force[i][String][Node]+=thisforce;
                                        Force[i][otherString][otherNode]-=thisforce;
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
    /*This part of the program calculates the movement of nodes. This process models the movement of nodes
    under the action of virtual forces. A number of steps are taken through time until the system reach
    an equilibrium position*/
    for(String=0;String<=LastString;String++)
    {
        for(Node=1;Node<=LastNode[String]-1;Node++)
        {
            for(EdgeOrLake=-1;EdgeOrLake<=1;EdgeOrLake+=2)Boundary();
            for(i=0;i<=1;i++)
            {
                Movement[i][String][Node]=0.95*Movement[i][String][Node]+0.2*Force[i][String][Node];
                x[i][String][Node]+=Movement[i][String][Node];
            }
        }
     }
}

void Graphics(float BackGroundRed,float BackGroundBlue,float BackGroundGreen)
{
    glutInitWindowSize(2*halfW, 2*halfH);
    //if(Macintosh==1)glutInit(&argc, argv);// Remove this line for PC or Sun
    glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE | GLUT_DEPTH);
    glutCreateWindow("String effect");
    glClearColor(BackGroundRed,BackGroundGreen,BackGroundBlue,0);
    glViewport(0, 0, 2*halfW, 2*halfH);
    gluOrtho2D(-halfW,halfW,-halfH,halfH);
    glClear(GL_COLOR_BUFFER_BIT);

    glLineWidth(1.0);
    glPointSize(2.0);

    glutKeyboardFunc(Key);
    glutIdleFunc(Draw);
    glutDisplayFunc(Draw);
    glutMainLoop();
}

static void Key(unsigned char key, int x, int y)
{
    switch (key)
    {
        case 'q':gluDeleteQuadric(quadObj);
        cout<<"Writing files\n";
        WriteDXF();
        exit(0);
    }
}

void WriteDXF(void)
{
    StringEffect<<"0\nSECTION\n2\nENTITIES\n";

    NumberOfBoundaryLines=1000;
    for(EdgeOrLake=-1;EdgeOrLake<=1;EdgeOrLake+=2)
    {
        if(EdgeOrLake==1)EdgeCentre();
        else LakeCentre();
        StringEffect<<"0\nPOLYLINE\n8\nBoundary\n66\n1\n";
        StringEffect<<"10\n0.0\n";
        StringEffect<<"20\n0.0\n";
        for(i=0;i<=NumberOfBoundaryLines;i++)
        {
            theta=(2.0*PI*i)/(1.0*NumberOfBoundaryLines+1.0);

            if(EdgeOrLake==1)EdgeRadius();
            else LakeRadius();

            drawx[0]=xBoundary[0]+BoundaryRadius*cos(2.0*PI*i/(1.0*NumberOfBoundaryLines));
            drawx[1]=xBoundary[1]+BoundaryRadius*sin(2.0*PI*i/(1.0*NumberOfBoundaryLines));

            StringEffect<<"0\nVERTEX\n8\nBoundary\n";
            StringEffect<<"10\n"<<drawx[0]<<"\n";
            StringEffect<<"20\n"<<drawx[1]<<"\n";
            StringEffect<<"62\n"<<0<<"\n";

        }
        StringEffect<<"0\nSEQEND\n8\nBoundary\n";
    }

    for(String=0;String<=LastString;String++)
    {
        for(Node=0;Node<=LastNode[String]-1;Node++)
        {
            lengthSq=(x[0][String][Node+1]-x[0][String][Node])*
```

```
                     (x[0][String][Node+1]-x[0][String][Node])+
                     (x[1][String][Node+1]-x[1][String][Node])*
            (x[1][String][Node+1]-x[1][String][Node]);

            ColourControl=sqrt(lengthSq/LSq);

            dxfColour=int(16.0*ColourControl);

            StringEffect<<"0\nLINE\n8\n"<<int(0.1*int(20.0*ColourControl))<<"\n";
            StringEffect<<"10\n"<<x[0][String][Node]<<"\n";
            StringEffect<<"20\n"<<x[1][String][Node]<<"\n";
            StringEffect<<"11\n"<<x[0][String][Node+1]<<"\n";
            StringEffect<<"21\n"<<x[1][String][Node+1]<<"\n";
            StringEffect<<"62\n"<<int(0.1*int(20.0*ColourControl))<<"\n";
        }

        for(Node=0;Node<=LastNode[String];Node++)
        {
            StringEffect<<"0\nCIRCLE\n8\nNodes\n";
            StringEffect<<"10\n"<<x[0][String][Node]<<"\n";
            StringEffect<<"20\n"<<x[1][String][Node]<<"\n";
            StringEffect<<"40\n"<<r<<"\n";
            StringEffect<<"62\n"<<0<<"\n";
        }
    }

    for(m=0;m<=LastZone[0]-1;m++)
    {
        for(n=0;n<=LastZone[1]-1;n++)
        {
            gridx[m][n]=ZoneSize*m;
            gridy[m][n]=ZoneSize*n;
        }
    }
    StringEffect<<"0\nENDSEC\n0\nEOF\n";
    StringEffect.close();
    cout<<"DXF file written, end of program\n";
}

void Boundary(void)
{
    float CentreDistance,ScalarProduct;

    if(EdgeOrLake==1)EdgeCentre();else LakeCentre();

    CentreDistance=sqrt((x[0][String][Node]-xBoundary[0])*
                        (x[0][String][Node]-xBoundary[0])+
                        (x[1][String][Node]-xBoundary[1])*
                        (x[1][String][Node]-xBoundary[1]));
    if(fabs(x[0][String][Node]-xBoundary[0])>fabs(x[1][String][Node]-xBoundary[1]))
    {
        theta=asin((x[1][String][Node]-xBoundary[1])/CentreDistance);
        if((x[0][String][Node]-xBoundary[0])<0.0)theta=PI-theta;
    }
    else
    {
        theta=acos((x[0][String][Node]-xBoundary[0])/CentreDistance);
        if((x[1][String][Node]-xBoundary[1])<0.0)theta=-theta;
    }

    if(EdgeOrLake==1)EdgeRadius();else LakeRadius();

    if((EdgeOrLake==+1&&CentreDistance>BoundaryRadius)||
    (EdgeOrLake==-1&&CentreDistance<BoundaryRadius)&&CentreDistance>r/20000.0)
    {
        if(Node==0)//This is for string ends
        {
            for(i=0;i<=1;i++)
            x[i][String][Node]=xBoundary[i]+(x[i][String][Node]-xBoundary[i])*BoundaryRadius/CentreDistance;
        }
        else
        {
            ScalarProduct=Force[0][String][Node]*(x[0][String][Node]-xBoundary[0])
            +Force[1][String][Node]*(x[1][String][Node]-xBoundary[1]);
            for(i=0;i<=1;i++)
            {
            Force[i][String][Node]+=(x[i][String][Node]-xBoundary[i])*
                                    (-(1.0*EdgeOrLake)/(10.0*CentreDistance)-0.95*ScalarProduct/
                                    (CentreDistance*CentreDistance));
            }
        }
    }
}

void EdgeCentre(void)
{
    xBoundary[0]=0.0*CircleRadius;
    xBoundary[1]=0.0*CircleRadius;
}

void EdgeRadius(void)
{
    BoundaryRadius=0.7*CircleRadius;}

void LakeCentre(void)
{
    xBoundary[0]=0.15*CircleRadius;
    xBoundary[1]=0.3*CircleRadius;
}

void LakeRadius(void)
{
    BoundaryRadius=0.25*CircleRadius;
}
```

Appendix D

An analytical representation of the ferry terminal case study is demonstrated in the following appendices (D, E, and F). As was explained in chapter 6, the ferry terminal case study was divided in three main scales of investigation: the macro-scale where path systems were generated, the meso-scale where the design of functional areas and circulation diagrams was examined, and finally the micro-scale where the design of facilities and their spatial configuration was investigated.

The results of simulation are represented in a single page and they include:

i. General information for each study:
   1. Initial parameters that were applied for each modelling like geometrical configuration and passenger data information.
   2. Information related to the passenger behavioural rules. The passenger rules are divided into passenger interaction and passenger movement rules.

ii. Basic steps of simulation. They are presented as images that have been taken in different time intervals, and they show the passenger movement and the generated path systems.

Appendix D demonstrates the studies that have been examined in the macro-scale for the path systems design. Parameters are organized as follows:

The initial geometrical configuration specifies the virtual environment in 2D or 3D, the boundaries that are actively involved in simulation that in this case is nil, the destination’s number that is set to two with two entry points, and the number of signs that are set to 88 (the same number with functional areas). Also, the total functional area in square meters is 4280m² and finally each study presents four images taken with a time step interval 1:15 minutes (virtual time).

The passengers’ initial information includes the number of passenger groups that is set to 4, the number of passenger in each group, the number of destinations, the random or the pre-determined distribution of groups, and the time of appearance that is random.

The passenger behavioural rules include the repulsive effect, the value of variable $a_{\text{interperson repel}}$ that is set to 24 length units, and the movement speed that is set to 1.0 length/time units. Finally, the passenger movement rules include the distribution of signs and the value of sign strength C_{parallel} that is set to 50 force units.

1. Design of path systems: 3D model: isometric view

• Initial simulation

Initial parameter

Initial geometrical configuration

1. 3D Virtual Environment
2. Boundaries: 0
3. Actual last destinations: 2, Entry points: 2
4. Signs: 88
5. Total functional area in square meters: 4280m²
6. Time step interval: (150 units) 1:15 minutes virtual clock time

Passenger initial features

1. No of groups: 4
2. No of people in each group: 1st departure group 5x50, 2nd departure group 5x50, 3rd arrival group 50x5, 4th arrival group 5x50
3. No of destinations and group distribution: 6 destinations, 1st and 2nd group 4 first destinations randomly, 3rd and 4th groups 5th and 6th destination randomly
4. Entry points: 2
5. Time of appearance: Random

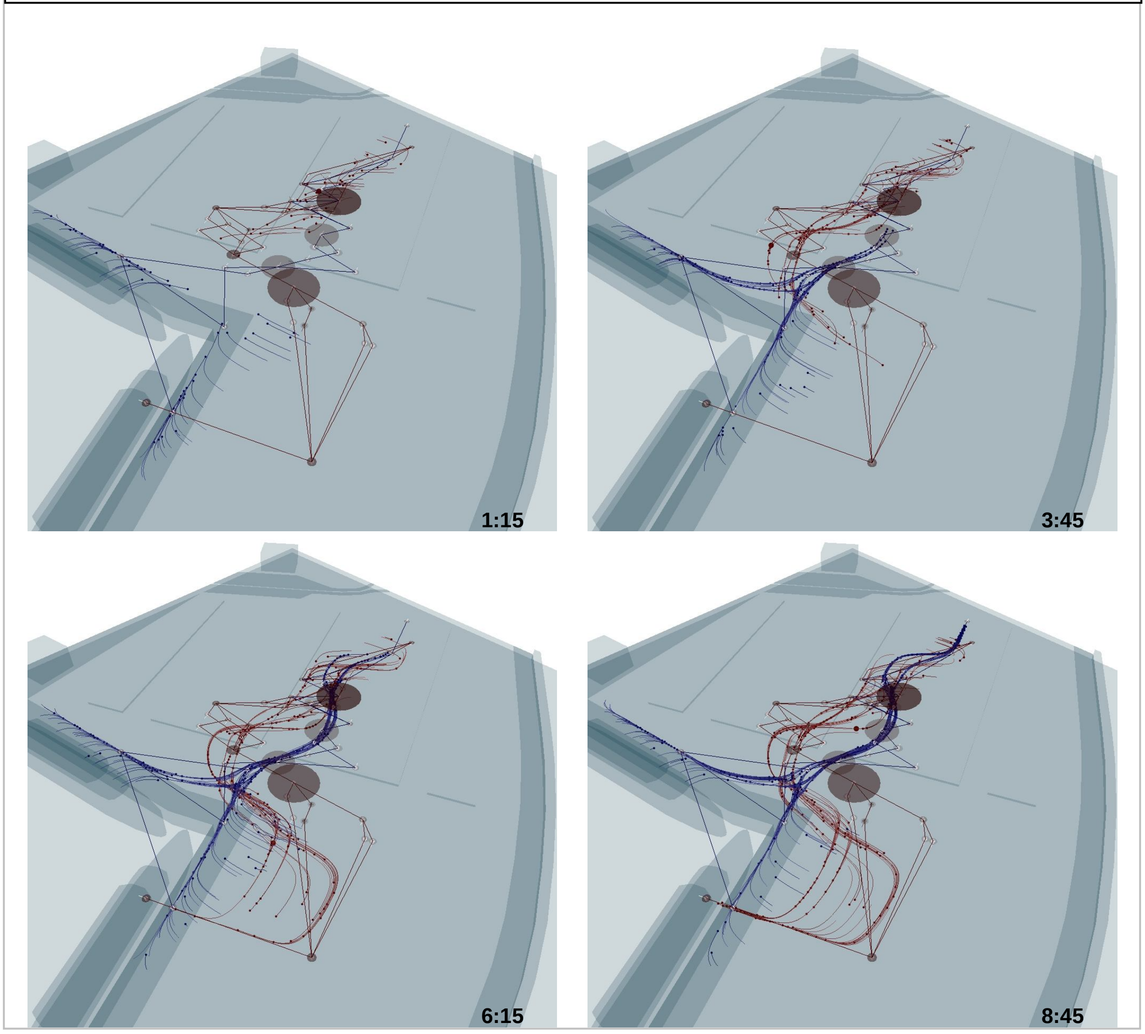
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:1.0 length/time units

Passenger movement rules

1. Sign distribution: according to destination points
2. Sign strength $C_{\text{parallel}} = 50$ force units



2. Design of path systems: 3D model: isometric view (zoom-in)

• Initial simulation

Initial parameter

Initial geometrical configuration

1. 3D Virtual Environment (zoom in)
2. Boundaries: 0
3. Actual last destinations: 2, Entry points: 2
4. Signs: 88
5. Total functional area in square meters: 4280m²
6. Time step interval: (150 units) 1:15 minutes virtual clock time

Passenger initial features

1. No of groups: 4
2. No of people in each group: 1st departure group 5x50, 2nd departure group 5x50, 3rd arrival group 50x5, 4th arrival group 5x50
3. No of destinations and group distribution: 6 destinations, 1st and 2nd group 4 first destinations randomly, 3rd and 4th groups 5th and 6th destination randomly
4. Entry points: 2
5. Time of appearance: Random

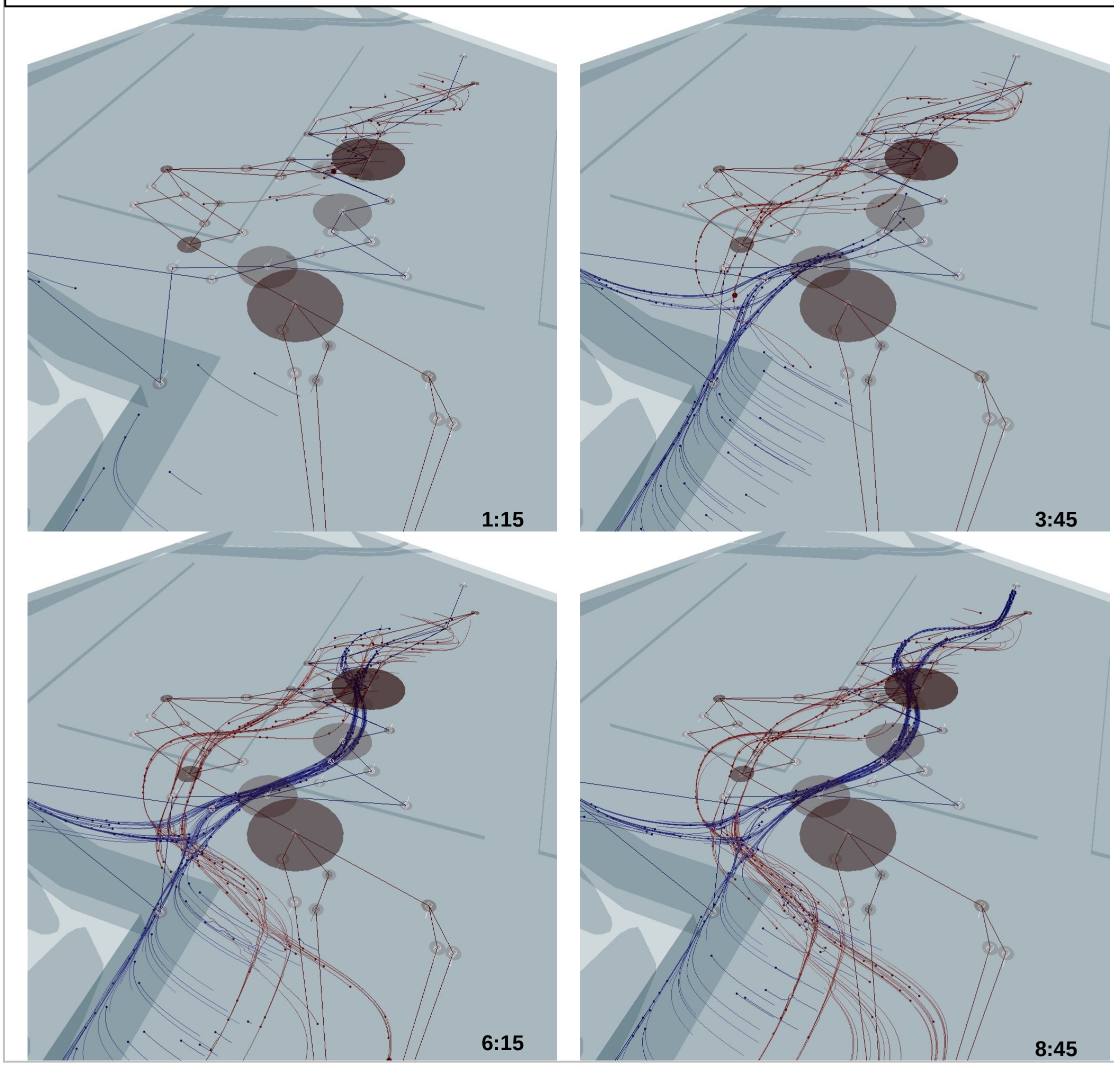
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:1.0 length/time units

Passenger movement rules

1. Sign distribution: according to destination points
2. Sign strength $C_{\text{parallel}} = 50$ force units



3. Design of path systems: 2D model: top view

- **Initial simulation**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment
2. Boundaries: 0
3. Actual last destinations: 2, Entry points: 2
4. Signs: 88
5. Total functional area in square meters: 4280m²
6. Time step interval: (150 units) 1:15 minutes virtual clock time

Passenger initial features

1. No of groups: 4
2. No of people in each group: 1st departure group 5x50, 2nd departure group 5x50,
   3rd arrival group 50x5, 4th arrival group 5x50
3. No of destinations and group distribution: 6 destinations, 1st and 2nd group 4 first destinations randomly,
   3rd and 4th groups 5th and 6th destination randomly
4. Entry points: 2
5. Time of appearance: Random

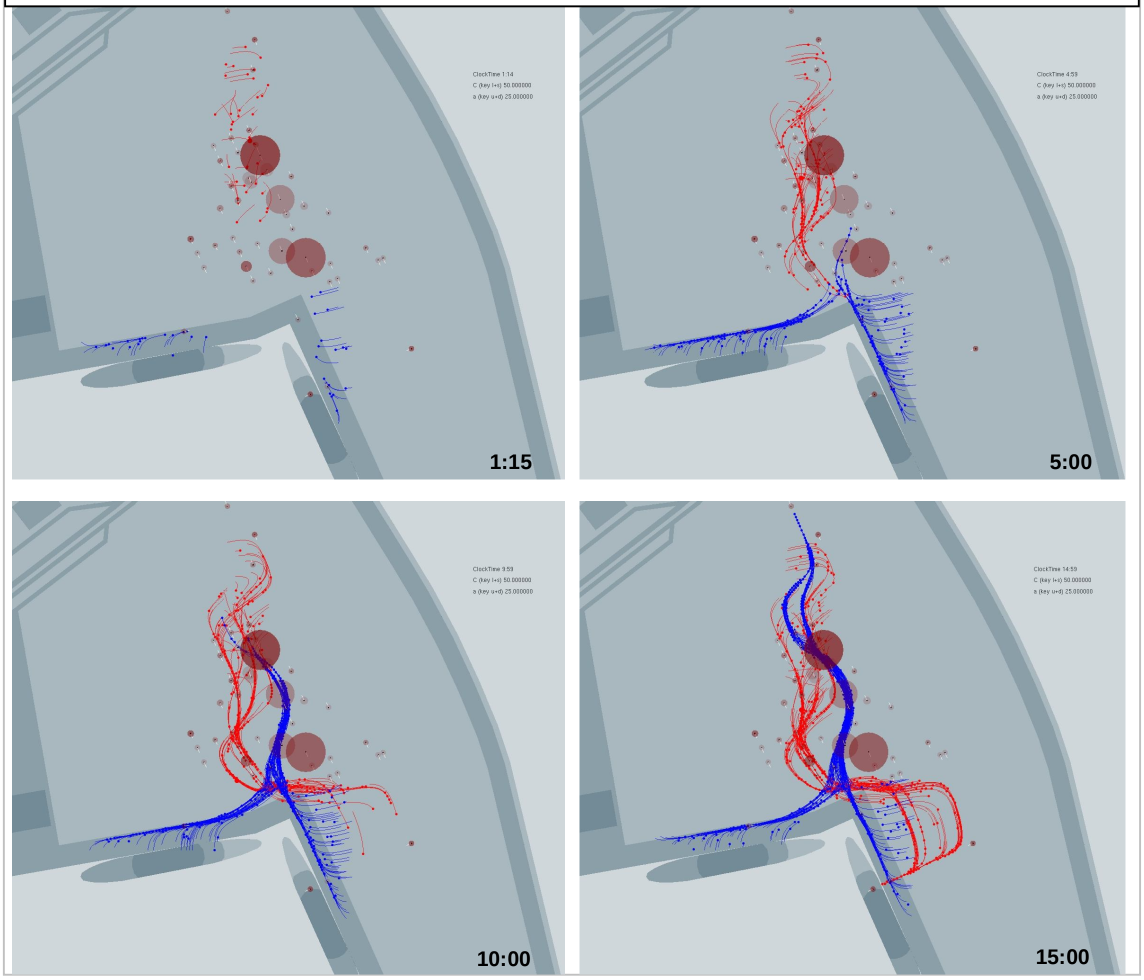
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:1.0 length/time units

Passenger movement rules

1. Sign distribution: according to destination points
2. Sign strength $C_{\text{parallel}} = 50$ force units



4. Design of path systems: 2D model: top view (zoom in)

• Initial simulation

Initial parameter
Initial geometrical configuration

1. 2D Virtual Environment
2. Boundaries: 0
3. Actual last destinations: 2, Entry points: 2
4. Signs: 88
5. Total functional area in square meters: 4280m²
6. Time step interval: (150 units) 1:15 minutes virtual clock time

Passenger initial features

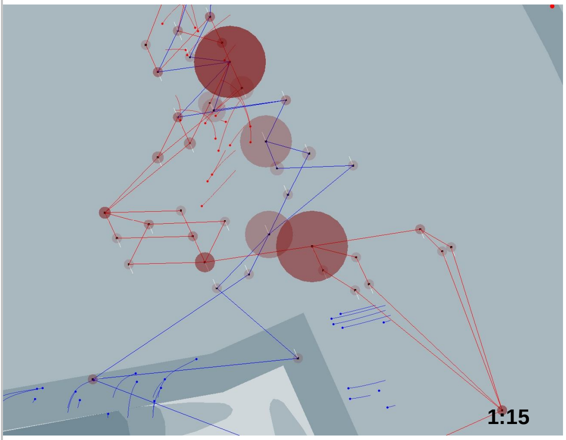
1. No of groups: 4
2. No of people in each group: 1st departure group 5x50, 2nd departure group 5x50, 3rd arrival group 50x5, 4th arrival group 5x50
3. No of destinations and group distribution: 6 destinations, 1st and 2nd group 4 first destinations randomly, 3rd and 4th groups 5th and 6th destination randomly
4. Entry points: 2
5. Time of appearance: Random

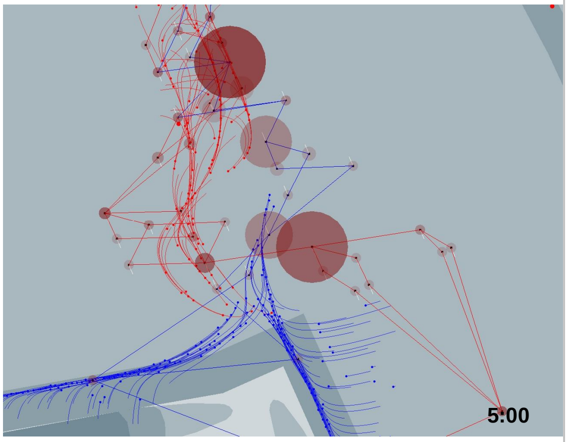
Passenger behavioural rules
Passenger interaction rules

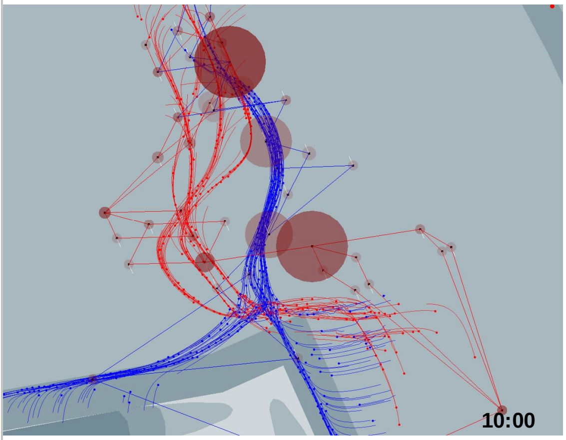
1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:1.0 length/time units

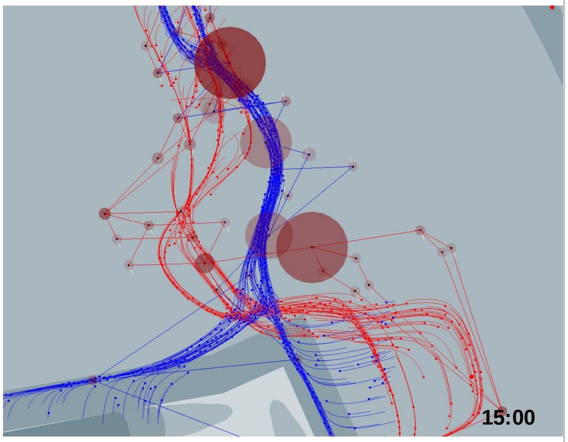
Passenger movement rules

1. Sign distribution: according to destination points
2. Sign strength $C_{\text{parallel}} = 50$ force units









5. Design of path systems: 2D model: top view (zoom-in on foyer area)

• Initial simulation

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment
2. Boundaries: 0
3. Actual last destinations: 2, Entry points: 2
4. Signs: 88
5. Total functional area in square meters: 4280m²
6. Time step interval: (150 units) 1:15 minutes virtual clock time

Passenger initial features

1. No of groups: 4
2. No of people in each group: 1st departure group 5x50, 2nd departure group 5x50,
   3rd arrival group 50x5, 4th arrival group 5x50
3. No of destinations and group distribution: 6 destinations, 1st and 2nd group 4 first destinations randomly,
   3rd and 4th groups 5th and 6th destination randomly
4. Entry points: 2
5. Time of appearance: Random

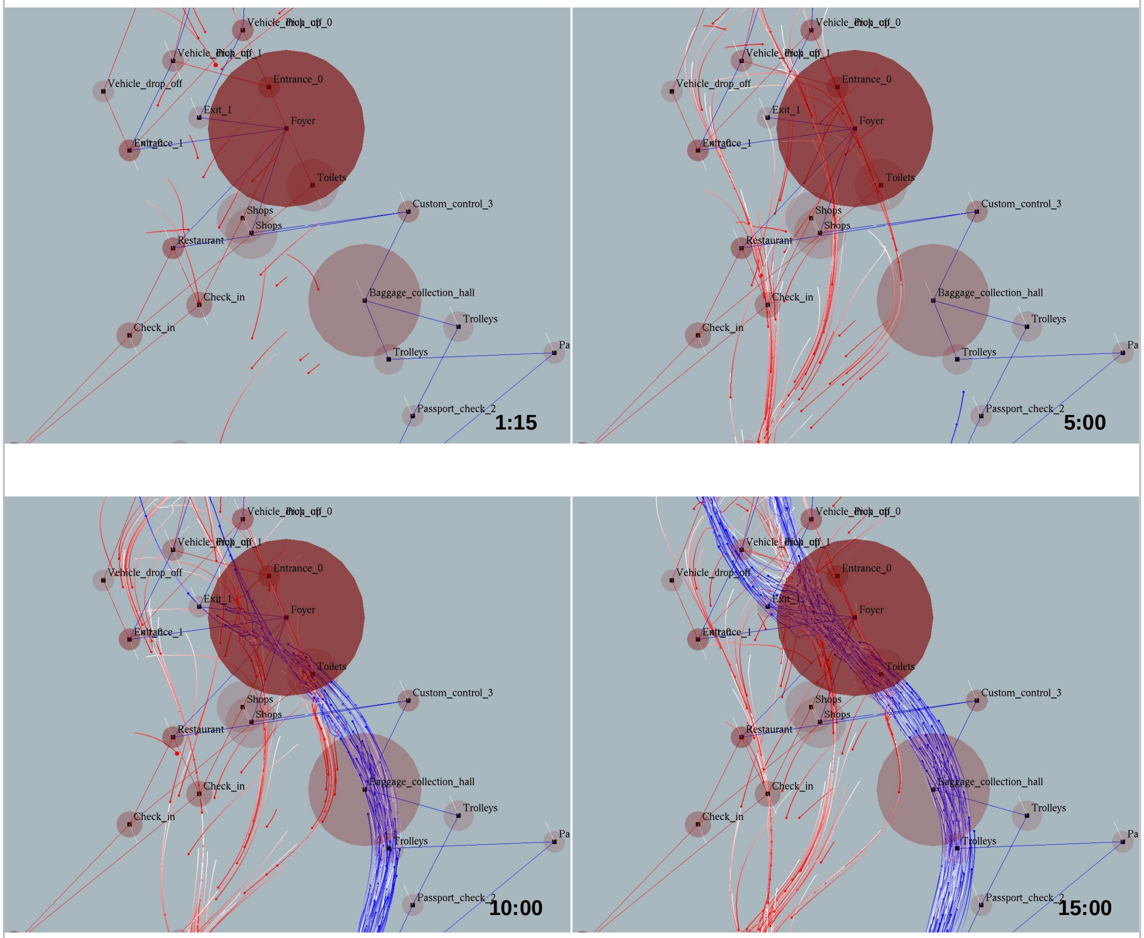
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:1.0 length/time units

Passenger movement rules

1. Sign distribution: according to destination points
2. Sign strength $C_{\text{parallel}} = 50$ force units



6. Design of path systems: 2D model (zoom-in on arrival/departures hall)

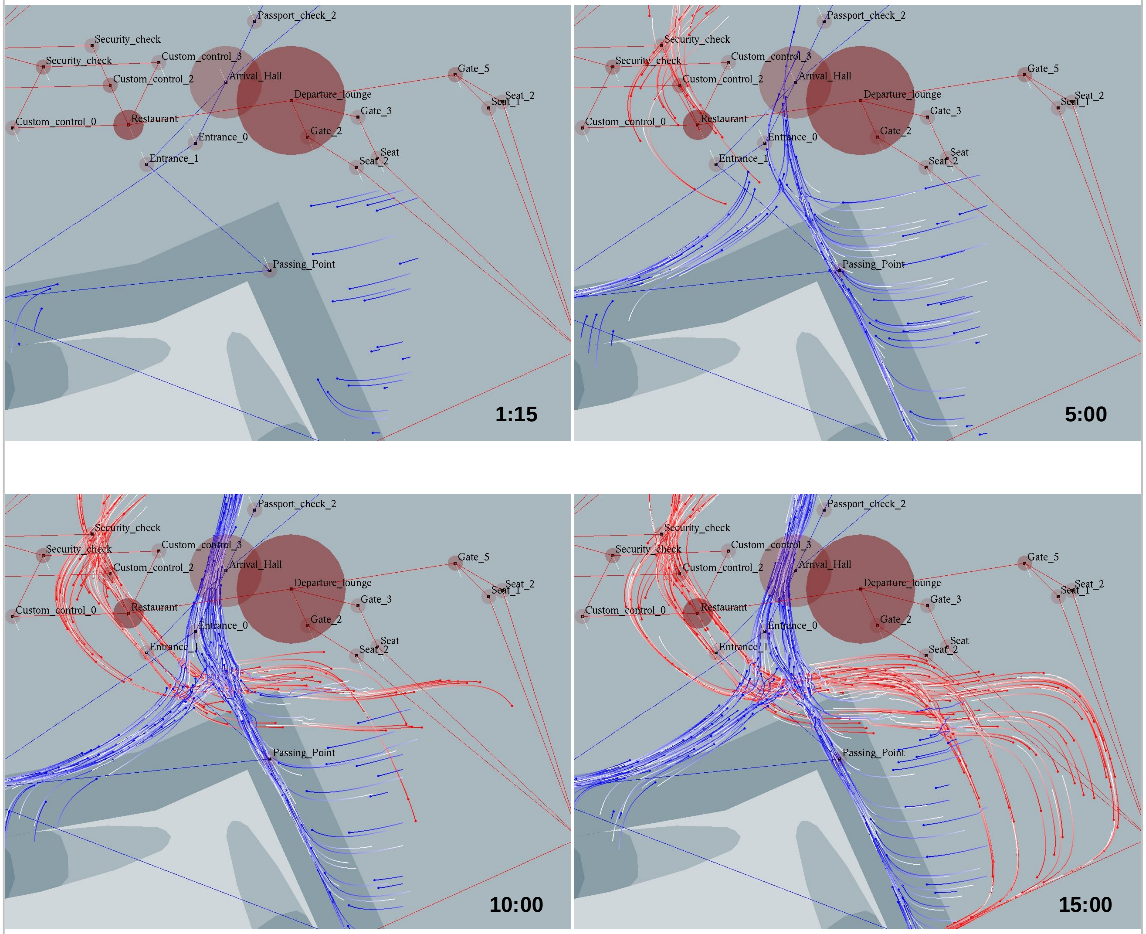
• Initial simulation

Initial parameter
Initial geometrical configuration
1. 2D Virtual Environment
2. Boundaries: 0
3. Actual last destinations: 2, Entry points: 2
4. Signs: 88
5. Total functional area in square meters: 4280m²
6. Time step interval: (150 units) 1:15 minutes virtual clock time

Passenger initial features
1. No of groups: 4
2. No of people in each group: 1st departure group 5x50, 2nd departure group 5x50, 3rd arrival group 50x5, 4th arrival group 5x50
3. No of destinations and group distribution: 6 destinations, 1st and 2nd group 4 first destinations randomly, 3rd and 4th groups 5th and 6th destination randomly
4. Entry points: 2
5. Time of appearance: Random

Passenger behavioural rules
Passenger interaction rules
1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:1.0 length/time units

Passenger movement rules
1. Sign distribution: according to destination points
2. Sign strength $C_{\text{parallel}} = 50$ force units



7. Design of path systems: 3D model: isometric view

- **1st suggested design/ simulation**

Initial parameter

Initial geometrical configuration

1. 3D Virtual Environment
2. Boundaries: 0
3. Actual last destinations: 2, Entry points: 2
4. Signs: 88
5. Total functional area in square meters: 4280m²
6. Time step interval: (150 units) 1:15 minutes virtual clock time

Passenger initial features

1. No of groups: 4
2. No of people in each group: 1st departure group 5x50, 2nd departure group 5x50,
   3rd arrival group 50x5, 4th arrival group 5x50
3. No of destinations and group distribution: 6 destinations, 1st and 2nd group 4 first destinations randomly,
   3rd and 4th groups 5th and 6th destination randomly
4. Entry points: 2
5. Time of appearance: Random

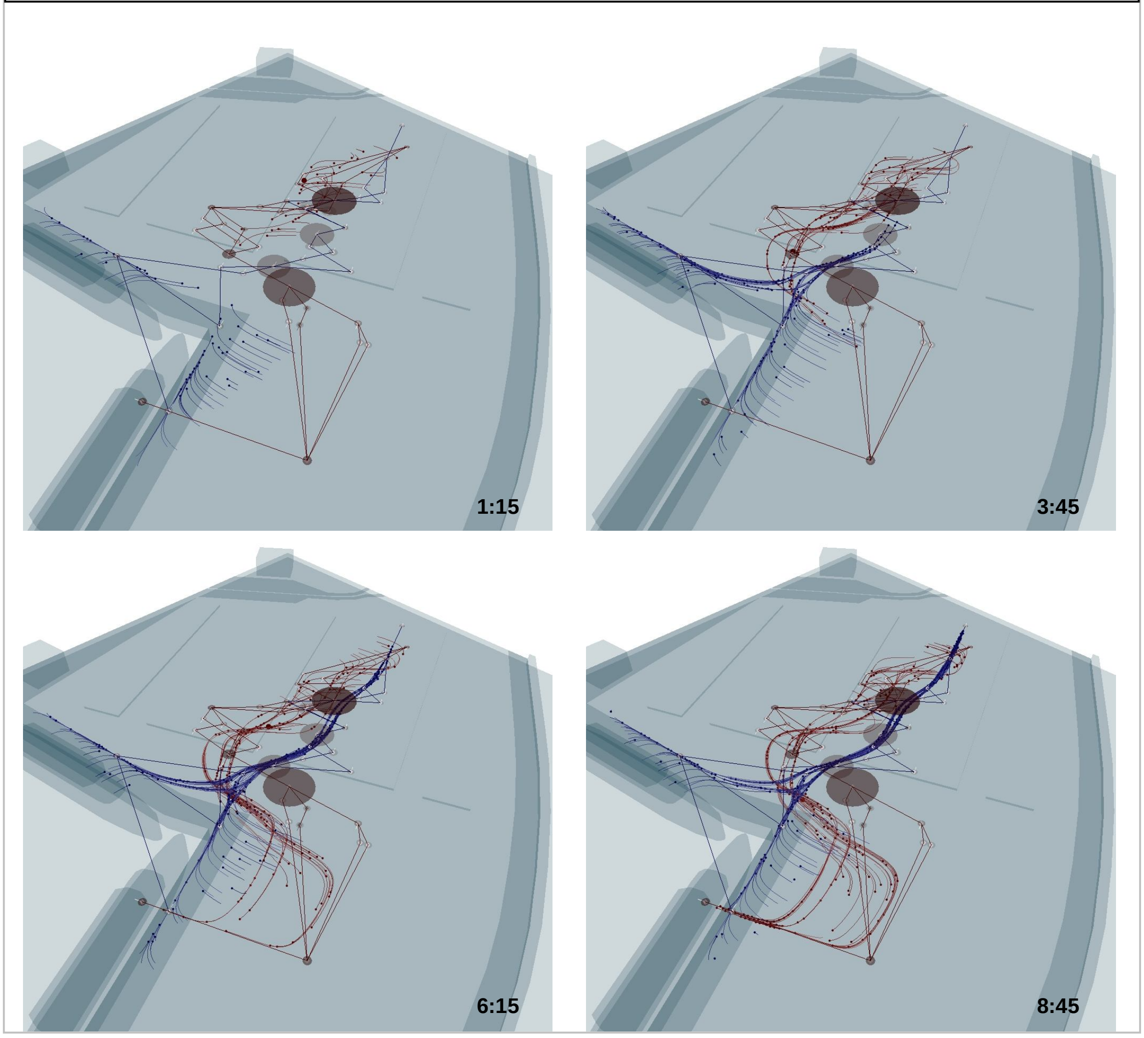
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:1.0 length/time units

Passenger movement rules

1. Sign distribution: according to destination points
2. Sign strength $C_{\text{parallel}} = 50$ force units



8. Design of path systems: 3D model: isometric view (zoom-in)

• 1st suggested design/ simulation

Initial parameter

Initial geometrical configuration

1. 3D Virtual Environment
2. Boundaries: 0
3. Actual last destinations: 2, Entry points: 2
4. Signs: 88
5. Total functional area in square meters: 4280m²
6. Time step interval: (150 units) 1:15 minutes virtual clock time

Passenger initial features

1. No of groups: 4
2. No of people in each group: 1st departure group 5x50, 2nd departure group 5x50, 3rd arrival group 50x5, 4th arrival group 5x50
3. No of destinations and group distribution: 6 destinations, 1st and 2nd group 4 first destinations randomly, 3rd and 4th groups 5th and 6th destination randomly
4. Entry points: 2
5. Time of appearance: Random

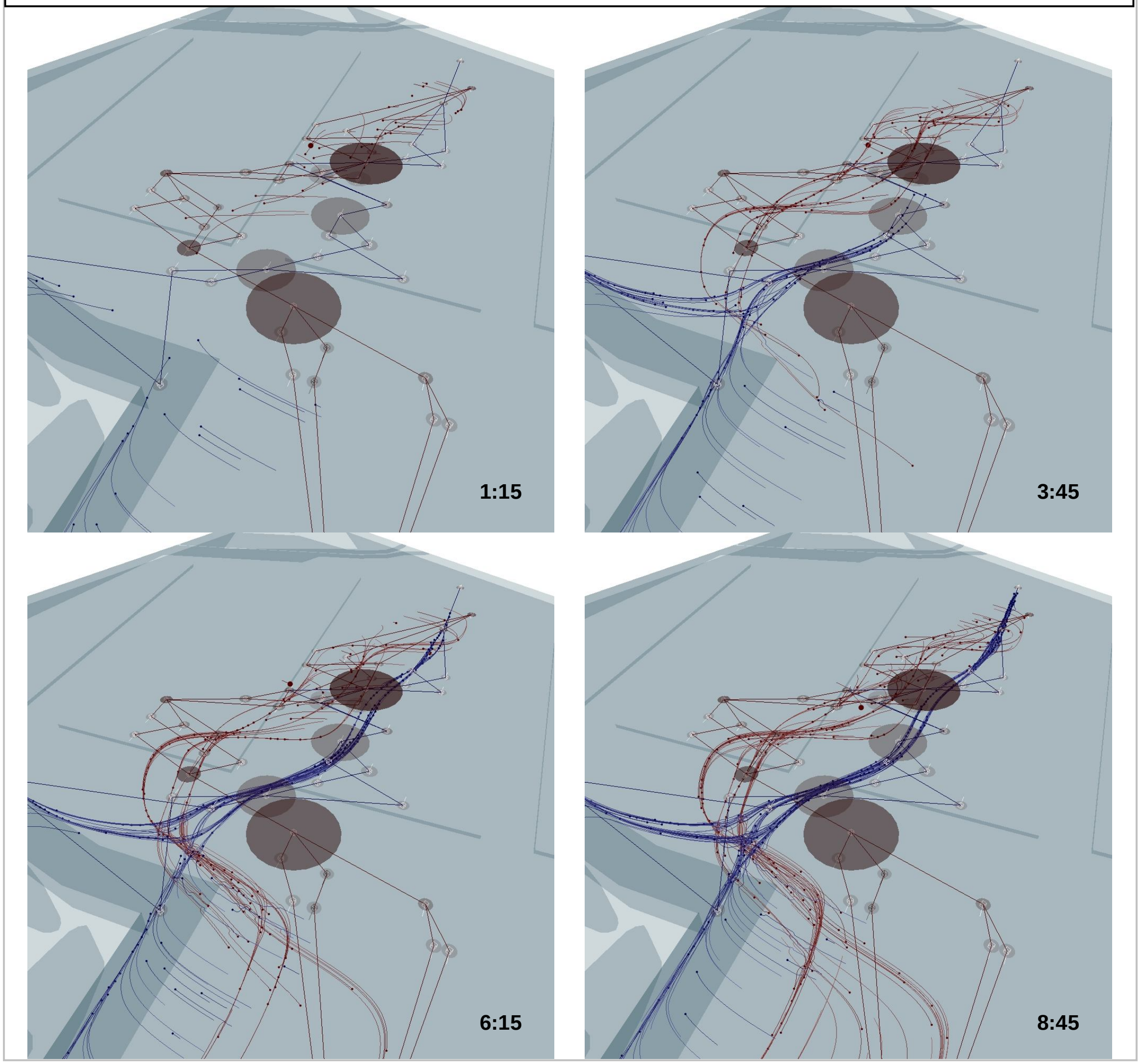
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:1.0 length/time units

Passenger movement rules

1. Sign distribution: according to destination points
2. Sign strength $C_{\text{parallel}} = 50$ force units



9. Design of path systems: 2D model: top view

• 1st suggested design/ simulation

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment
2. Boundaries: 0
3. Actual last destinations: 2, Entry points: 2
4. Signs: 88
5. Total functional area in square meters: 4280m²
6. Time step interval: (150 units) 1:15 minutes virtual clock time

Passenger initial features

1. No of groups: 4
2. No of people in each group: 1st departure group 5x50, 2nd departure group 5x50, 3rd arrival group 50x5, 4th arrival group 5x50
3. No of destinations and group distribution: 6 destinations, 1st and 2nd group 4 first destinations randomly, 3rd and 4th groups 5th and 6th destination randomly
4. Entry points: 2
5. Time of appearance: Random

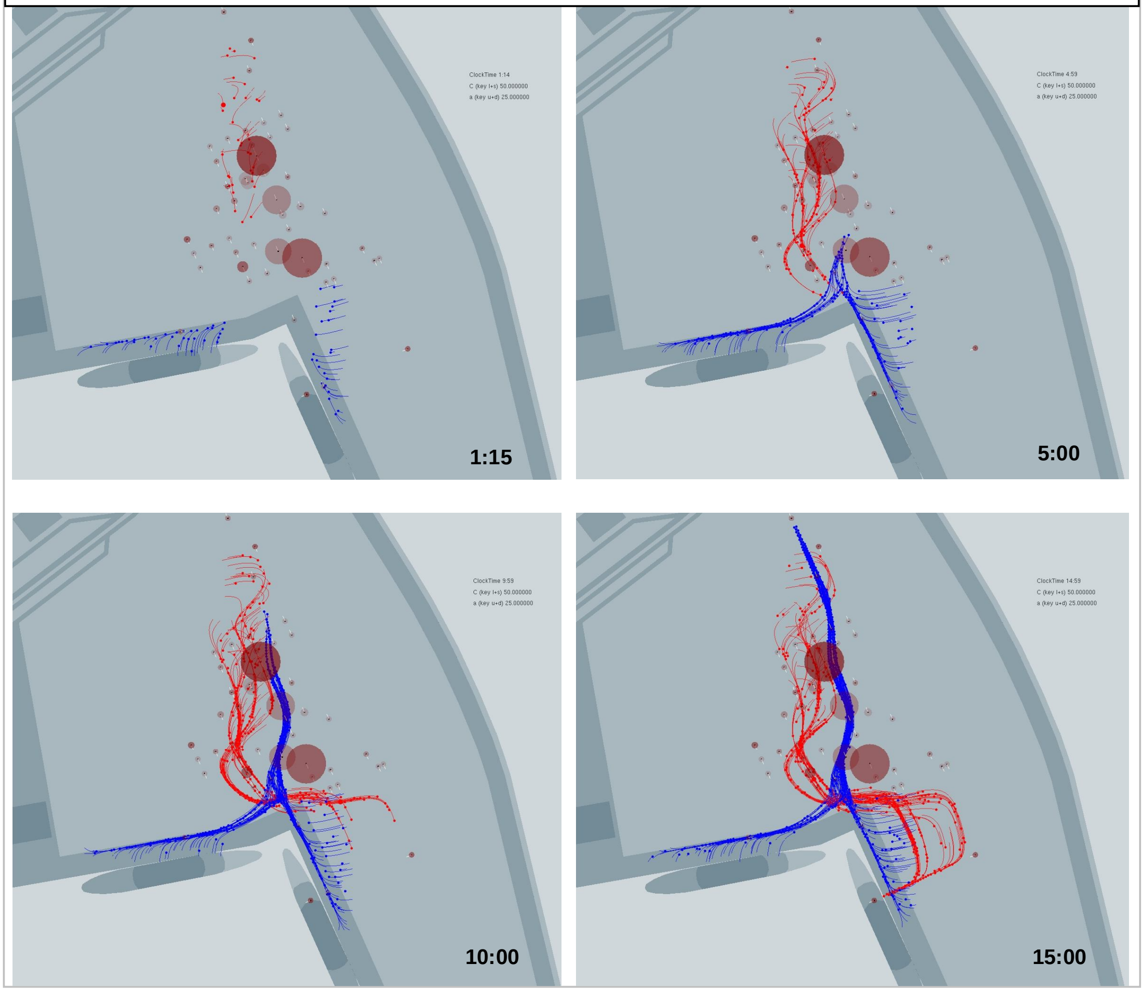
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:1.0 length/time units

Passenger movement rules

1. Sign distribution: according to destination points
2. Sign strength $C_{\text{parallel}} = 50$ force units



10. Design of path systems: 2D model: top view (zoom in)

- **1st suggested design/ simulation**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment
2. Boundaries: 0
3. Actual last destinations: 2, Entry points: 2
4. Signs: 88
5. Total functional area in square meters: 4280m²
6. Time step interval: (150 units) 1:15 minutes virtual clock time

Passenger initial features

1. No of groups: 4
2. No of people in each group: 1st departure group 5x75, 2nd departure group 5x75, 3rd arrival group 50x5, 4th arrival group 5x50l
3. No of destinations and group distribution: 6 destinations, 1st and 2nd group 4 first destinations randomly, 3rd and 4th groups 5th and 6th destination randomly
4. Entry points: 2
5. Time of appearance: Random

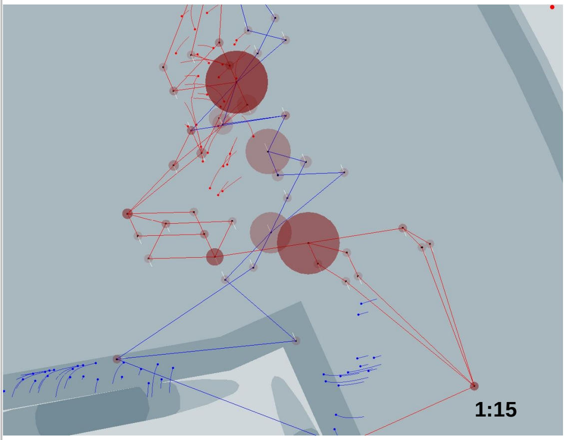
Passenger behavioural rules

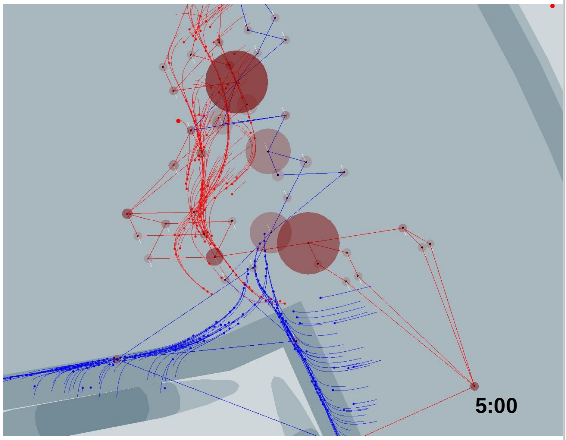
Passenger interaction rules

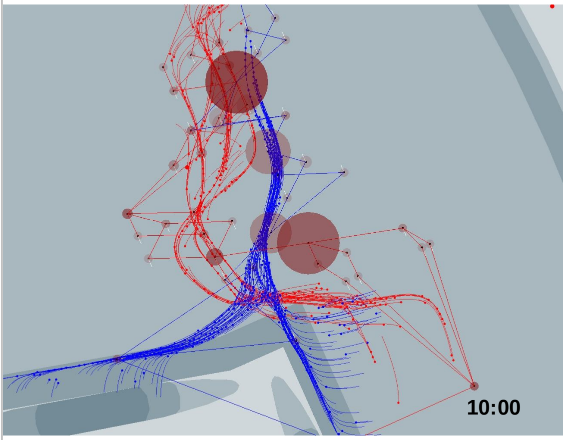
1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:1.0 length/time units

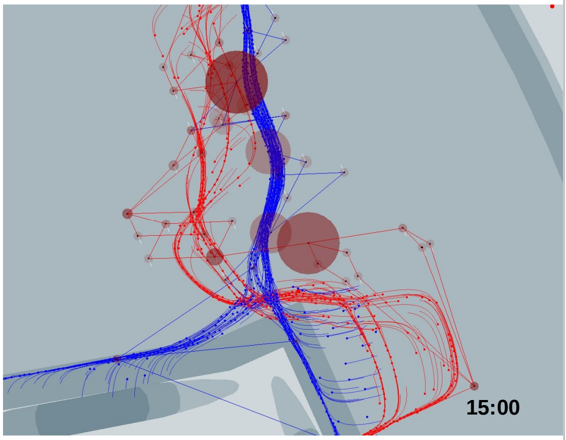
Passenger movement rules

1. Sign distribution: according to destination points
2. Sign strength $C_{\text{parallel}} = 50$ force units









11. Design of path systems: 2D model: top view (zoom-in on foyer area)

- **1st suggested design/ simulation**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment
2. Boundaries: 0
3. Actual last destinations: 2, Entry points: 2
4. Signs: 88
5. Total functional area in square meters: 4280m²
6. Time step interval: (150 units) 1:15 minutes virtual clock time

Passenger initial features

1. No of groups: 4
2. No of people in each group: 1st departure group 5x75, 2nd departure group 5x75, 3rd arrival group 50x5, 4th arrival group 5x50l
3. No of destinations and group distribution: 6 destinations, 1st and 2nd group 4 first destinations randomly, 3rd and 4th groups 5th and 6th destination randomly
4. Entry points: 2
5. Time of appearance: Random

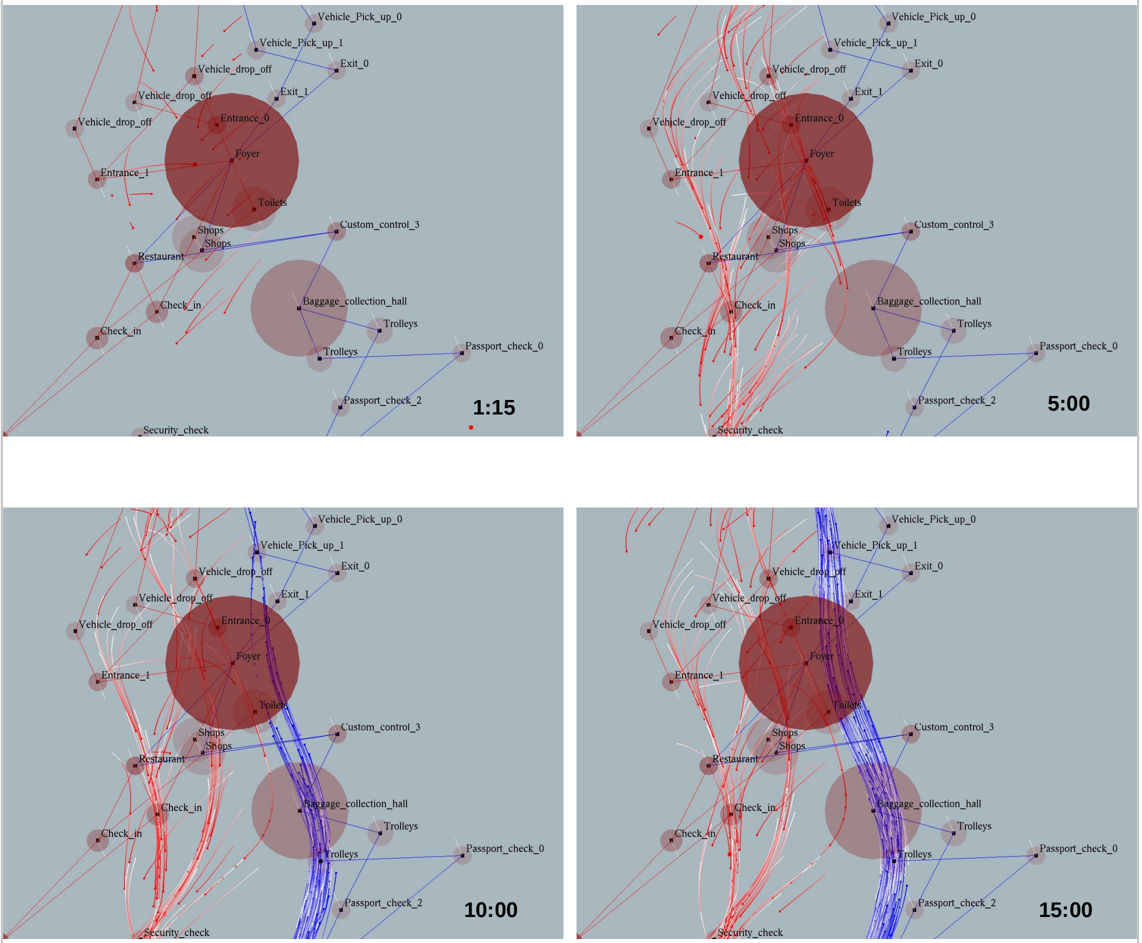
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:1.0 length/time units

Passenger movement rules

1. Sign distribution: according to destination points
2. Sign strength $C_{\text{parallel}} = 50$ force units



Appendix E

Appendix E contains the studies that were examined at the meso-scale. At this scale the spatial configuration of functional areas and the design of circulation diagrams in foyer area (localized model) were studied. The parameters are organized as follow:

The initial geometrical configuration consists of parameters that specify the virtual environment in 2D or 3D. Boundaries are not included, and last destinations' number is set to two with the same number for entry points. The number of signs is set to 24 destinations x 50 x 50. Also, the total functional area in square meters is 1700m². The time step interval for each captured image is set to 1:40 minutes (virtual clock time).

The passengers' initial information includes: the number of passenger groups that is set to 2, the number of passengers in each group, the number of destinations that is set to 24, the group distribution, and finally the time of appearance that is random.

The passenger behavioural rules include the repulsive effect, the value of variable $a_{\text{interperson repel}}$ that is set to 25 length units and the movement speed that is set to 2.5 length/time units.

Finally, the passengers' movement rules include the signs distribution and the value of sign strength C_{parallel} that is defined as 15 force units.

1. Functional areas and circulation design: 2D model: top view of foyer area

- **Initial simulation**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 10Units
2. Boundaries: 0
3. Last destinations: 2, Entry points: 2
4. Signs: 24 destinations x50x50 signs
5. Total functional area in square meters: 1700m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 100x1, 2nd arrival group 100x1,
3. No of destinations and group distribution: 24 (0-23) destinations,
   1st group 0-12 destination: compulsory and optional 2nd group 13-23 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

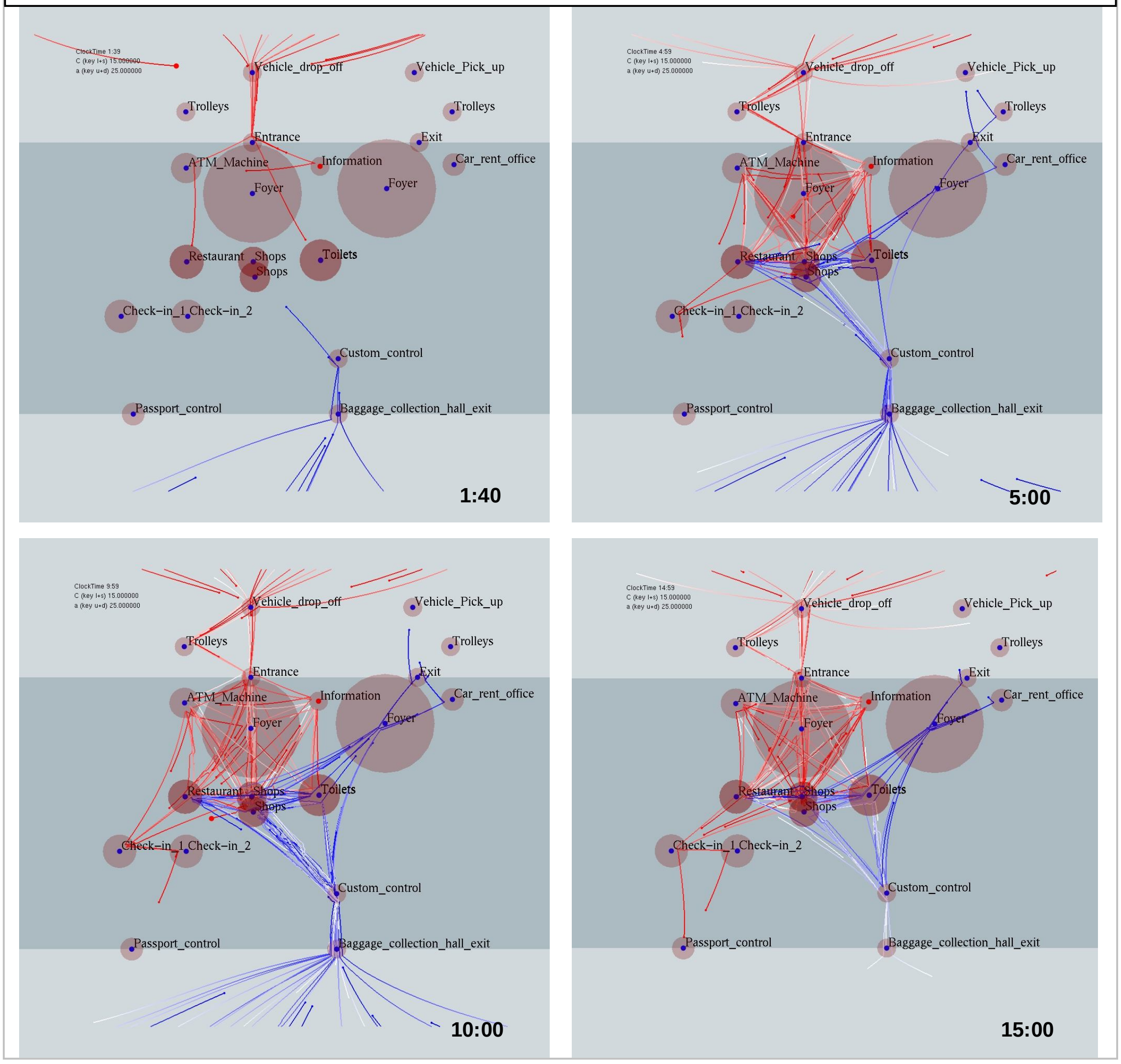
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:2.5 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 15$ force units



2. Functional areas and circulation design: 2D model: top view of foyer area B

- **Initial simulation**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 10Units
2. Boundaries: 0
3. Last destinations: 2, Entry points: 2
4. Signs: 24 destinations x50x50 signs
5. Total functional area in square meters: 1700m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 100x1, 2nd arrival group 100x1,
3. No of destinations and group distribution: 24 (0-23) destinations, 1st group 0-12 destination: compulsory and optional 2nd group 13-23 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

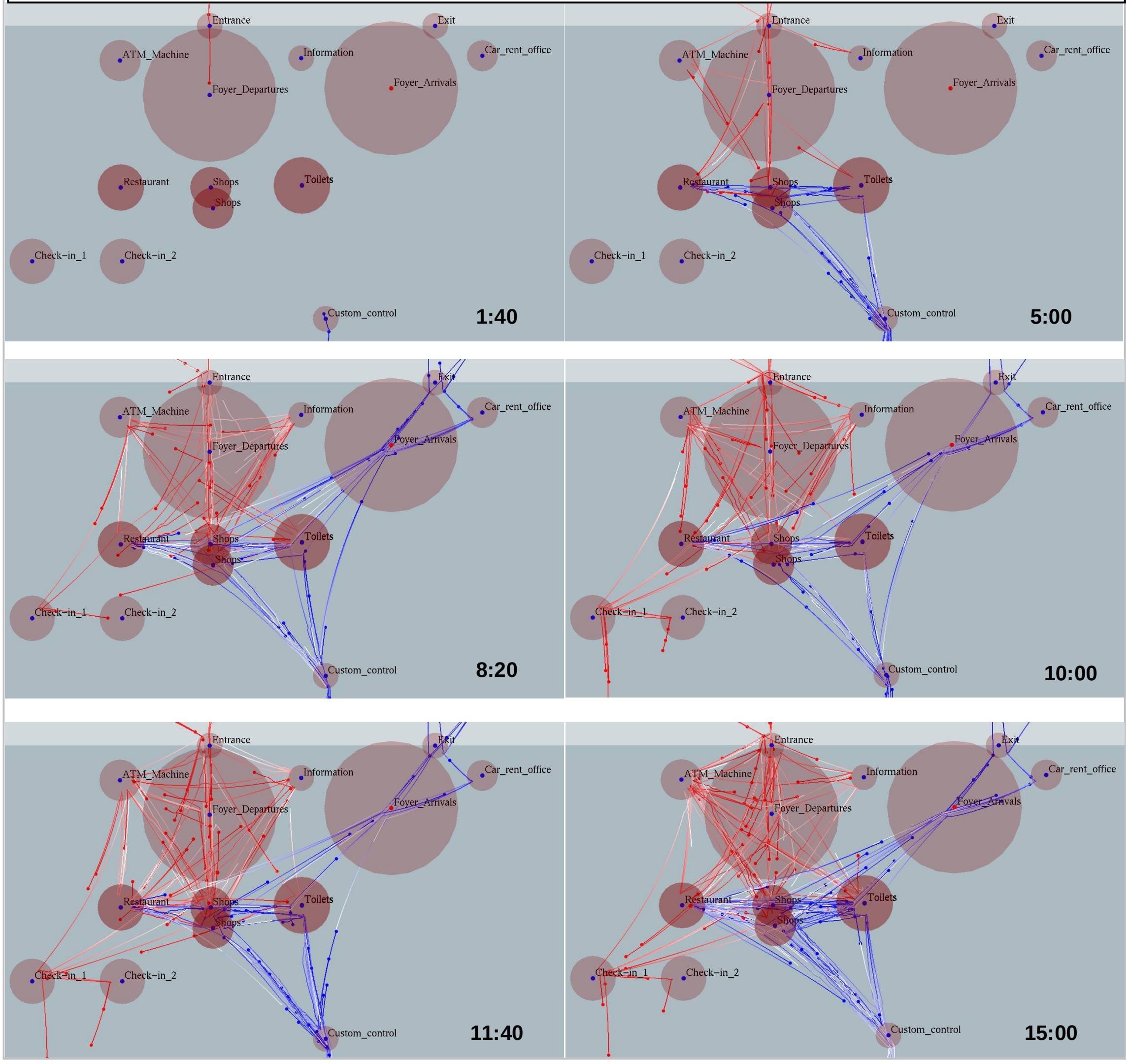
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:2.5 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 15$ force units



3. Functional areas and circulation design: 3D model: isometric view

- **Initial simulation**

Initial parameter

Initial geometrical configuration

1. 3D Virtual Environment, scale: 1m: 10Units
2. Boundaries: 0
3. Last destinations: 2, Entry points: 2
4. Signs: 24 destinations x50x50 signs
5. Total functional area in square meters: 1700m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 100x1, 2nd arrival group 100x1,
3. No of destinations and group distribution: 24 (0-23) destinations,
   1st group 0-12 destination: compulsory and optional 2nd group 13-23 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

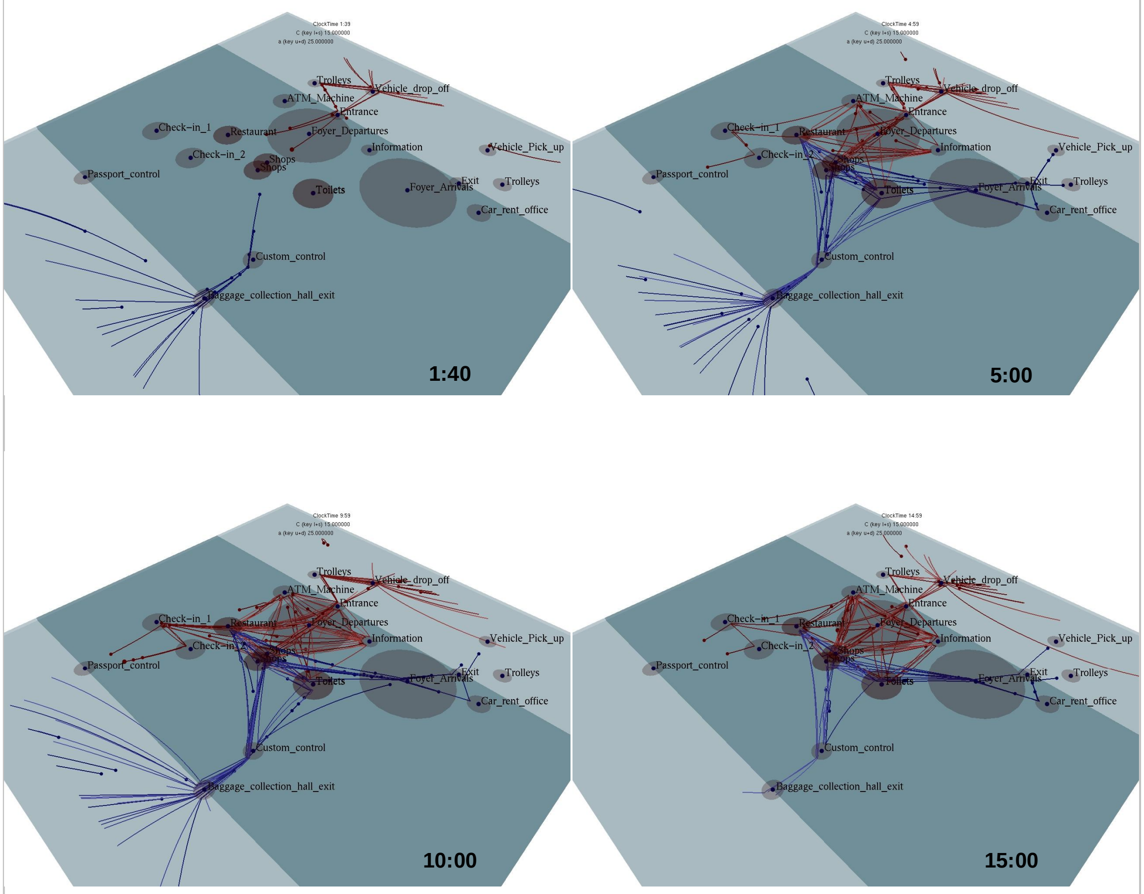
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:2.5 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 15$ force units



4. Functional areas and circulation design: 3D model: isometric view (zoom-in)

- **Initial simulation**

Initial parameter

Initial geometrical configuration

1. 3D Virtual Environment, scale: 1m: 10Units
2. Boundaries: 0
3. Last destinations: 2, Entry points: 2
4. Signs: 24 destinations x50x50 signs
5. Total functional area in square meters: 1700m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 100x1, 2nd arrival group 100x1,
3. No of destinations and group distribution: 24 (0-23) destinations,
   1st group 0-12 destination: compulsory and optional 2nd group 13-23 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

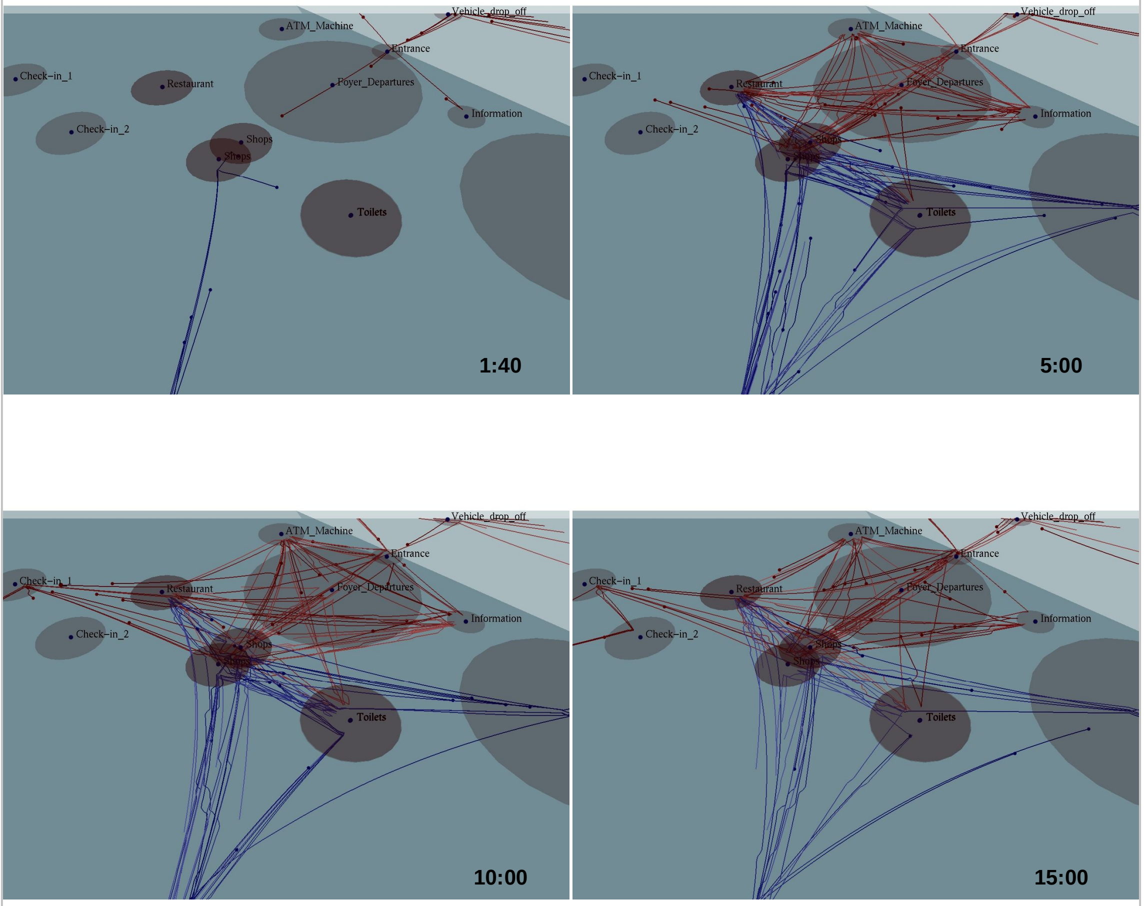
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:2.5 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 15$ force units



5. Functional areas and circulation design: 2D model: top view of foyer area

- **1st suggested design/ simulation**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 10Units
2. Boundaries: 0
3. Last destinations: 2, Entry points: 2
4. Signs: 24 destinations x50x50 signs
5. Total functional area in square meters: 1700m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 100x1, 2nd arrival group 100x1
3. No of destinations and group distribution: 24 (0-23) destinations,
   1st group 0-12 destination: compulsory and optional 2nd group 13-23 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

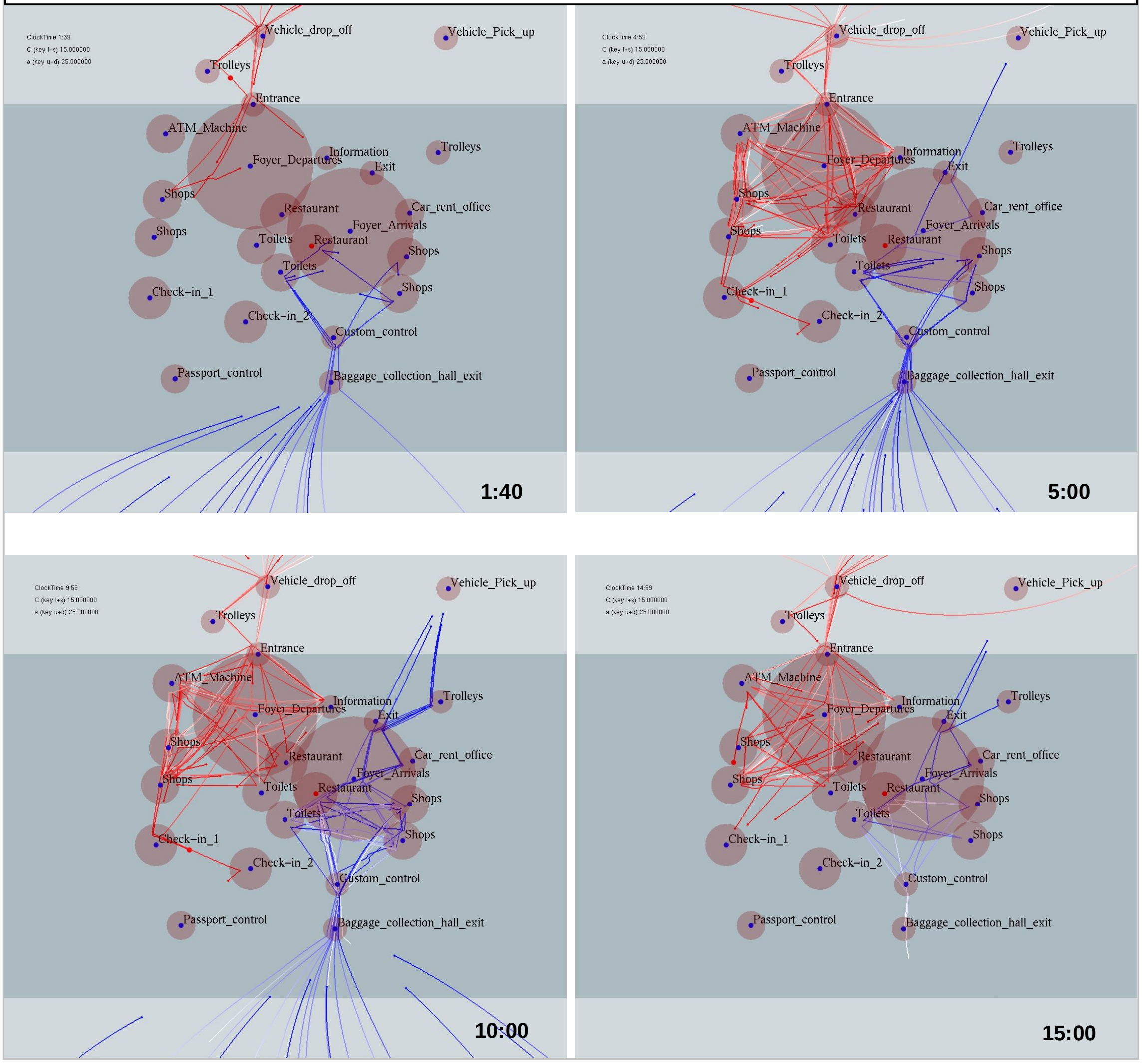
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:2.5 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 15$ force units



6. Functional areas and circulation design: 2D model: top view of foyer area B

- **1st suggested design/ simulation**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 10Units
2. Boundaries: 0
3. Last destinations: 2, Entry points: 2
4. Signs: 24 destinations x50x50 signs
5. Total functional area in square meters: 1700m^2
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 100x1, 2nd arrival group 100x1
3. No of destinations and group distribution: 24 (0-23) destinations,
   1st group 0-12 destination: compulsory and optional 2nd group 13-23 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

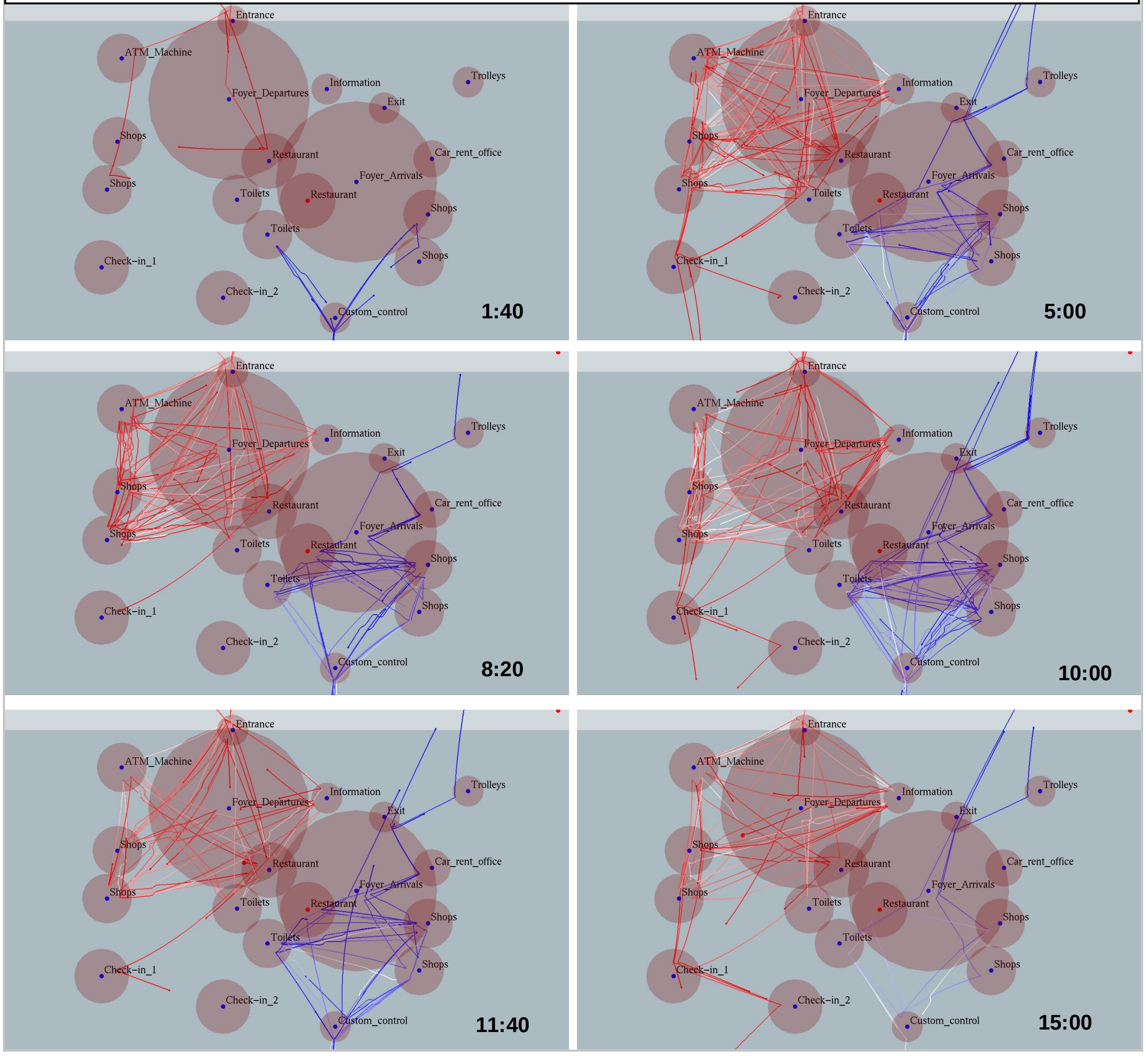
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:2.5 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 15$ force units



7. Functional areas and circulation design: 2D model: top view of foyer area B

- **2^{nd} suggested design/ simulation**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 10Units
2. Boundaries: 0
3. Last destinations: 2, Entry points: 2
4. Signs: 24 destinations x50x50 signs
5. Total functional area in square meters: 1700m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1^{st} departure group 100x1, 2^{nd} arrival group 100x1
3. No of destinations and group distribution: 24 (0-23) destinations, 1^{st} group 0-12 destination: compulsory and optional 2^{nd} group 13-23 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

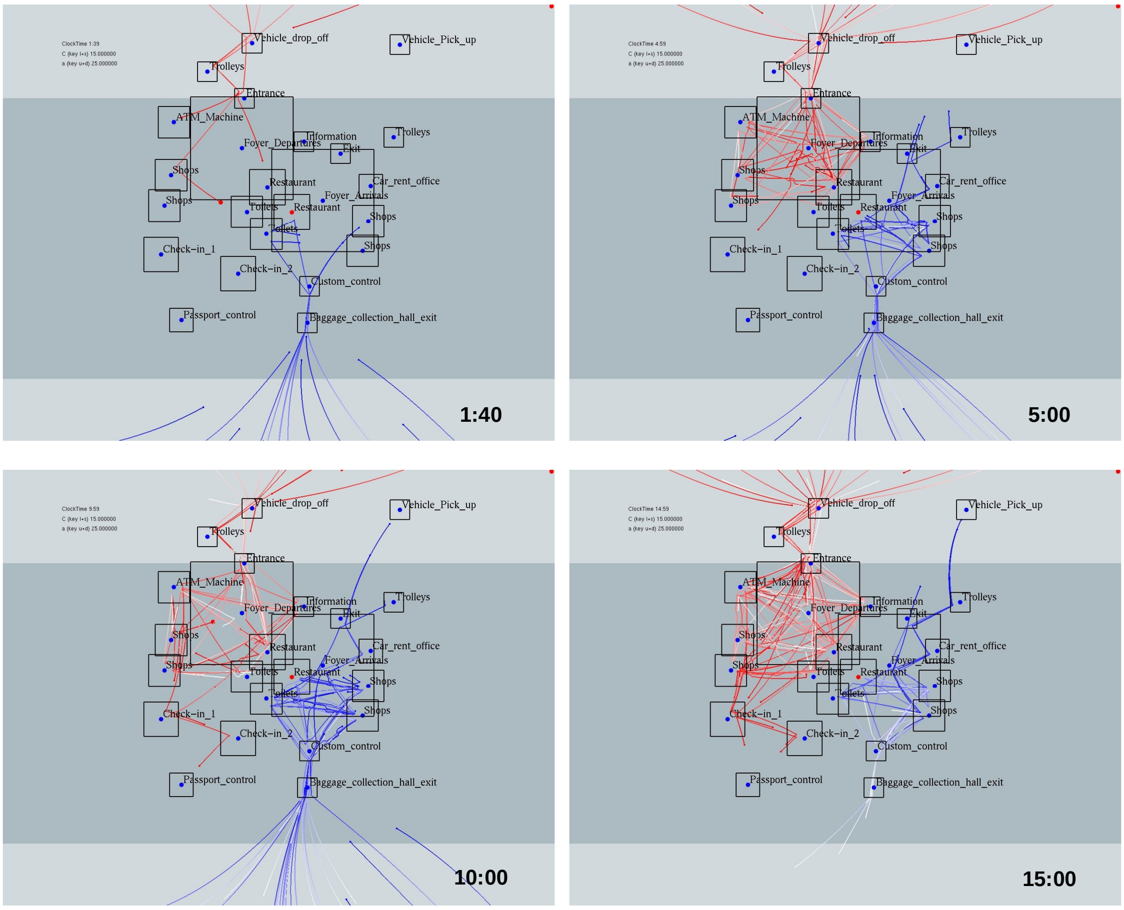
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:2.5 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 15$ force units



8. Functional areas and circulation design: 2D model: top view of foyer area B

- **3rd suggested design/ simulation**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 10Units
2. Boundaries: 0
3. Last destinations: 2, Entry points: 2
4. Signs: 24 destinations x50x50 signs
5. Total functional area in square meters: 1700m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 100x1, 2nd arrival group 100x1
3. No of destinations and group distribution: 24 (0-23) destinations,
   1st group 0-12 destination: compulsory and optional 2nd group 13-23 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

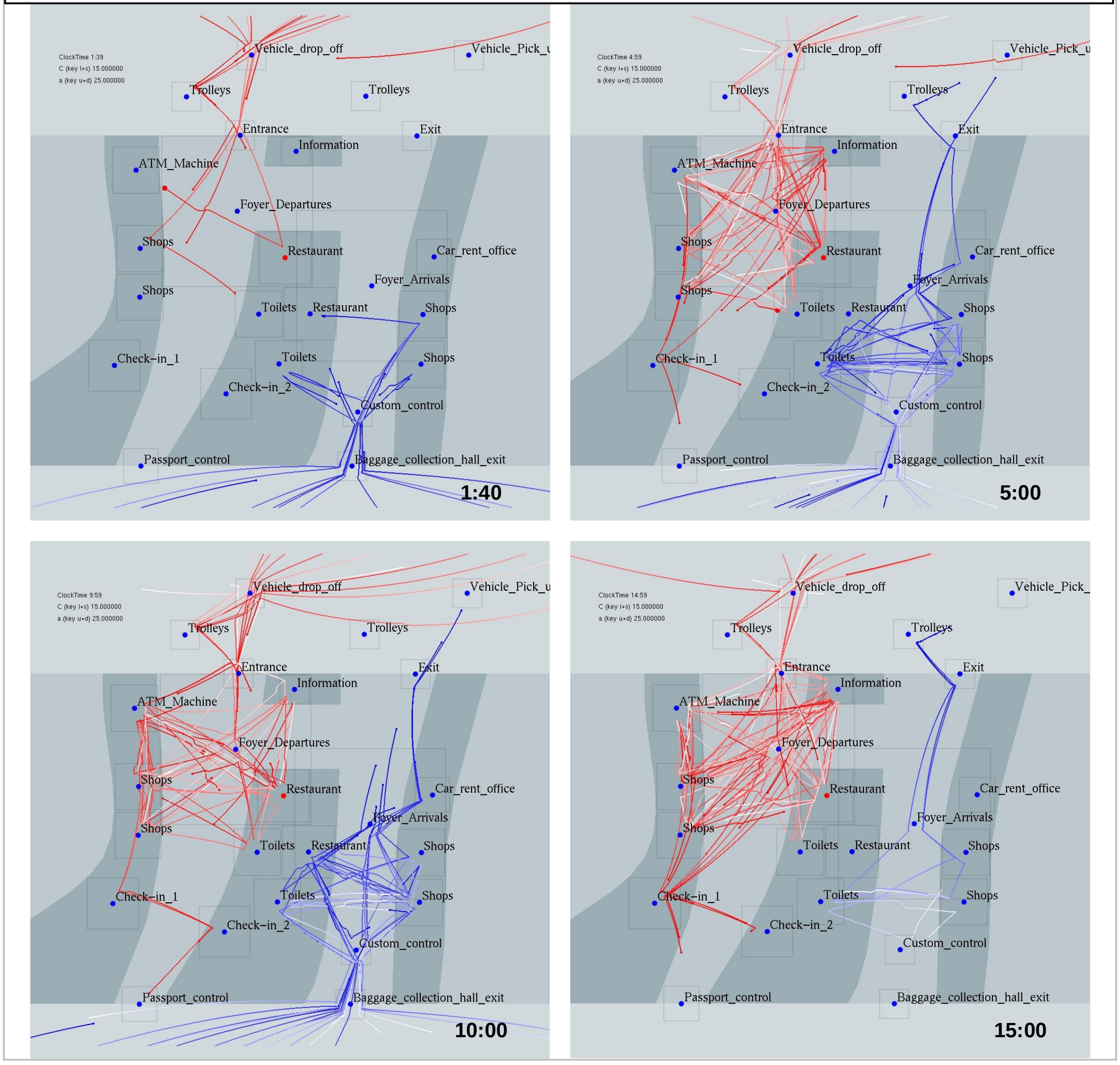
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:2.5 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 15$ force units



9. Functional areas and circulation design: 2D model: top view of foyer area

- **3rd suggested design/ simulation**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 10Units
2. Boundaries: 0
3. Last destinations: 2, Entry points: 2
4. Signs: 24 destinations x50x50 signs
5. Total functional area in square meters: 1700m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 100x1, 2nd arrival group 100x1
3. No of destinations and group distribution: 24 (0-23) destinations,
   1st group 0-12 destination: compulsory and optional 2nd group 13-23 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

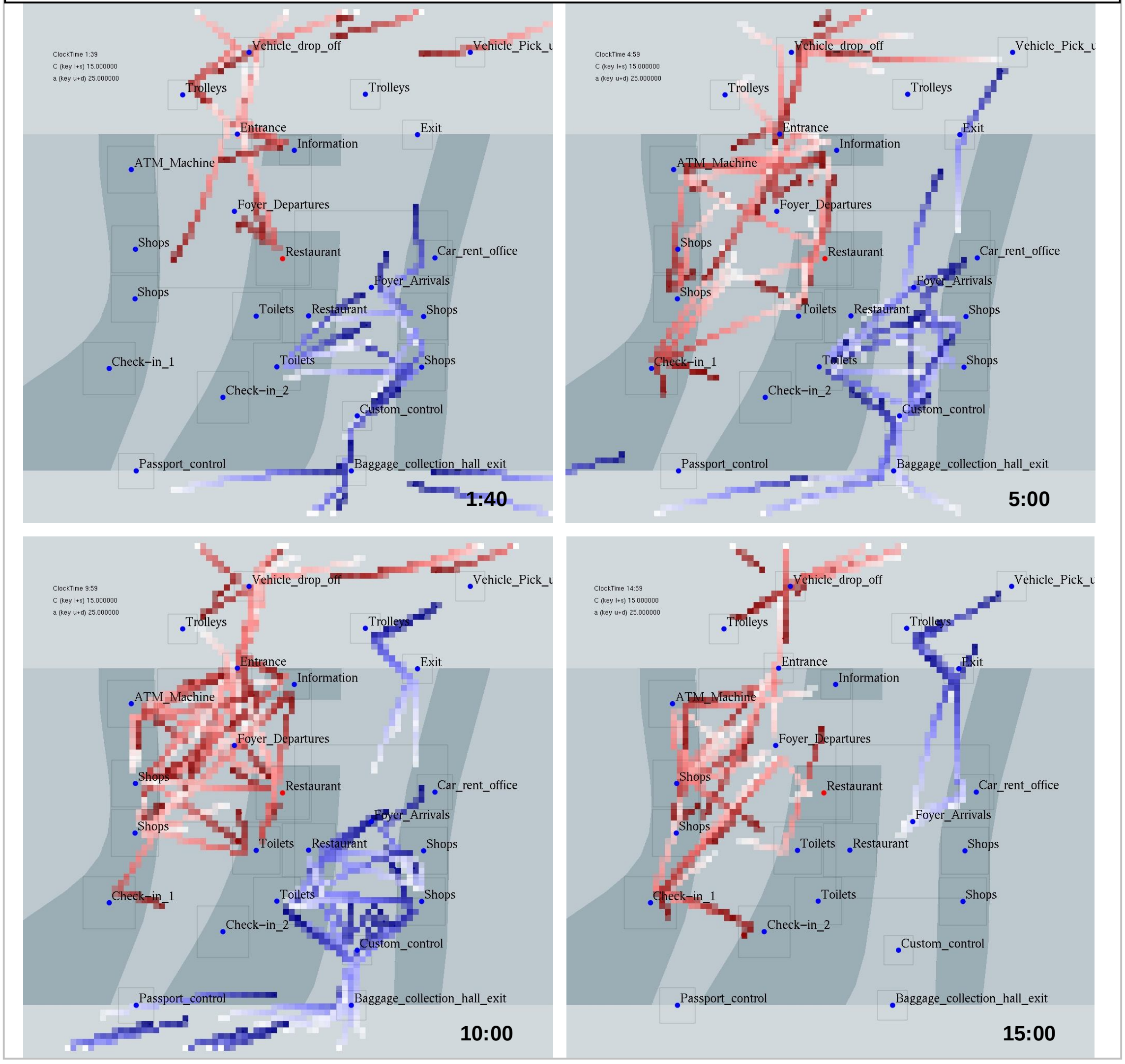
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:2.5 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 15$ force units



10. Functional areas and circulation design: 2D model: top view of foyer area

- **3rd suggested design/ simulation**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 10Units
2. Boundaries: 0
3. Last destinations: 2, Entry points: 2
4. Signs: 24 destinations x50x50 signs
5. Total functional area in square meters: 1700m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 100x1, 2nd arrival group 100x1
3. No of destinations and group distribution: 24 (0-23) destinations, 1st group 0-12 destination: compulsory and optional 2nd group 13-23 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

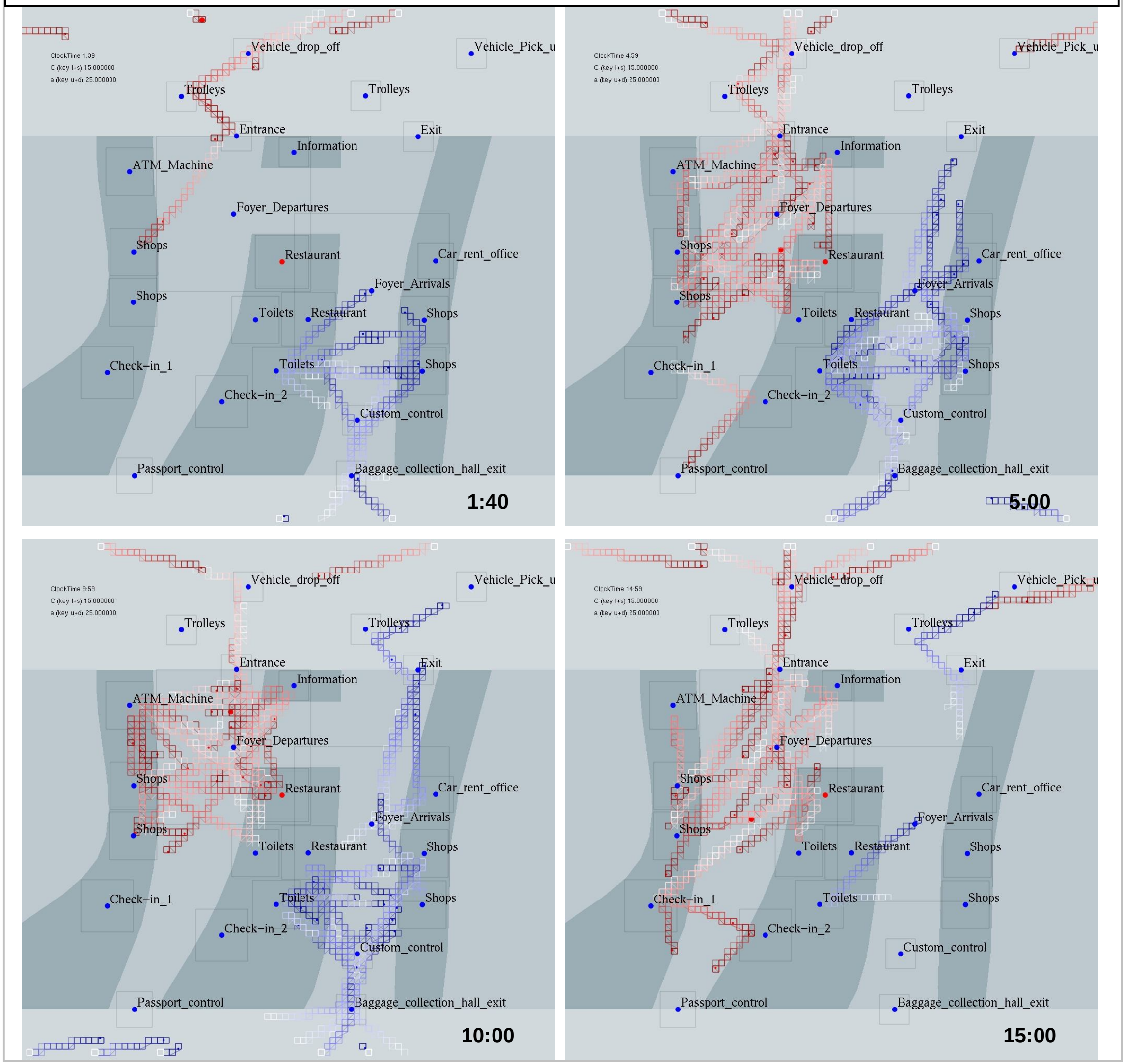
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:2.5 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 15$ force units



11. Functional areas and circulation design: 3D model: isometric view of foyer area

- **3rd suggested design/ simulation**

Initial parameter

Initial geometrical configuration

1. 3D Virtual Environment, scale: 1m: 10Units
2. Boundaries: 0 0x50 signs
3. Last destinations: 2, Entry points: 2
4. Signs: 24 destinations x50x50 signs
5. Total functional area in square meters: 1700m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 100x1, 2nd arrival group 100x1
3. No of destinations and group distribution: 24 (0-23) destinations,
   1st group 0-12 destination: compulsory and optional 2nd group 13-23 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

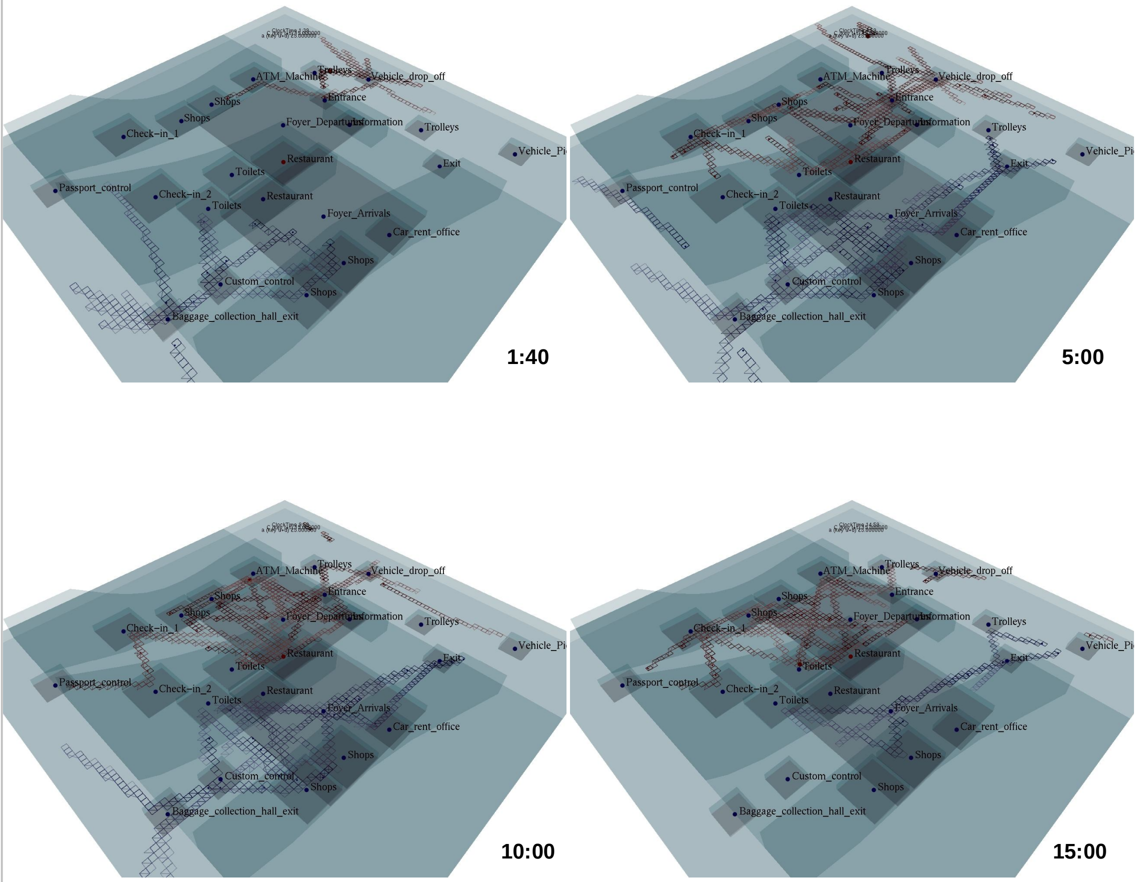
Passenger behavioural rules

Passenger interaction rules

1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:2.5 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 15$ force units



1:40

5:00

10:00

15:00

12. Functional areas and circulation design: 3D model: isometric view of foyer area

- **3^{rd} suggested design/ simulation**

Initial parameter

Initial geometrical configuration

1. 3D Virtual Environment, scale: 1m: 10Units
2. Boundaries: 0
3. Last destinations: 2, Entry points: 2
4. Signs: 24 destinations x50x50 signs
5. Total functional area in square meters: 1700m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1^{st} departure group 100x1, 2^{nd} arrival group 100x1
3. No of destinations and group distribution: 24 (0-23) destinations, 1^{st} group 0-12 destination: compulsory and optional 2^{nd} group 13-23 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

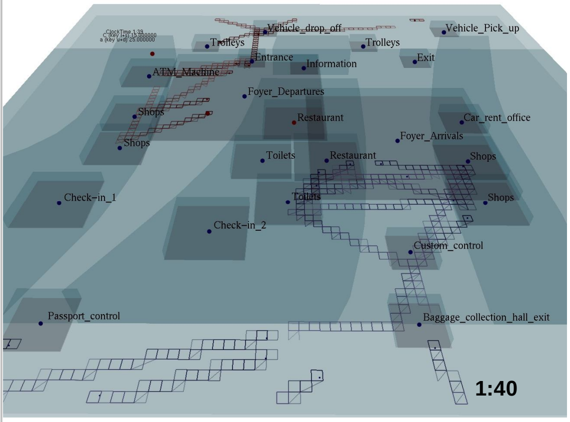
Passenger behavioural rules

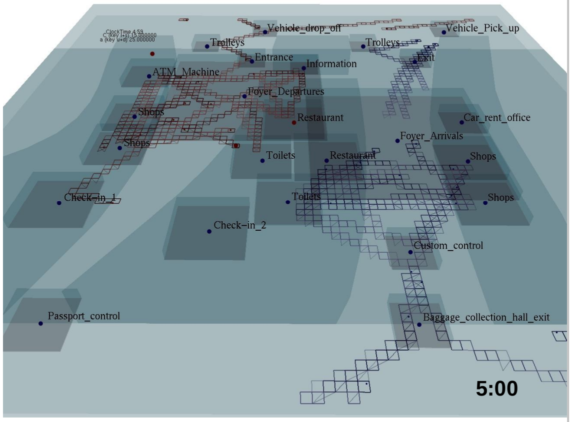
Passenger interaction rules

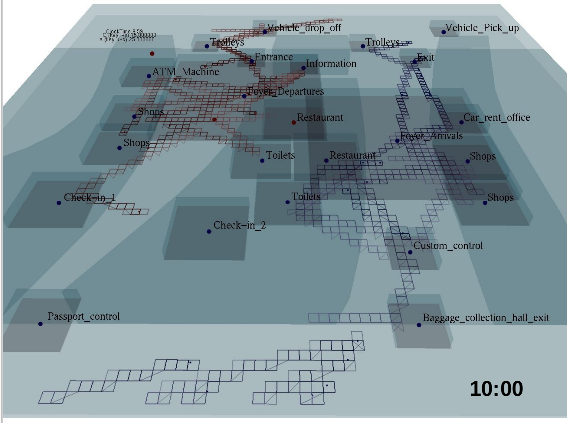
1. Repulsive effect (see section 5.6.2.1)
2. Variable $a_{\text{interperson repel}} = 25$ length units
3. Movement speed:2.5 length/time units

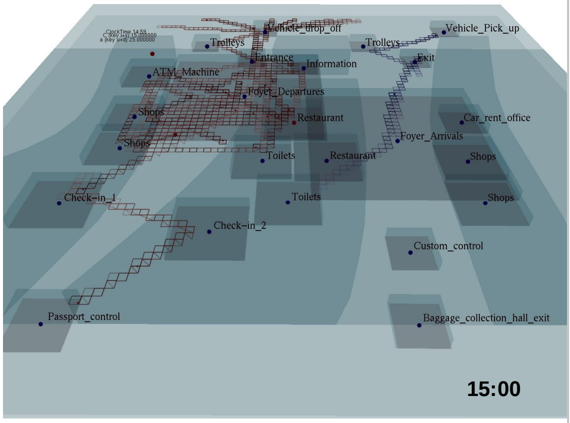
Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 15$ force units









Appendix F

Appendix F presents the studies at the micro-scale where the circulation design and the configuration of passengers' facilities in the restaurant area/foyer were examined. At this scale the studies use three different approaches of modelling. In the first approach, circulation diagrams in a given area are generated. In the second, stationary passengers' facilities are positioned and circulation diagrams are produced, and in the third approach passengers' facilities are allowed to re-adjust their position in relation to passengers' movement behaviour. Parameters are organized as follow:

The initial geometrical configuration consists of parameters that specify the virtual environment in 2D dimensions. The boundaries that interact with the facilities or other passengers are included and their number varies according to the design solution. Also, the last destination and the entry points vary according to user's choice. The number of signs varies and it depends on the number of destinations. The total functional area in square meters is 240m² and the time step interval for each image captured is set to 1:40 minutes (virtual clock time).

The passengers' initial parameters consist of: the number of passenger groups that varies according to the design case, the number of passengers in each group, the number of destinations and the group distribution, and finally the time of appearance that is random.

The passenger behavioural rules are: the repulsive effect, the value of variable $a_{\text{interperson repel}}$ that is set to 60 length units, and the movement speed that is set to 5.0 length/time units.

Finally, the passengers' movement rules include the signs distribution and the value of sign strength C_{parallel} that is defined as 50 force units.

1. Circulation design and configuration of passenger facilities

• Initial simulation/circulation diagrams

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 100Units
2. Boundaries: 5
3. Last destinations: 2, Entry points: 2
4. Signs: 12 destinations x50x50 signs
5. Total functional area in square meters: 240m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 25x25, 2nd arrival group 25x25
3. No of destinations and group distribution: 12 (0-1) destinations,
   1st group 0-5 destination: compulsory and optional 2nd group 6-11 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

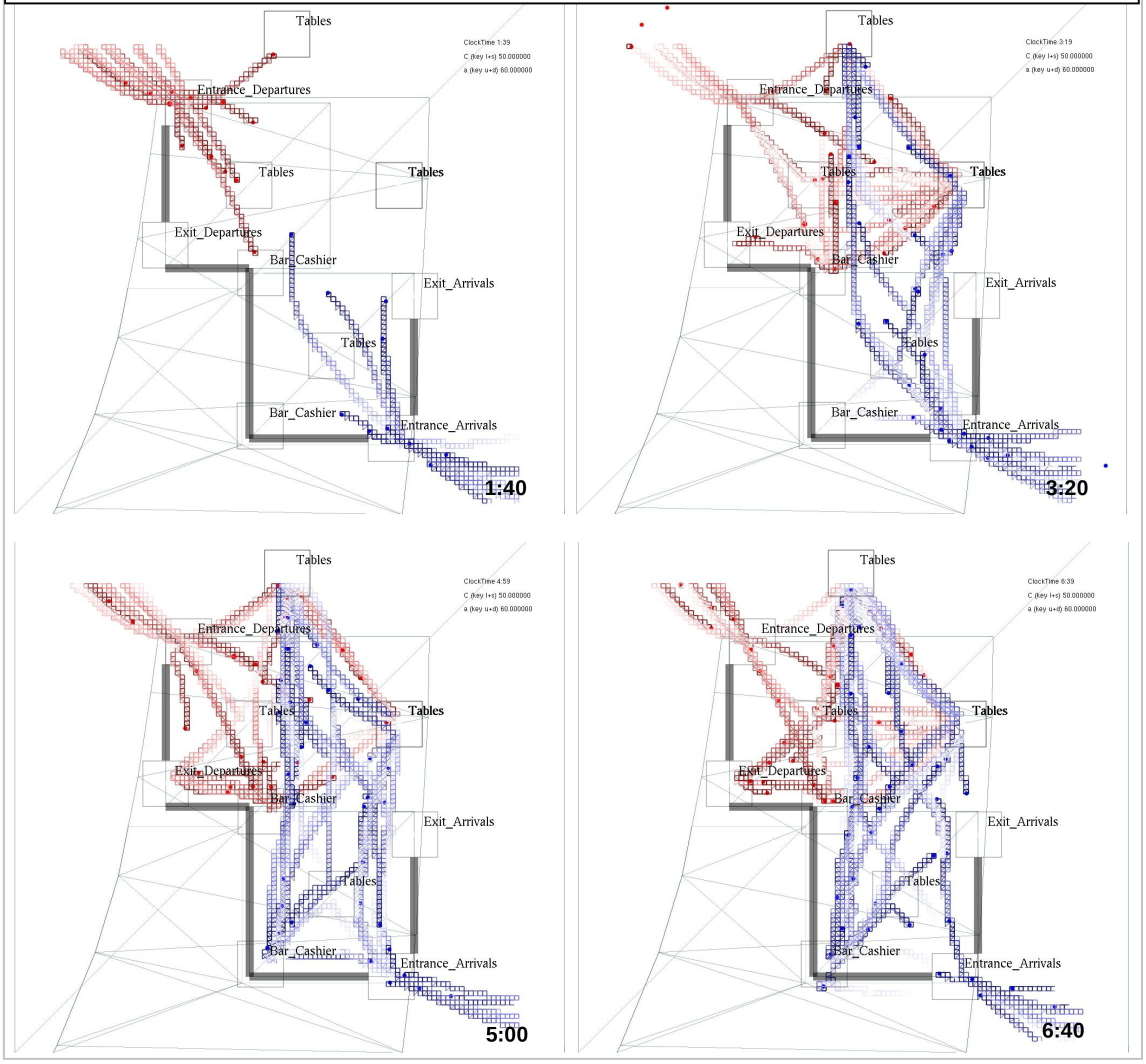
Passenger behavioural rules

Passenger interaction rules

1. Repulsive, obstacle and boundaries avoidance effects (see chapter 5)
2. Variable $a_{\text{interperson repel}} = 60$ length units
3. Movement speed:5.0 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 50$ force units



2. Circulation design and configuration of passenger facilities

- Initial simulation/1st approach of modelling

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 100Units
2. Boundaries: 5
3. Last destinations: 2, Entry points: 2
4. Signs: 12 destinations x50x50 signs
5. Total functional area in square meters: 240m²
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 25x25, 2nd arrival group 25x25
3. No of destinations and group distribution: 12 (0-1) destinations,
   1st group 0-5 destination: compulsory and optional 2nd group 6-11 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

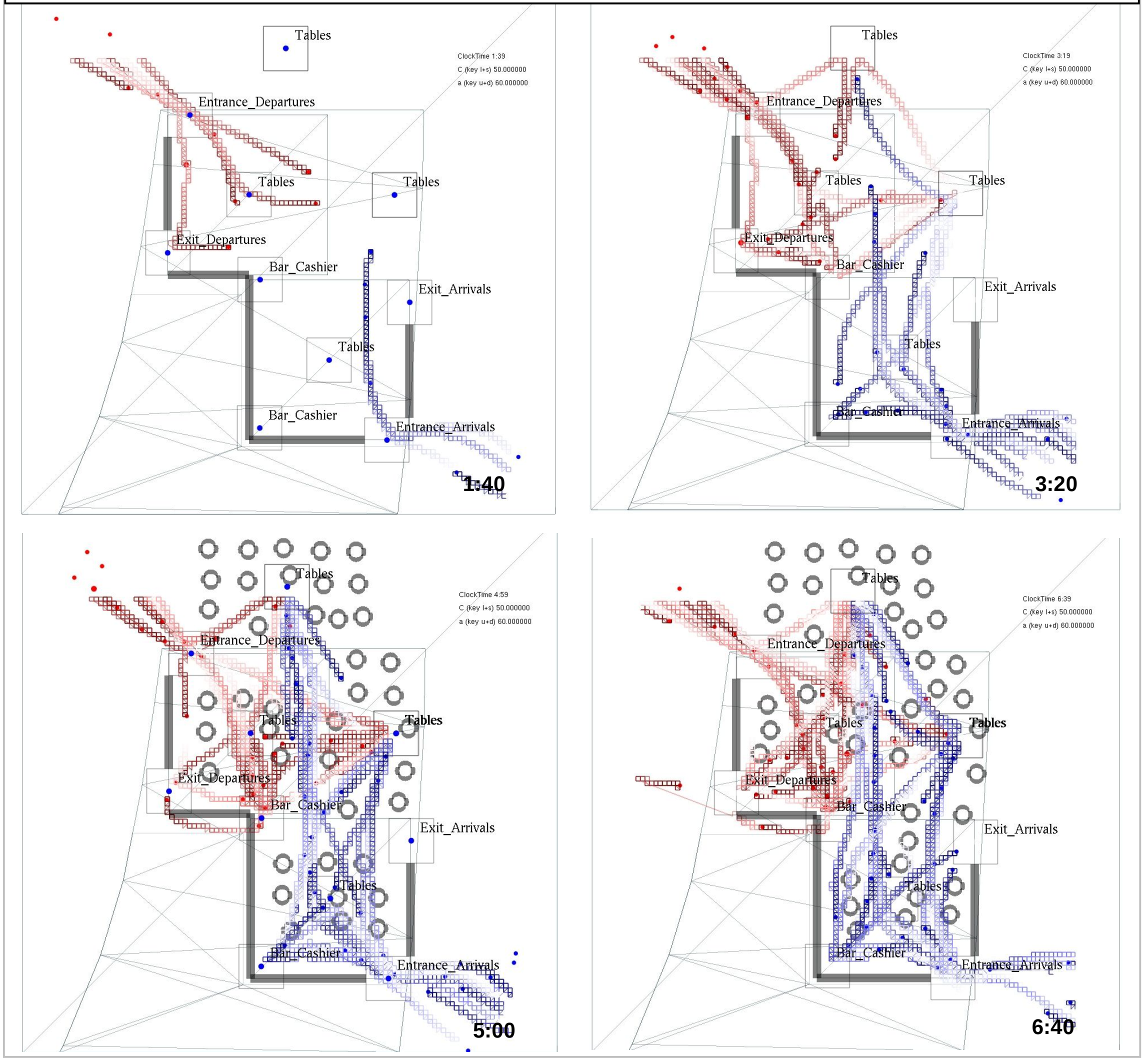
Passenger behavioural rules

Passenger interaction rules

1. Repulsive, obstacle and boundaries avoidance effects (see chapter 5)
2. Variable $a_{\text{interperson repel}} = 60$ length units
3. Movement speed:5.0 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 50$ force units



3. Circulation design and configuration of passenger facilities

• Initial simulation/2nd approach of modelling

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 100Units
2. Boundaries: 5
3. Last destinations: 2, Entry points: 2
4. Signs: 12 destinations x50x50 signs
5. Total functional area in square meters: 240m2
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 25x25, 2nd arrival group 25x25
3. No of destinations and group distribution: 12 (0-1) destinations,
   1st group 0-5 destination: compulsory and optional 2nd group 6-11 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

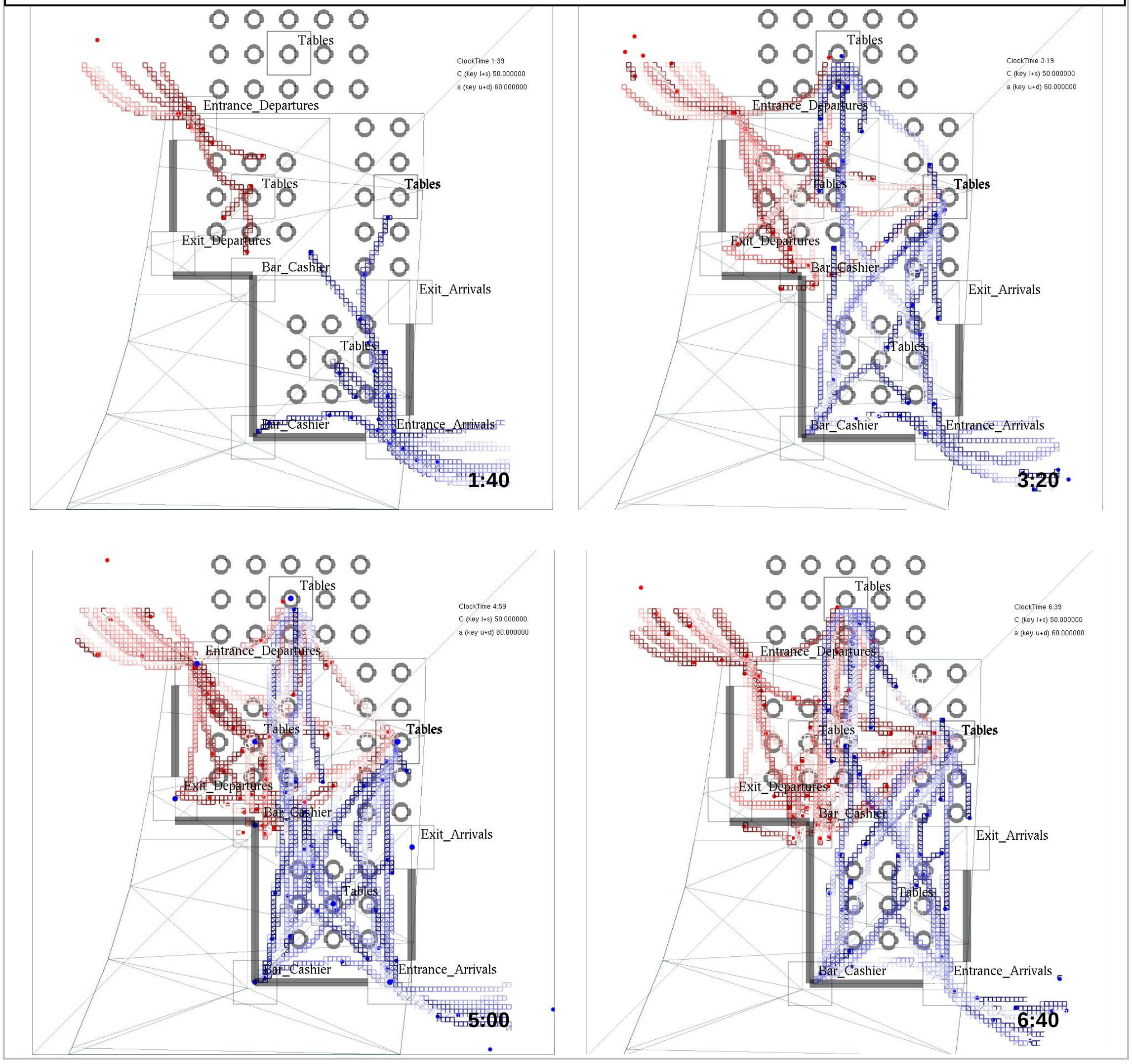
Passenger behavioural rules

Passenger interaction rules

1. Repulsive, obstacle and boundaries avoidance effects (see chapter 5)
2. Variable $a_{\text{interperson repel}} = 60$ length units
3. Movement speed:5.0 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 50$ force units



4. Circulation design and configuration of passenger facilities

- **Initial simulation/3rd approach of modelling**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 100Units
2. Boundaries: 5
3. Last destinations: 2, Entry points: 2
4. Signs: 12 destinations x50x50 signs
5. Total functional area in square meters: 240m2
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 2
2. No of people in each group: 1st departure group 25x25, 2nd arrival group 25x25
3. No of destinations and group distribution: 12 (0-1) destinations,
   1st group 0-5 destination: compulsory and optional 2nd group 6-11 destinations: compulsory and optional
4. Entry points: 2
5. Time of appearance: Random

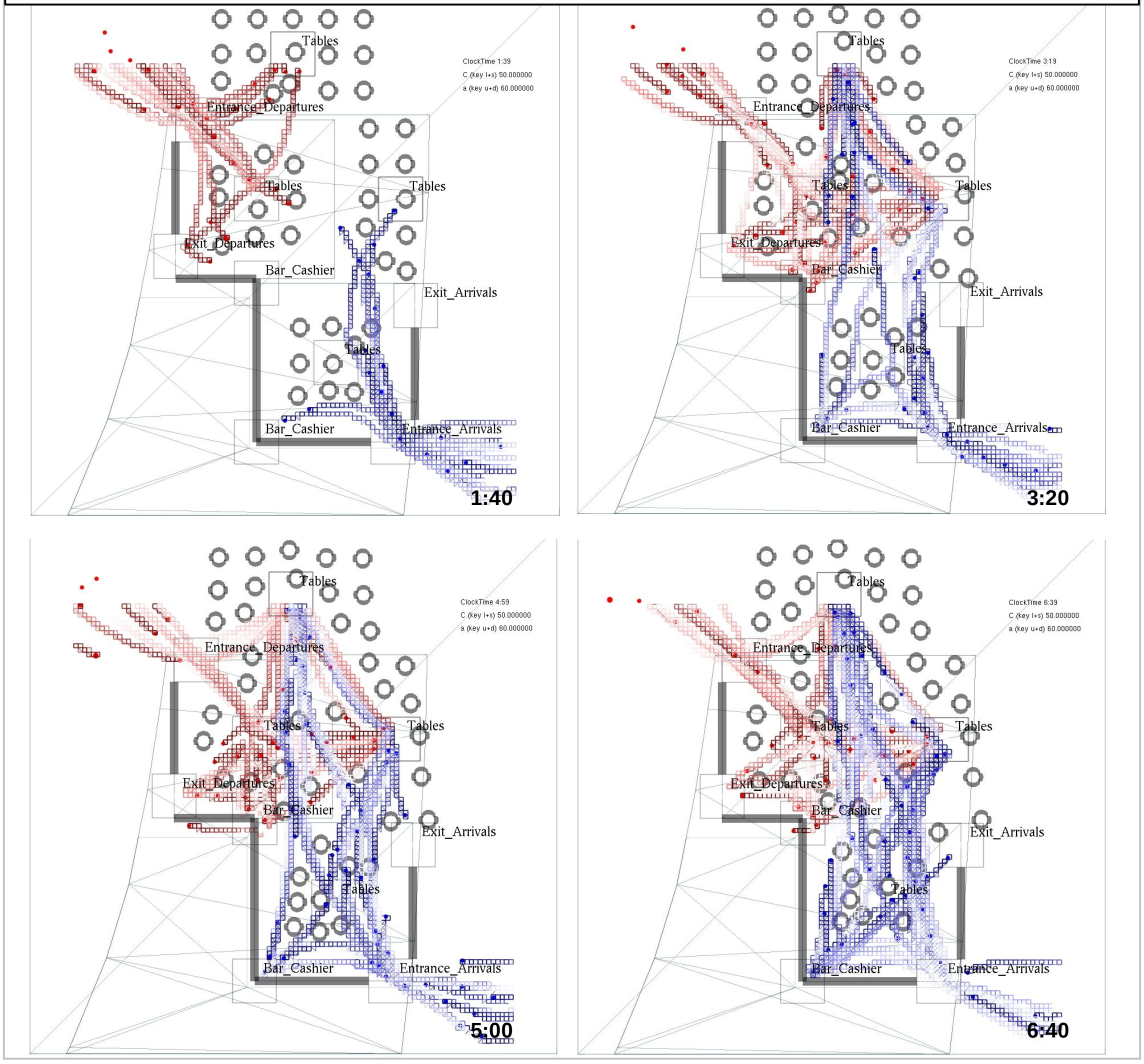
Passenger behavioural rules

Passenger interaction rules

1. Repulsive, obstacle and boundaries avoidance effects (see chapter 5)
2. Variable $a_{\text{interperson repel}} = 60$ length units
3. Movement speed:5.0 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 50$ force units



5. Circulation design and configuration of passenger facilities

- **1^{st} suggested design/1^{st} approach of modelling/circulation diagrams**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 100Units
2. Boundaries: 9
3. Last destinations: 3, Entry points: 3
4. Signs: 17 destinations x50x50 signs
5. Total functional area in square meters: 295m2
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 3
2. No of people in each group: 1^{st} departure group 25x25, 2^{nd} departure group 25x25, 3^{rd} arrival group 25x25
3. No of destinations and group distribution: 17 (0-16) destinations, 1^{st} group 0-5 destination: compulsory and optional, 2^{nd} group 6-10 destinations: compulsory and optional, 3^{rd} group 11-16: compulsory and optional
4. Entry points: 3
5. Time of appearance: Random

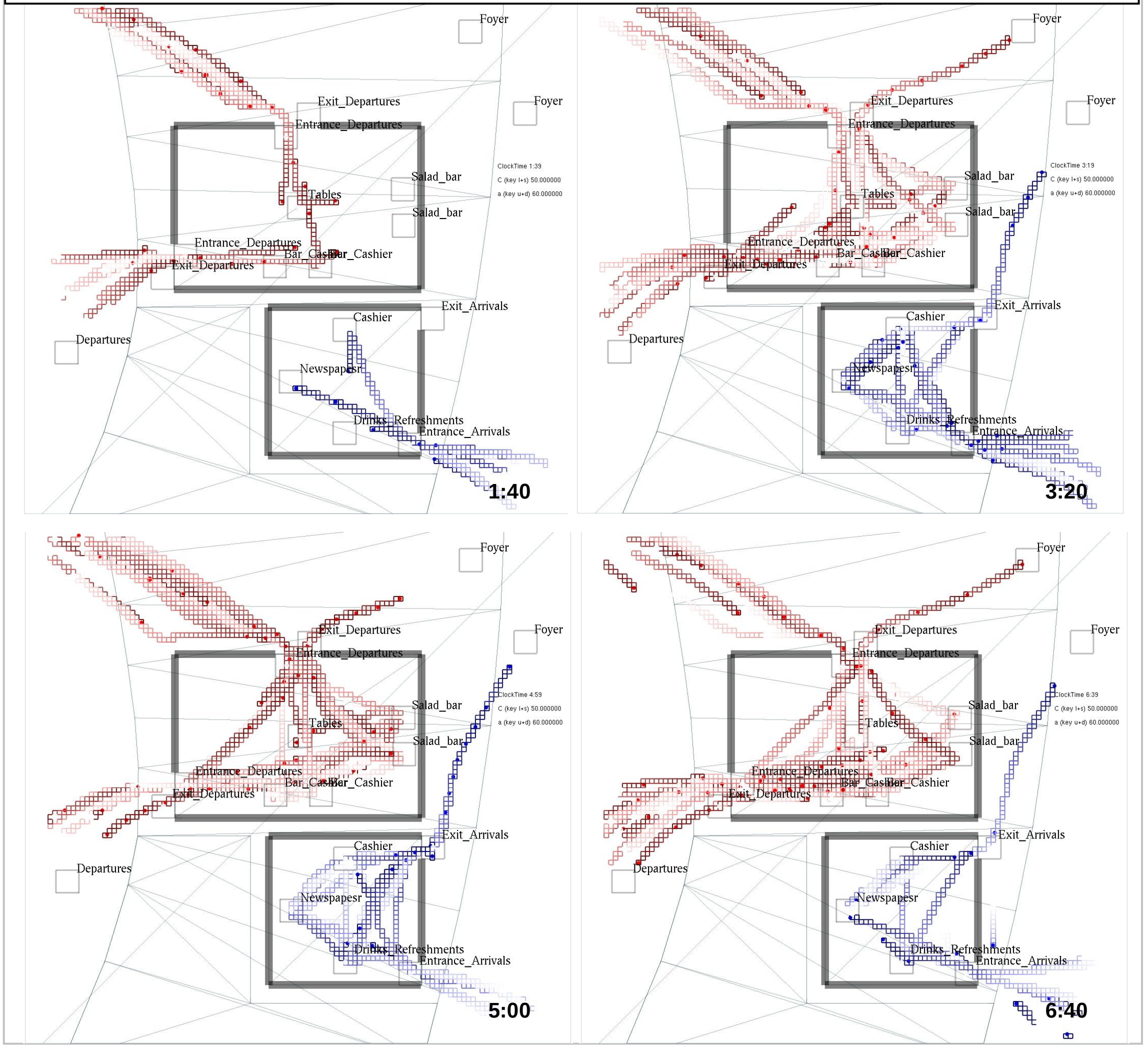
Passenger behavioural rules

Passenger interaction rules

1. Repulsive, obstacle and boundaries avoidance effects (see chapter 5)
2. Variable $a_{\text{interperson repel}} = 60$ length units
3. Movement speed:5.0 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 50$ force units



6. Circulation design and configuration of passenger facilities

• 1st suggested design/2nd approach of modelling

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 100Units
2. Boundaries: 9
3. Last destinations: 3, Entry points: 3
4. Signs: 12 destinations x50x50 signs
5. Total functional area in square meters: 295m2
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 3
2. No of people in each group: 1st departure group 25x25, 2nd departure group 25x25, 3rd arrival group 25x25
   No of destinations and group distribution: 17 (0-16) destinations,
   1st group 0-5 destination: compulsory and optional, 2nd group 6-10 destinations: compulsory and optional,
   3rd group 11-16: compulsory and optional
3. Entry points: 3
4. Time of appearance: Random

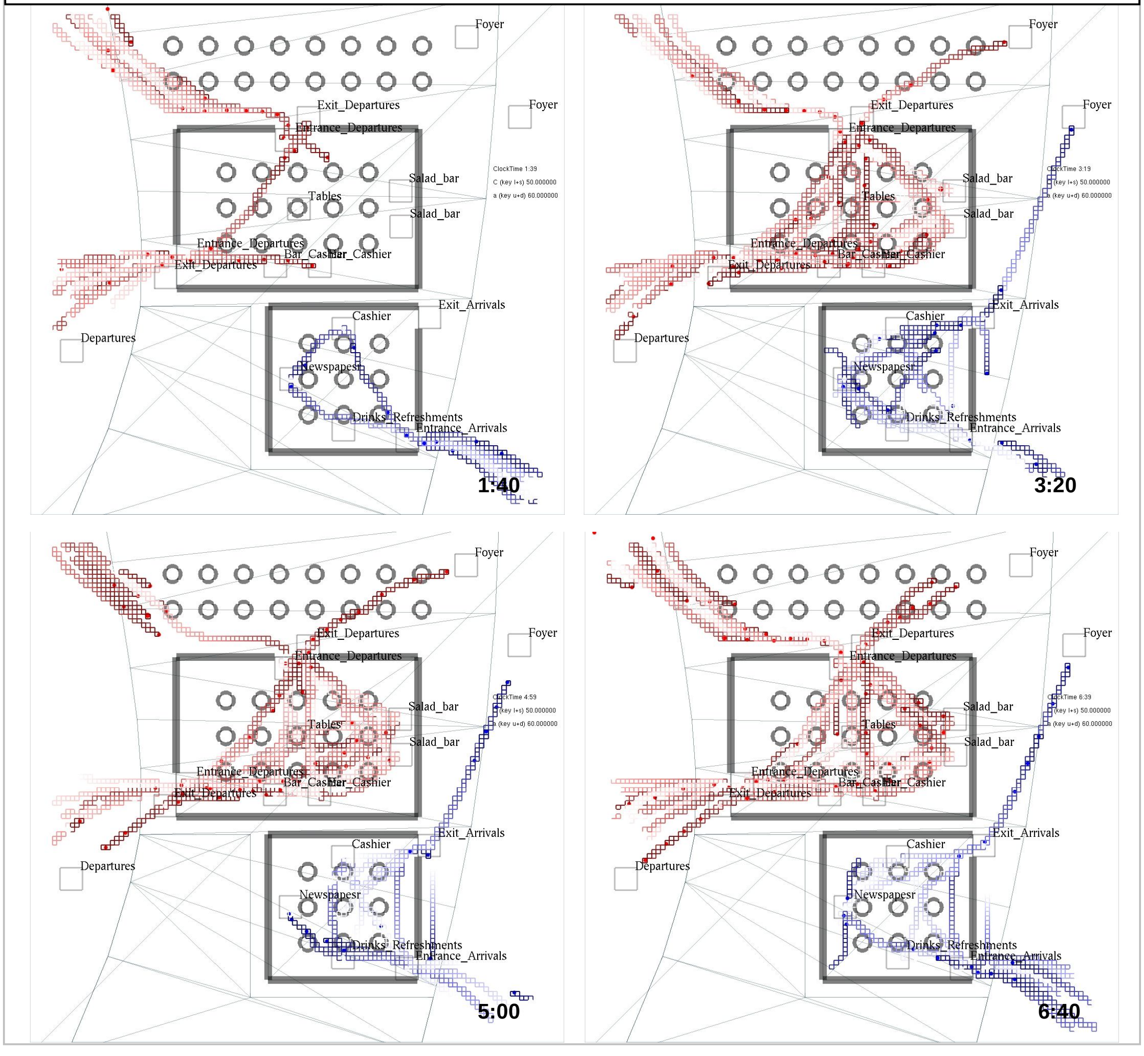
Passenger behavioural rules

Passenger interaction rules

1. Repulsive, obstacle and boundaries avoidance effects (see chapter 5)
2. Variable $a_{\text{interperson repel}} = 60$ length units
3. Movement speed:5.0 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 50$ force units



7. Circulation design and configuration of passenger facilities

- **1^{st} suggested design/3^{rd} approach of modelling**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 100Units
2. Boundaries: 9
3. Last destinations: 3, Entry points: 3
4. Signs: 12 destinations x50x50 signs
5. Total functional area in square meters: 295m2
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 3
2. No of people in each group: 1^{st} departure group 25x25, 2^{nd} departure group 25x25, 3^{rd} arrival group 25x25
   No of destinations and group distribution: 17 (0-16) destinations,
   1^{st} group 0-5 destination: compulsory and optional, 2^{nd} group 6-10 destinations: compulsory and optional,
   3^{rd} group 11-16: compulsory and optional
3. Entry points: 3
4. Time of appearance: Random

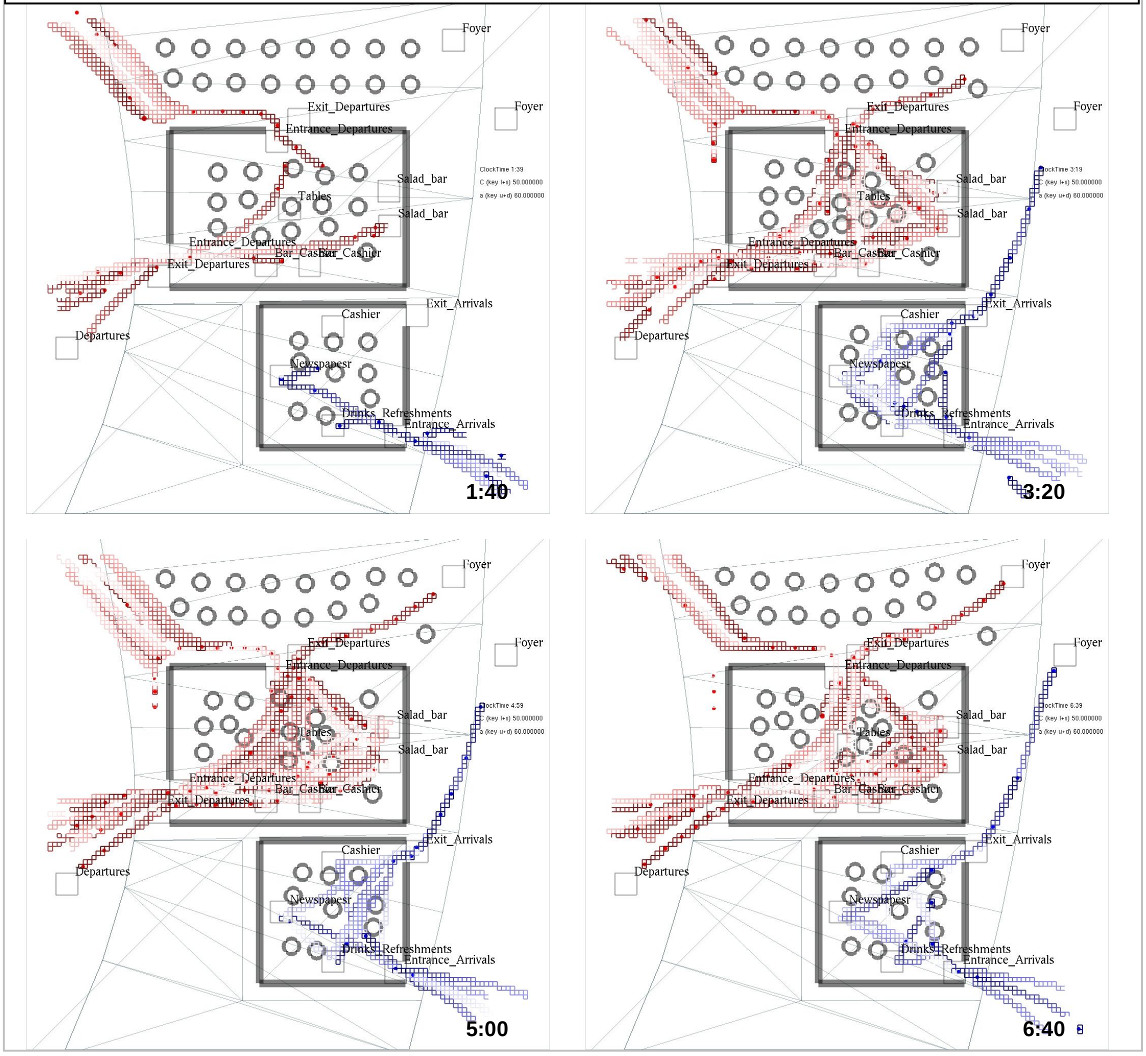
Passenger behavioural rules

Passenger interaction rules

1. Repulsive, obstacle and boundaries avoidance effects (see chapter 5)
2. Variable $a_{\text{interperson repel}} = 60$ length units
3. Movement speed:5.0 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 50$ force units



8. Circulation design and configuration of passenger facilities

- 2nd suggested design/1st approach of modelling/circulation diagrams

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 100Units
2. Boundaries: 9
3. Last destinations: 3, Entry points: 4
4. Signs: 12 destinations x50x50 signs
5. Total functional area in square meters: 295m2
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 4
2. No of people in each group: 1st departure group 25x25, 2nd departure group 25x25, 3rd arrival group 25x25, 4th arrival group 25x25
3. No of destinations and group distribution: 21 (0-20) destinations,
   1st group 0-5 destination: compulsory and optional 2nd group 6-10 destinations: compulsory and optional
   3rd group11-16 destination: compulsory and optional 4th group 17-20 destinations: compulsory and optional
4. Entry points: 4
5. Time of appearance: Random

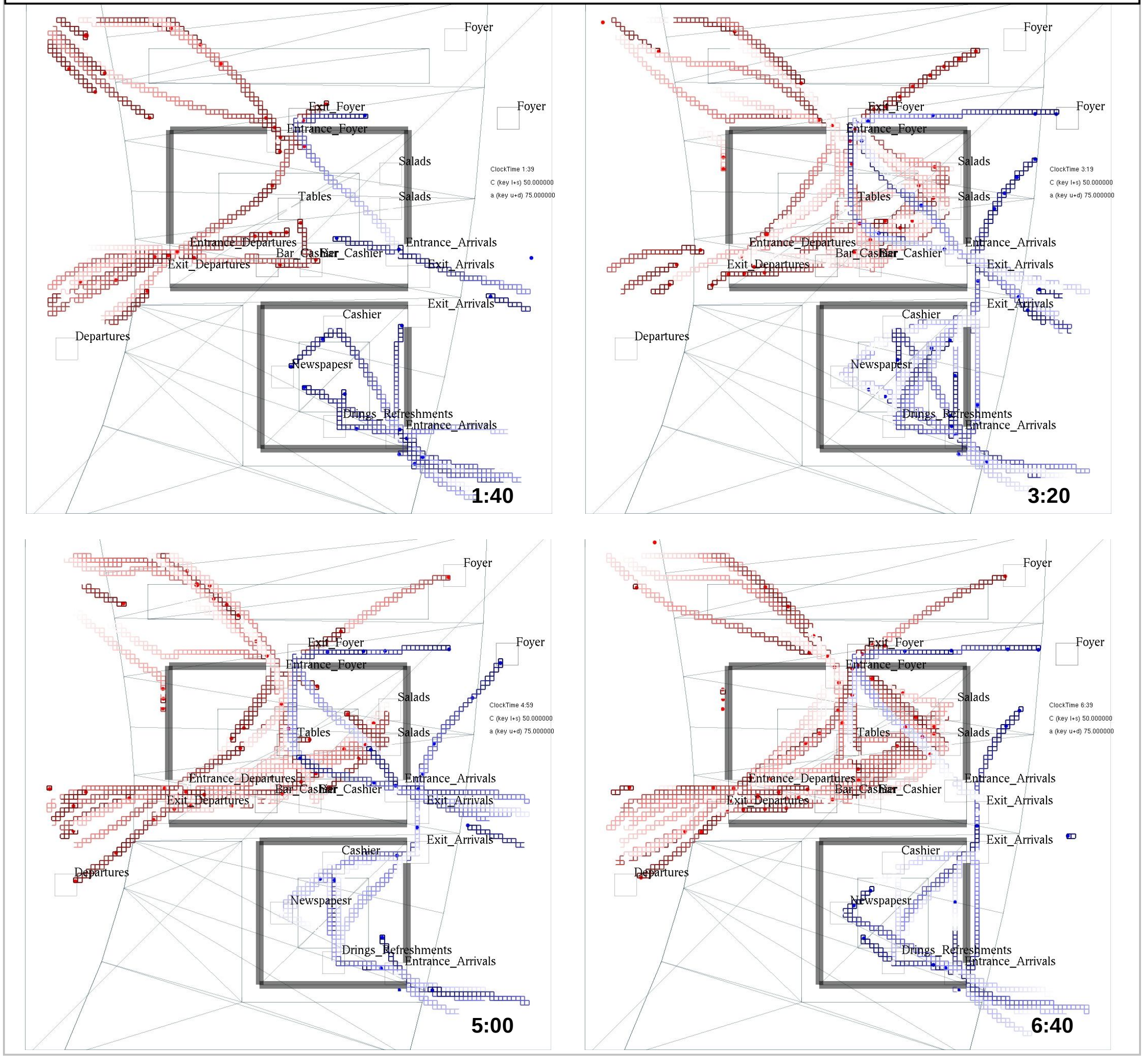
Passenger behavioural rules

Passenger interaction rules

1. Repulsive, obstacle and boundaries avoidance effects (see chapter 5)
2. Variable $a_{\text{interperson repel}} = 75$ length units
3. Movement speed:1st, 2nd and 3rd group 5.0 length/time units, 4th group 10.0 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 50$ force units



9. Circulation design and configuration of passenger facilities

- **2nd suggested design/2nd approach of modelling**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 100Units
2. Boundaries: 9
3. Last destinations: 3, Entry points: 4
4. Signs: 12 destinations x50x50 signs
5. Total functional area in square meters: 295m2
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 4
2. No of people in each group: 1st departure group 25x25, 2nd departure group 25x25,
   3rd arrival group 25x25, 4th arrival group 25x25
3. No of destinations and group distribution: 21 (0-20) destinations,
   1st group 0-5 destination: compulsory and optional 2nd group 6-10 destinations: compulsory and optional
   3rd group11-16 destination: compulsory and optional 4th group 17-20 destinations: compulsory and optional
4. Entry points: 4
5. Time of appearance: Random

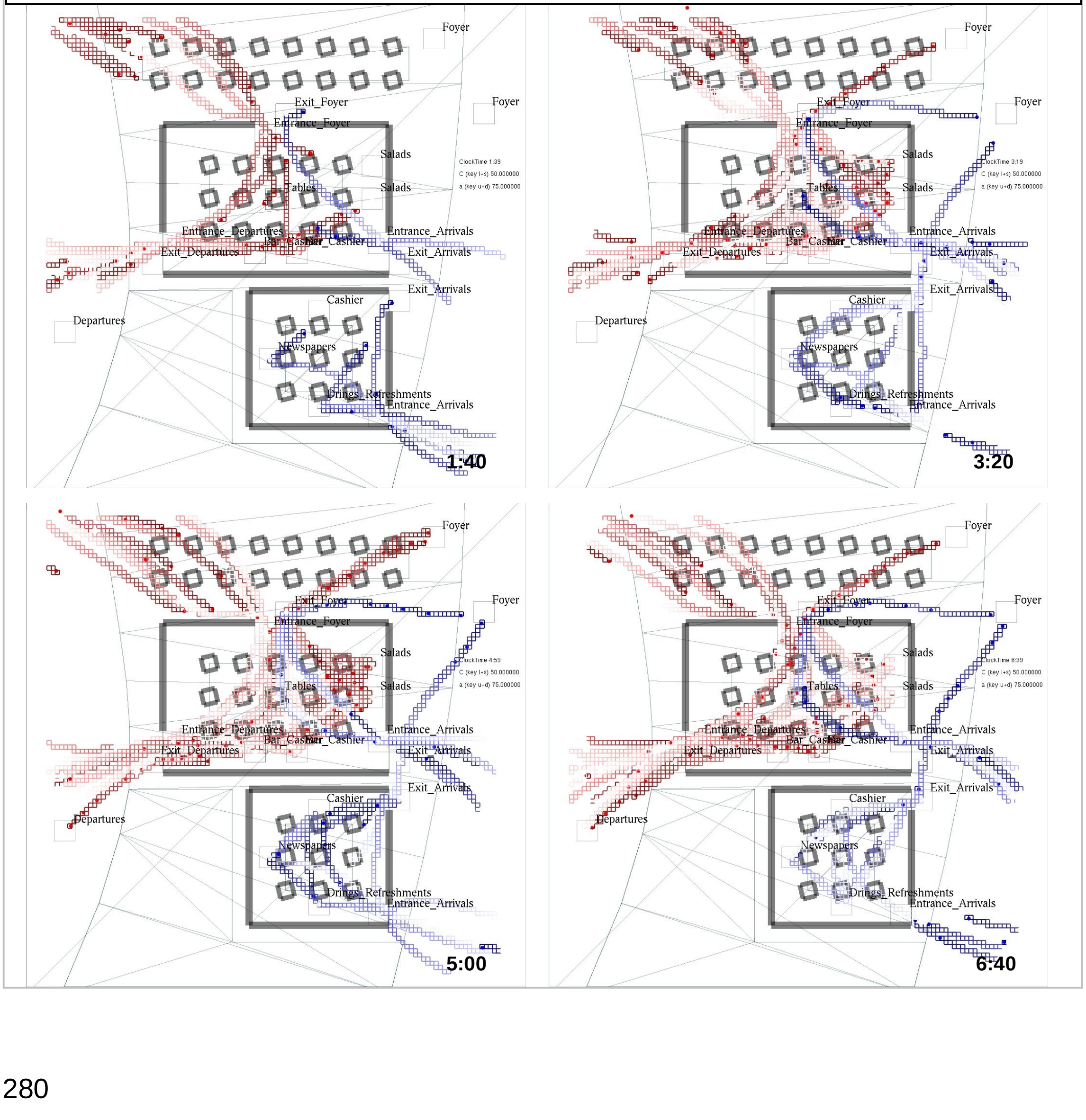
Passenger behavioural rules

Passenger interaction rules

1. Repulsive, obstacle and boundaries avoidance effects (see chapter 5)
2. Variable $a_{\text{interperson repel}} = 75$ length units
3. Movement speed:1st, 2nd and 3rd group 5.0 length/time units, 4th group 10.0 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 50$ force units



10. Circulation design and configuration of passenger facilities

- **2nd suggested design/3rd approach of modelling**

Initial parameter

Initial geometrical configuration

1. 2D Virtual Environment, scale: 1m: 100Units
2. Boundaries: 9
3. Last destinations: 3, Entry points: 4
4. Signs: 12 destinations x50x50 signs
5. Total functional area in square meters: 295m2
6. Time step interval: (200 units) 1:40 minutes virtual clock time

Passenger initial features

1. No of groups: 4
2. No of people in each group: 1st departure group 25x25, 2nd departure group 25x25, 3rd arrival group, 3rd arrival group 25x25, 4th arrival group 25x25
3. No of destinations and group distribution: 21 (0-20) destinations, 1st group 0-5 destination: compulsory and optional 2nd group 6-10 destinations: compulsory and optional 3rd group11-16 destination: compulsory and optional 4th group 17-20 destinations: compulsory and optional
4. Entry points: 4
5. Time of appearance: Random

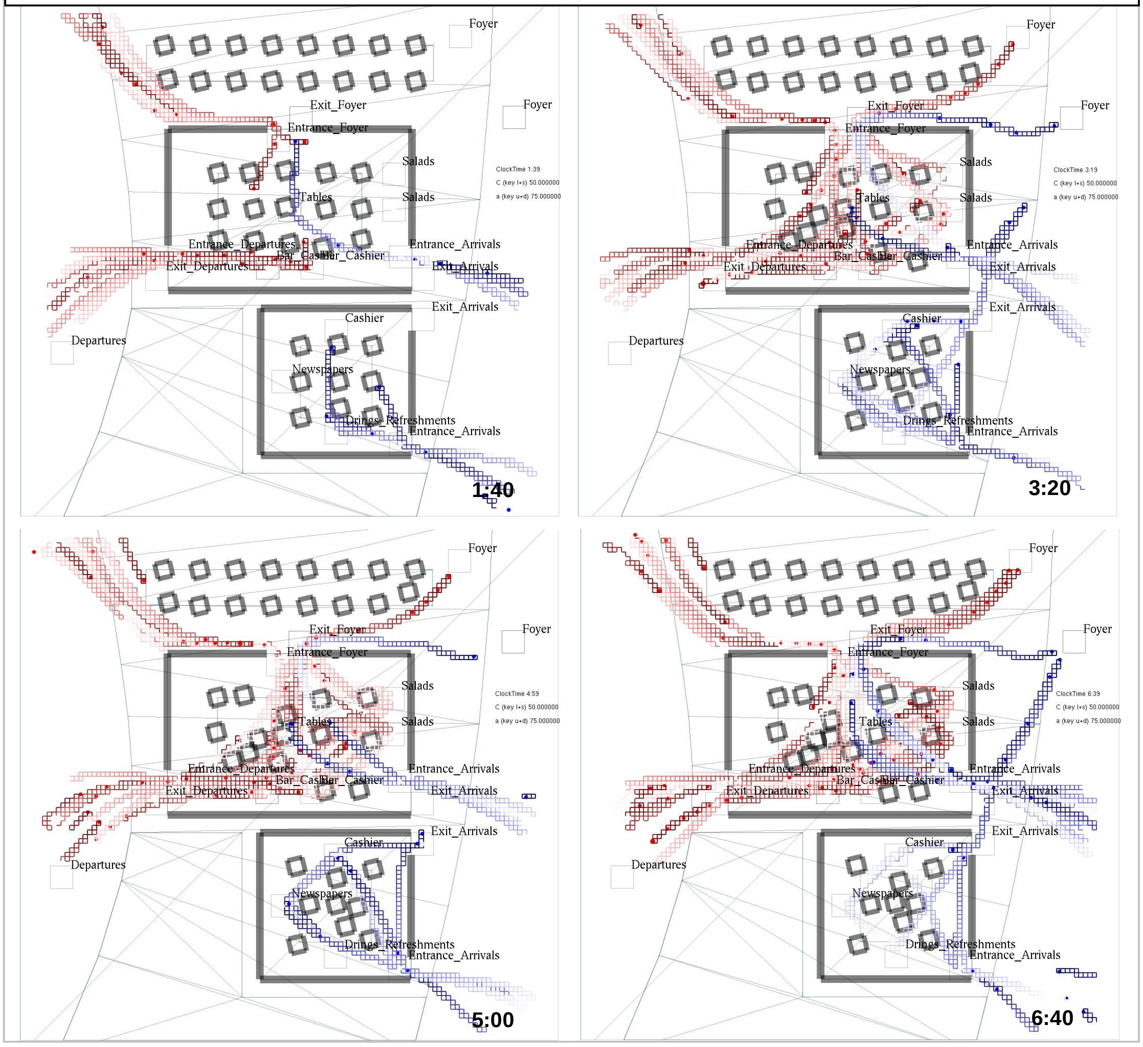
Passenger behavioural rules

Passenger interaction rules

1. Repulsive, obstacle and boundaries avoidance effects (see chapter 5)
2. Variable $a_{\text{interperson repel}} = 75$ length units
3. Movement speed:1st, 2nd and 3rd group 5.0 length/time units, 4th group 10.0 length/time units

Passenger movement rules

1. Sign distribution: random grid (50x50)
2. Sign strength $C_{\text{parallel}} = 50$ force units



Appendix G

Appendix G presents the C++ source code for the design of path systems (macro-scale). The code was written specifically for the 1st suggested design simulation. It was developed following the same structure with the simple programs that are demonstrated in appendix B. It can be used as reference, providing an explanation of the program used in macro-scale study, and demonstrating its role in the design process.

The source code includes functions for capturing .tga images, for creating output .dxf files, mouse function for drag and move, and so on. However, the .txt files that specify the position of destinations are not included. This can be easily created by the users according to their specific design problems.

Design of Path Systems

Source code in C++

```
#include <iostream>
#include <fstream>
#include <cmath>
#include <cstdlib>
#include <time.h>
using namespace std;
//3D Loader
#include "Port.h"
#include "3dsloader.h"
//#include <GLUT/glut.h>  // For Macintosh
#include <GL/glut.h>    // For PC or Sun
#define   Macintosh 0   //1 for Macintosh any other number for PC or Sun
#define AbsoluteLastPerson 100000
#define AbsoluteLastSign 500
#define AbsoluteLastDestination   500
#define AbsoluteLastTime 200//Default value 100
#define HalfWidth     650
#define HalfHeight    500
#define AbsoluteLastK 1000
#define AbsoluteLastL 1000
void Graphics(float BackGroundRed,float BackGroundBlue,float BackGroundGreen);
void DoTheDrawing(void);
static void Draw(void);
void CalculateMotion(void);
static void Key(unsigned char key, int x, int y);
void MouseClick(int button, int state, int x, int y);
void MouseDrag(int x, int y);
void CreateData(void);
void WriteDXFFile(void);
void Rewrite(void);
void SavePicture(void);
typedef struct
{
   char id_len;                 // ID Field (Number of bytes - max 255)
   char map_type;               // Colormap Field (0 or 1)
   char img_type;               // Image Type (7 options - color vs. compression)
   int  map_first;              // Color Map stuff - first entry index
   int  map_len;                // Color Map stuff - total entries in file
   char map_entry_size;         // Color Map stuff - number of bits per entry
   int  x;                      // X-coordinate of origin
   int  y;                      // Y-coordinate of origin
   int  width;                  // Width in Pixels
   int  height;                 // Height in Pixels
   char bpp;                    // Number of bits per pixel
   char misc;                   // Other stuff - scan origin and alpha bits
}targa_header;
void writeheader(targa_header h, FILE *tga);
GLUquadricObj *quadObj;
float  Coord[AbsoluteLastTime+1][2][AbsoluteLastPerson+1],
       Vely[2][AbsoluteLastPerson+1],
       SignVely[2][AbsoluteLastPerson+1],RepulsionOverMass[2][AbsoluteLastPerson+1],
       RepulsiveVely[2][AbsoluteLastPerson+1],
       Colour[3][AbsoluteLastPerson+1],
       RepulsiveForce[2][AbsoluteLastPerson+1],
       PI,
       aSq,
       XWalking,
       YWalking,
       SignCoord[2][AbsoluteLastSign+1][AbsoluteLastDestination+1],
       Array,
       C,
       Theta[AbsoluteLastSign+1][AbsoluteLastDestination+1],
       ArrayCoord[2][AbsoluteLastSign+1][AbsoluteLastDestination+1],
       Walking,
       RepulsionOverMassFactor,
       RepulsionDecayLengthSq,
       SignMotionFactor,
       LastSign[AbsoluteLastDestination+1],
       drawx[2],
       TargetCoord[2][AbsoluteLastSign+1][AbsoluteLastDestination+1],
       TargetDistance[AbsoluteLastPerson+1],
       L,
       R[AbsoluteLastSign+1][AbsoluteLastDestination+1],
       LSq,
       SeparationSq,
       CaptureRadiusSq,
       InitialZoom,
       Zoom,
       ZoomIncrement,
       a,
       PreviousCoord[2],
       x[AbsoluteLastK+1][AbsoluteLastK+1],
       y[AbsoluteLastL+1][AbsoluteLastL+1],
       DestinationColour[3][AbsoluteLastDestination+1],
       FileNumber,DrawSaveInterval,
       deltat,mass;
int    Person,
       ThisPerson,
       LastPerson,
       xyz,
       clickx,
       clicky,
       argc,
       LengthOfString,
       MyCharacter,
       LastTime,
       Time,
       DrawTime,
       Sign,
```

```
       ThisSign,
       ThisDestination,
       LastDestination,
       Destination,
       WhichDestination[AbsoluteLastPerson+1],
       ArrivalTime[AbsoluteLastPerson+1],
       Active[AbsoluteLastPerson+1],
       TimeBeforePresent,
       ClockTime,
       CapturedSign,
       CapturedDestination,
       PreviousCapturedSign,
       PreviousCapturedDestination,
       MousePosition_x,
       MousePosition_y,
       Scale=1,
       l_index,
       PersonHistory,
       TimeStepFactor[AbsoluteLastPerson+1],
       k,
       l,
       LastK,
       LastL;
//this will hold the openGL texture object for the loaded texture
unsigned int g_Texture=0;
char **argv;
bool LeftMouseDown=0,RightMouseDown=0;
char NumericalValue[101];
char *MyText;
ifstream Dim("Signs.txt");
ifstream Odysseas("Areas.txt");
ofstream Maud("People.dxf");
//Flag for rendering as lines or filled polygons
int filling=1; //0=OFF 1=ON
obj_type object;
int main(void)
{
    srand(time(NULL));
    PI=4.0*atan(1.0);
    L=75.0;//Default value 200.0
    LSq=L*L;
    CaptureRadiusSq=30;
    CapturedSign=-1;
    CapturedDestination=-1;
    PreviousCapturedSign=-1;
    PreviousCapturedDestination=-1;
    MousePosition_x=0;
    MousePosition_y=0;
    cout<<"This program runs forever until the letter 'q' is pressed on the keyboard\n\n";
    C=50.0;//By changing the value of C, trails' curves are also changed
           //Default value C=35.0 gives intersting results
    aSq=100000.0;
    Array=50.0;
    deltat=1.0;mass=1.0;
    a=25.0;//Default value 35.0
    RepulsionDecayLengthSq=0.2*a*a;//Default value 0.2*a*a
    RepulsionOverMassFactor=0.5;//Default value 0.5
    Dim>>LastDestination;
    for(Destination=0;Destination<=LastDestination;Destination++)
    {
        Dim>>ThisDestination;
        Dim>>LastSign[Destination];
        for(Sign=0;Sign<=LastSign[Destination];Sign++)
        {
            Dim>>ThisSign;
            Dim>>ThisDestination>>SignCoord[0][ThisSign][ThisDestination]>>SignCoord[1][ThisSign][ThisDestination]
               >>Theta[ThisSign][ThisDestination];
        }
    }
    cout<<"Read signs\n\n";
    Odysseas>>LastDestination;
    for(Destination=0;Destination<=LastDestination;Destination++)
    {
        Odysseas>>ThisDestination;
        Odysseas>>LastSign[Destination];
        for(Sign=0;Sign<=LastSign[Destination];Sign++)
        {
            Odysseas>>ThisSign;
            Odysseas>>ThisDestination>>R[ThisSign][ThisDestination];
        }
    }
    cout<<"Read functional areas\n\n";
    CreateData();
    Maud<<"0\nSECTION\n2\nENTITIES\n";
    Graphics(1.0,1.0,1.0);
    return 0;
}
static void Draw(void)
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    ClockTime++;
    CalculateMotion();
    DoTheDrawing();
    if(ClockTime==DrawSaveInterval*int(ClockTime/DrawSaveInterval))
    SavePicture();
}
void Graphics(float BackGroundRed,float BackGroundGreen,float BackGroundBlue)
{
    glutInitWindowSize(2*HalfWidth,2*HalfHeight);
    //if(Macintosh==1)glutInit(&argc, argv);//Remove this line for PC or Sun
    glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE | GLUT_DEPTH);
    glutCreateWindow("Path Systems");
    glClearColor(BackGroundRed,BackGroundGreen,BackGroundBlue,0);
    Load3DS (&object,"Boundaries.3ds");//3DS Loader
    gluOrtho2D(-HalfWidth,HalfWidth,-HalfHeight,HalfHeight);
    glClear(GL_COLOR_BUFFER_BIT);
    glutKeyboardFunc(Key);
```

```
    glutIdleFunc(Draw);
    glutDisplayFunc(Draw);
    glutMouseFunc(MouseClick);
    glutMotionFunc(MouseDrag);
    glEnable(GL_POINT_SMOOTH);//glDisable(GL_POINT_SMOOTH) is the reverse. Enable means antialiasing
    glEnable(GL_BLEND);//glDisable(GL_BLEND) does not allow blending
    glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    glShadeModel(GL_FLAT);
    glutMainLoop();
}
static void Key(unsigned char key, int x, int y)
{
    switch (key)
    {
        case 'z':if(Scale==0)Scale=1;else Scale=0;break;
        case 'f':SavePicture();break;
        case 'r': case 'R':
        if (filling==0)
        {
            glPolygonMode (GL_FRONT_AND_BACK, GL_FILL); //Polygon rasterization mode (polygon filled)
            filling=1;
        }
        else
        {
            glPolygonMode (GL_FRONT_AND_BACK, GL_LINE); //Polygon rasterization mode (polygon outlined)
            filling=0;
        }
        break;
        //Change the C value
        case 'l':C=C*1.01;break;
        case 's':C=C/1.01;break;
        //Change the a value
        case 'u':a=a*1.01;RepulsionDecayLengthSq=0.2*a*a;break;
        case 'd':a=a/1.01;RepulsionDecayLengthSq=0.2*a*a;break;
        //Show Person History
        case 'p':if(PersonHistory==0)PersonHistory=1;else PersonHistory=0;break;
        //Move signs
        case 'h':SignCoord[0][CapturedSign][CapturedDestination]=SignCoord[0][CapturedSign][CapturedDestination]+10;
glutPostRedisplay();break;
        case 'H':SignCoord[0][CapturedSign][CapturedDestination]=SignCoord[0][CapturedSign][CapturedDestination]-10;
glutPostRedisplay();break;
        case 'v':SignCoord[1][CapturedSign][CapturedDestination]=SignCoord[1][CapturedSign][CapturedDestination]+10;
glutPostRedisplay();break;
        case 'V':SignCoord[1][CapturedSign][CapturedDestination]=SignCoord[1][CapturedSign][CapturedDestination]-10;
glutPostRedisplay();break;
        case 'q':gluDeleteQuadric(quadObj);WriteDXFFile();exit(0);
    }
}
void MouseClick(int button, int state, int x, int y)
{
    MousePosition_x=x-HalfWidth;
    MousePosition_y=HalfHeight-y;
    if ((button==GLUT_LEFT_BUTTON)&&(state==GLUT_DOWN))
    {
        clickx=x;
        clicky=y;
        LeftMouseDown=1;
    }
    if ((button==GLUT_LEFT_BUTTON)&&(state==GLUT_UP))
    {
        LeftMouseDown=0;
        CapturedSign=-1;
    }
    if ((button==GLUT_RIGHT_BUTTON)&&(state==GLUT_DOWN))
    {
        clickx=x;
        clicky=y;
        RightMouseDown=1;
    }
    if ((button==GLUT_RIGHT_BUTTON)&&(state==GLUT_UP))
    {
        RightMouseDown=0;
    }
}
void MouseDrag(int x, int y)
{
    float ZoomIncrement,TransInc;
    if (LeftMouseDown)
    {
        if(CapturedSign!=-1)
        {
            if(CapturedDestination!=-1)
            {
                SignCoord[0][CapturedSign][CapturedDestination]+=static_cast<float>(x-clickx);
                SignCoord[1][CapturedSign][CapturedDestination]-=static_cast<float>(y-clicky);
            }
        }
        clickx=x;
        clicky=y;
    }
    if (RightMouseDown)
    {
        if(Scale==1)
        {
            ZoomIncrement=(1.0+0.001*static_cast<float>(x-clickx))*(1.0-0.001*static_cast<float>(y-clicky));
            glScalef(ZoomIncrement,ZoomIncrement,ZoomIncrement);
            Zoom=Zoom*ZoomIncrement;
        }
        clickx=x;
        clicky=y;
    }
}
void DoTheDrawing(void)
{
    //3D_Background_In this case 2DSolid
    glColor4f(0.06,0.225,0.30,0.20);//glColor4f(0.05,0.15,0.25,0.1); Default
    glBegin(GL_TRIANGLES);//glBegin and glEnd delimit the vertices that define a primitive (in our case triangles)
```

```
for (l_index=0;l_index<object.polygons_qty;l_index++)
{
    //----------------- FIRST VERTEX -----------------
    //Coordinates of the first vertex
    glVertex3f(object.vertex[ object.polygon[l_index].a ].x,
               object.vertex[ object.polygon[l_index].a ].y,
               object.vertex[ object.polygon[l_index].a ].z);

    //----------------- SECOND VERTEX -----------------
    //Coordinates of the second vertex
    glVertex3f(object.vertex[ object.polygon[l_index].b ].x,
               object.vertex[ object.polygon[l_index].b ].y,
               object.vertex[ object.polygon[l_index].b ].z);

   //----------------- THIRD VERTEX -----------------
   //Coordinates of the Third vertex
   glVertex3f(object.vertex[ object.polygon[l_index].c ].x,
              object.vertex[ object.polygon[l_index].c ].y,
              object.vertex[ object.polygon[l_index].c ].z);
}
//Draw person's position
for(Person=0;Person<=LastPerson;Person++)
{
    if(Active[Person]==1)
    {
        glColor4f(Colour[0][Person],Colour[1][Person],Colour[2][Person],1.0);

        glPointSize(5.0);
        glBegin(GL_POINTS);
            glVertex2f(Coord[0][0][Person],Coord[0][1][Person]);
        glEnd();
        glDisable(GL_LINE_STIPPLE);
        glLineWidth(1.0);
        glLineStipple(1,0x00FF);
        glBegin(GL_LINE_STRIP);
        for(TimeBeforePresent=0;TimeBeforePresent<=AbsoluteLastTime;TimeBeforePresent++)
        {
            glColor4f(Colour[0][Person],Colour[1][Person],Colour[2][Person],0.75);
            drawx[0]=Coord[TimeBeforePresent][0][Person];
            drawx[1]=Coord[TimeBeforePresent][1][Person];
            glVertex2fv(drawx);
        }
        glEnd();
    }
}
//Mouse position
glColor4f(1.0,0.0,0.0,1.0);
glPointSize(10);
glBegin(GL_POINTS);
     glVertex2f(MousePosition_x/Zoom,MousePosition_y/Zoom);
glEnd();
//For mouse drag and move
CapturedSign=-1;
CapturedDestination=-1;
for(Destination=0;Destination<=LastDestination;Destination++)
{
    for(Sign=0;Sign<=LastSign[Destination];Sign++)
    {
        SeparationSq=(1.0*MousePosition_x/Zoom-SignCoord[0][Sign][Destination])*
                     (1.0*MousePosition_x/Zoom-SignCoord[0][Sign][Destination])+
                     (1.0*MousePosition_y/Zoom-SignCoord[1][Sign][Destination])*
                     (1.0*MousePosition_y/Zoom-SignCoord[1][Sign][Destination]);
        if(SeparationSq<CaptureRadiusSq||
          (PreviousCapturedSign==Sign&&SeparationSq<500.0*CaptureRadiusSq)&&
           SeparationSq<CaptureRadiusSq||
          (PreviousCapturedDestination==Destination&&SeparationSq<500.0*CaptureRadiusSq))
        {
            CapturedSign=Sign;
            PreviousCapturedSign=Sign;
            CapturedDestination=Destination;
            PreviousCapturedDestination=Destination;
        }
    }
}
for(Destination=0;Destination<=LastDestination;Destination++)
{
    for(Sign=0;Sign<=LastSign[Destination];Sign++)
    {
        if(Sign==CapturedSign)
        {
            if(Destination==CapturedDestination)
            {
                glColor4f(1.0,0.0,0.0,1.0);
                glPointSize(20);
                glBegin(GL_POINTS);
                    glVertex2f(SignCoord[0][Sign][Destination],SignCoord[1][Sign][Destination]);
                glEnd();
            }
            else
            {
                glColor4f(0.0,0.0,1.0,1.0);
                glPointSize(20);
                glBegin(GL_POINTS);
                    glVertex2f(SignCoord[0][Sign][Destination],SignCoord[1][Sign][Destination]);
                glEnd();
            }
        }
    }
}
//Draw sign's position
for(Destination=0;Destination<=LastDestination;Destination++)
{
    for(Sign=0;Sign<=LastSign[Destination];Sign++)
    {
        glColor4f(0.0,0.0,0.0,1.0);
        glBegin(GL_POLYGON);
            glVertex2f(SignCoord[0][Sign][Destination]-5.0,SignCoord[1][Sign][Destination]-5.0);
```

```
                glVertex2f(SignCoord[0][Sign][Destination]+5.0,SignCoord[1][Sign][Destination]-5.0);
                glVertex2f(SignCoord[0][Sign][Destination]+5.0,SignCoord[1][Sign][Destination]+5.0);
                glVertex2f(SignCoord[0][Sign][Destination]-5.0,SignCoord[1][Sign][Destination]+5.0);
            glEnd();
            ArrayCoord[0][Sign][Destination]=SignCoord[0][Sign][Destination]+Array*(cos(Theta[Sign][Destination]));
            ArrayCoord[1][Sign][Destination]=SignCoord[1][Sign][Destination]+Array*(sin(Theta[Sign][Destination]));
            glDisable(GL_LINE_STIPPLE);
            glLineWidth(1.0);
            glLineStipple(1,0x00FF);
            glColor4f(1.0,1.0,1.0,1.0);
            glBegin(GL_LINES);
                glVertex2f(SignCoord[0][Sign][Destination],SignCoord[1][Sign][Destination]);
                glVertex2f(ArrayCoord[0][Sign][Destination],ArrayCoord[1][Sign][Destination]);
            glEnd();
        }
    }
    //Draw PersonHistory
    for(Destination=0;Destination<=LastDestination;Destination++)
    {
        for(Sign=0;Sign<=LastSign[Destination]-1;Sign++)
        {
            glColor4f(DestinationColour[0][Destination],
                      DestinationColour[1][Destination],
                      DestinationColour[2][Destination],1.0);
            glDisable(GL_LINE_STIPPLE);
            glLineWidth(1.0);
            glLineStipple(1,0x00FF);
            if(PersonHistory==1)
            {
                glBegin(GL_LINE_STRIP);
                    glVertex2f(SignCoord[0][Sign][Destination],SignCoord[1][Sign][Destination]);
                    glVertex2f(SignCoord[0][Sign+1][Destination],SignCoord[1][Sign+1][Destination]);
                glEnd();
            }
        }
    }
    //Functional Areas in Sign
    for(Destination=0;Destination<=LastDestination;Destination++)
    {
        for(Sign=0;Sign<=LastSign[Destination];Sign++)
        {
            int z,x=24;
            float angle;
            glBegin(GL_POLYGON);
            for(z=0;z<=x;z++)
            {
                glColor4f(0.5,0.0,0.0,0.15);
                angle=(2.0*PI*z)/(1.0*x);
                TargetCoord[0][Sign][Destination]=SignCoord[0][Sign][Destination]+R[Sign][Destination]*(cos(angle));
                TargetCoord[1][Sign][Destination]=SignCoord[1][Sign][Destination]+R[Sign][Destination]*(sin(angle));
                glVertex2f(TargetCoord[0][Sign][Destination],TargetCoord[1][Sign][Destination]);
            }
            glEnd();
        }
    }
    //Clock Time
    float TimeFactor=0.5;
    glColor4f(0.0,0.0,0.0,0.75);
    glRasterPos2f(HalfWidth+1000,HalfHeight+1000);
    MyText="ClockTime ";
    LengthOfString=strlen(MyText);
    for (MyCharacter=0;MyCharacter<LengthOfString;MyCharacter++)
    glutBitmapCharacter(GLUT_BITMAP_HELVETICA_12,MyText[MyCharacter]);
    sprintf(NumericalValue,"%i:%i",int(TimeFactor*ClockTime/60),
            int(TimeFactor*ClockTime)-60*int(TimeFactor*ClockTime/60));
    LengthOfString=strlen(NumericalValue);
    for(MyCharacter=0;MyCharacter<LengthOfString;MyCharacter++)
    glutBitmapCharacter(GLUT_BITMAP_HELVETICA_12,NumericalValue[MyCharacter]);
    //C value
    glColor4f(0.0,0.0,0.0,0.75);
    glRasterPos2f(HalfWidth+1000,HalfHeight+900);
    MyText="C (key l+s) ";
    LengthOfString=strlen(MyText);
    for (MyCharacter=0;MyCharacter<LengthOfString;MyCharacter++)
    glutBitmapCharacter(GLUT_BITMAP_HELVETICA_12,MyText[MyCharacter]);
    sprintf(NumericalValue,"%f",C);
    LengthOfString=strlen(NumericalValue);
    for(MyCharacter=0;MyCharacter<LengthOfString;MyCharacter++)
    glutBitmapCharacter(GLUT_BITMAP_HELVETICA_12,NumericalValue[MyCharacter]);
    //a Value
    glColor4f(0.0,0.0,0.0,0.75);
    glRasterPos2f(HalfWidth+1000,HalfHeight+800);
    MyText="a (key u+d) ";
    LengthOfString=strlen(MyText);
    for (MyCharacter=0;MyCharacter<LengthOfString;MyCharacter++)
    glutBitmapCharacter(GLUT_BITMAP_HELVETICA_12,MyText[MyCharacter]);
    sprintf(NumericalValue,"%f",a);
    LengthOfString=strlen(NumericalValue);
    for(MyCharacter=0;MyCharacter<LengthOfString;MyCharacter++)
    glutBitmapCharacter(GLUT_BITMAP_HELVETICA_12,NumericalValue[MyCharacter]);
    Rewrite();
    glutSwapBuffers();
}
void CalculateMotion(void)
{
    float MagnitudeFactor,distanceSq,distance,scalarproduct,
          DeltaCoord[2],ForceMagnitude,SeparationSq;
    int otherPerson;
    for(Person=0;Person<=LastPerson;Person++)
    {
        RepulsionOverMass[0][Person]=0.0;
        RepulsionOverMass[1][Person]=0.0;
    }
    for(Person=0;Person<=LastPerson-1;Person++)
    {
        if(ArrivalTime[Person]<ClockTime&&Active[Person]!=2)Active[Person]=1;
    }
```

```
//Repulsive effect
for(Person=0;Person<=LastPerson-1;Person++)
{
    if(Active[Person]==1)
    {
        for(otherPerson=Person+1;otherPerson<=LastPerson;otherPerson++)
        {
            if(Active[otherPerson]==1)
            {
                for(xyz=0;xyz<=1;xyz++)
                DeltaCoord[xyz]=Coord[0][xyz][Person]-Coord[0][xyz][otherPerson];

                SeparationSq=DeltaCoord[0]*DeltaCoord[0]+DeltaCoord[1]*DeltaCoord[1];
                ForceMagnitude=RepulsionOverMassFactor*exp(-SeparationSq/RepulsionDecayLengthSq);

                for(xyz=0;xyz<=1;xyz++)
                {
                    RepulsiveForce[xyz][     Person]+=ForceMagnitude*DeltaCoord[xyz]/sqrt(SeparationSq);
                    RepulsiveForce[xyz][otherPerson]-=ForceMagnitude*DeltaCoord[xyz]/sqrt(SeparationSq);
                }
            }
        }
    }
}
//Sign effect
for(Person=0;Person<=LastPerson;Person++)
{
    if(Active[Person]==1)
    {
        XWalking=0.0;
        YWalking=0.0;
        for(Sign=0;Sign<=LastSign[WhichDestination[Person]];Sign++)
        {
            distanceSq=(SignCoord[0][Sign][WhichDestination[Person]]-Coord[0][0][Person])*
                       (SignCoord[0][Sign][WhichDestination[Person]]-Coord[0][0][Person])+
                       (SignCoord[1][Sign][WhichDestination[Person]]-Coord[0][1][Person])*
                       (SignCoord[1][Sign][WhichDestination[Person]]-Coord[0][1][Person]);
            distance=sqrt(distanceSq);
            scalarproduct=(SignCoord[1][Sign][WhichDestination[Person]]-Coord[0][1][Person])*
                                                  cos(Theta[Sign][WhichDestination[Person]])-
                          (SignCoord[0][Sign][WhichDestination[Person]]-Coord[0][0][Person])*
                                                  sin(Theta[Sign][WhichDestination[Person]]);

            XWalking+=C*(cos(Theta[Sign][WhichDestination[Person]]))*exp(-distanceSq/aSq);
            YWalking+=C*(sin(Theta[Sign][WhichDestination[Person]]))*exp(-distanceSq/aSq);

            XWalking-=scalarproduct*(sin(Theta[Sign][WhichDestination[Person]]))*exp(-distanceSq/aSq);
            YWalking+=scalarproduct*(cos(Theta[Sign][WhichDestination[Person]]))*exp(-distanceSq/aSq);
        }
        MagnitudeFactor=sqrt(XWalking*XWalking+YWalking*YWalking)/2.0;
        if(MagnitudeFactor>1.0e-6)
        {
            XWalking=XWalking/MagnitudeFactor;
            YWalking=YWalking/MagnitudeFactor;
        }
        SignVely[0][Person]=SignMotionFactor*deltat*XWalking/mass;
        SignVely[1][Person]=SignMotionFactor*deltat*YWalking/mass;
    }
}
//Resultant force
for(Person=0;Person<=LastPerson;Person++)
{
    if(Active[Person]==1)
    {
        for(xyz=0;xyz<=1;xyz++)
        RepulsiveVely[xyz][Person]=deltat*RepulsiveForce[xyz][Person]/mass;
        for(xyz=0;xyz<=1;xyz++)
        {
            for(TimeBeforePresent=AbsoluteLastTime;TimeBeforePresent>=1;TimeBeforePresent-=1)
            {
                Coord[TimeBeforePresent][xyz][Person]=Coord[TimeBeforePresent-1][xyz][Person];
            }
           Coord[0][xyz][Person]=
           Coord[0][xyz][Person]+2.0*deltat*SignVely[xyz][Person]+deltat*RepulsiveVely[xyz][Person];
        }
    }
}
//Route choice behaviour
for(Person=0;Person<=LastPerson;Person++)
{
    if(Active[Person]==1)
    {
        for(Sign=0;Sign<=LastSign[WhichDestination[Person]];Sign++)
        {
            TargetDistance[Person]=(Coord[0][0][Person]-SignCoord[0][Sign][WhichDestination[Person]])*
                                   (Coord[0][0][Person]-SignCoord[0][Sign][WhichDestination[Person]])+
                                   (Coord[0][1][Person]-SignCoord[1][Sign][WhichDestination[Person]])*
                                   (Coord[0][1][Person]-SignCoord[1][Sign][WhichDestination[Person]]);

            if(TargetDistance[Person]<LSq)
            {
                if(Sign==LastSign[WhichDestination[Person]])
                Active[Person]=2;
            }
        }
    }
}
for(Person=0;Person<=LastPerson;Person++)
{
    if(Active[Person]==1)
    {
        PreviousCoord[0]=Coord[1][0][Person];
        PreviousCoord[1]=Coord[1][1][Person];

        Coord[1][0][Person]=Coord[0][0][Person]+Vely[0][Person];
        Coord[1][1][Person]=Coord[0][1][Person]+Vely[1][Person];
```

```
            Maud<<"0\nLINE\n8\nTrack "<<"1"<<"\n";
            Maud<<"10\n"<<PreviousCoord[0]<<"\n";
            Maud<<"20\n"<<PreviousCoord[1]<<"\n";
            Maud<<"11\n"<<Coord[0][0][Person]<<"\n";
            Maud<<"21\n"<<Coord[0][1][Person]<<"\n";
            Maud<<"62\n"<<"1"<<"\n";
        }
    }
}
void CreateData(void)
{
    InitialZoom=1.0;
    Zoom=InitialZoom;
    int i,j,m,n;
    ClockTime=0;
    SignMotionFactor=1.0;
    for(Destination=0;Destination<4;Destination++)
    {
        for(Sign=0;Sign<=LastSign[Destination];Sign++)
        {
            DestinationColour[0][Destination]=1.0;
            DestinationColour[1][Destination]=0.0;
            DestinationColour[2][Destination]=0.0;
        }
    }
    for(Destination=4;Destination<=LastDestination;Destination++)
    {
        for(Sign=0;Sign<=LastSign[Destination];Sign++)
        {
            DestinationColour[0][Destination]=0.0;
            DestinationColour[1][Destination]=0.0;
            DestinationColour[2][Destination]=1.0;
        }
    }
    //Create data for departures first group of people.
    m=5,n=50;
    Person=-1;
    for(i=0;i<=m;i++)
    {
        for(j=0;j<=n;j++)
        {
            Person++;
            Coord[0][0][Person]=-550.0+a*(1.0*i-0.5*m);
            Coord[0][1][Person]=1150.0+a*(1.0*j-0.5*n);
            ArrivalTime[Person]=int((3000*rand())/(1.0*RAND_MAX));
            Active[Person]=0;
            TimeStepFactor[Person]=1.0;
            for(xyz=0;xyz<=1;xyz++)
            {
                for(TimeBeforePresent=1;TimeBeforePresent<=AbsoluteLastTime;TimeBeforePresent++)
                {
                    Coord[TimeBeforePresent][xyz][Person]=Coord[0][xyz][Person];
                }
            }
            //For destinations 0 to 3
            WhichDestination[Person]=int((1.0*(LastDestination-2)*rand())/(1.0*RAND_MAX)+0.5);
            Colour[0][Person]=1.0;Colour[1][Person]=0.0;Colour[2][Person]=0.0;
        }
    }
    //Create data for departures second group of people.
    m=5,n=50;
    for(i=0;i<=m;i++)
    {
        for(j=0;j<=n;j++)
        {
            Person++;
            Coord[0][0][Person]=-250.0+a*(1.0*i-0.5*m);
            Coord[0][1][Person]=850.0+a*(1.0*j-0.5*n);
            ArrivalTime[Person]=int((3000*rand())/(1.0*RAND_MAX));
            Active[Person]=0;
            TimeStepFactor[Person]=1.0;
            for(xyz=0;xyz<=1;xyz++)
            {
                for(TimeBeforePresent=1;TimeBeforePresent<=AbsoluteLastTime;TimeBeforePresent++)
                {
                    Coord[TimeBeforePresent][xyz][Person]=Coord[0][xyz][Person];
                }
            }
            //For destinations 0 to 3
            WhichDestination[Person]=int((1.0*(LastDestination-2)*rand())/(1.0*RAND_MAX)+0.5);
            Colour[0][Person]=1.0;Colour[1][Person]=0.0;Colour[2][Person]=0.0;
        }
    }
    //Create data for arrivals first group of people
    m=50,n=5;
    for(i=0;i<=m;i++)
    {
        for(j=0;j<=n;j++)
        {
            Person++;
            Coord[0][0][Person]=-1250.0+a*(1.0*i-0.5*m);
            Coord[0][1][Person]=-950.0+a*(1.0*j-0.5*n);
            ArrivalTime[Person]=int((2000*rand())/(1.0*RAND_MAX));
            Active[Person]=0;
            TimeStepFactor[Person]=1.0;
            for(xyz=0;xyz<=1;xyz++)
            {
                for(TimeBeforePresent=1;TimeBeforePresent<=AbsoluteLastTime;TimeBeforePresent++)
                {
                    Coord[TimeBeforePresent][xyz][Person]=Coord[0][xyz][Person];
                }
            }
            WhichDestination[Person]=int((1.0*(LastDestination)*rand())/(1.0*RAND_MAX)+0.5);
            if(WhichDestination[Person]==0)WhichDestination[Person]=4;
            else if(WhichDestination[Person]==1)WhichDestination[Person]=5;
            if(WhichDestination[Person]==2)WhichDestination[Person]=4;
            else if(WhichDestination[Person]==3)WhichDestination[Person]=5;
```

```
            Colour[0][Person]=0.0;Colour[1][Person]=0.0;Colour[2][Person]=1.0;
        }
    }
    //Create data for arrivals second group of people
    m=5,n=50;
    for(i=0;i<=m;i++)
    {
        for(j=0;j<=n;j++)
        {
            Person++;
            Coord[0][0][Person]=500.0+a*(1.0*i-0.5*m);
            Coord[0][1][Person]=-1000.0+a*(1.0*j-0.5*n);
            ArrivalTime[Person]=int((2000*rand())/(1.0*RAND_MAX));
            Active[Person]=0;
            TimeStepFactor[Person]=1.0;
            for(xyz=0;xyz<=1;xyz++)
            {
                for(TimeBeforePresent=1;TimeBeforePresent<=AbsoluteLastTime;TimeBeforePresent++)
                {
                    Coord[TimeBeforePresent][xyz][Person]=Coord[0][xyz][Person];
                }
            }
            WhichDestination[Person]=int((1.0*(LastDestination)*rand())/(1.0*RAND_MAX)+0.5);

            if(WhichDestination[Person]==0)WhichDestination[Person]=4;
            else if(WhichDestination[Person]==1)WhichDestination[Person]=5;
            if(WhichDestination[Person]==2)WhichDestination[Person]=4;
            else if(WhichDestination[Person]==3)WhichDestination[Person]=5;
            Colour[0][Person]=0.0;Colour[1][Person]=0.0;Colour[2][Person]=1.0;
        }
    }
    LastPerson=Person;
    float Z=10.0;
    LastK=250;
    LastL=425;
    for(k=0;k<=LastK;k++)
    {
        for(l=0;l<=LastL;l++)
        {
            x[k][l]=Z*(1.0*k-0.5*LastK);
            y[k][l]=Z*(1.0*l-0.5*LastL);
        }
    }
    DrawSaveInterval=150;
    if(ClockTime==-1)FileNumber=-1;
    else FileNumber=int(ClockTime/DrawSaveInterval);
}
void WriteDXFFile(void)
{
    int r=2.0;
    cout<<"Writing dxf file\n";
    for (l_index=0;l_index<object.polygons_qty;l_index++)
    {
        Maud<<"0\n3DFACE\n8\nBoundaries "<<"5"<<"\n";
        Maud<<"10\n"<<object.vertex[ object.polygon[l_index].a ].x<<"\n";
        Maud<<"20\n"<<object.vertex[ object.polygon[l_index].a ].y<<"\n";
        Maud<<"30\n"<<object.vertex[ object.polygon[l_index].a ].z<<"\n";
        Maud<<"11\n"<<object.vertex[ object.polygon[l_index].b ].x<<"\n";
        Maud<<"21\n"<<object.vertex[ object.polygon[l_index].b ].y<<"\n";
        Maud<<"31\n"<<object.vertex[ object.polygon[l_index].b ].z<<"\n";
        Maud<<"12\n"<<object.vertex[ object.polygon[l_index].c ].x<<"\n";
        Maud<<"22\n"<<object.vertex[ object.polygon[l_index].c ].y<<"\n";
        Maud<<"32\n"<<object.vertex[ object.polygon[l_index].c ].z<<"\n";
        Maud<<"13\n"<<object.vertex[ object.polygon[l_index].a ].x<<"\n";
        Maud<<"23\n"<<object.vertex[ object.polygon[l_index].a ].y<<"\n";
        Maud<<"33\n"<<object.vertex[ object.polygon[l_index].a ].z<<"\n";
        Maud<<"62\n"<<"5"<<"\n";
    }
    for(Destination=0;Destination<=LastDestination;Destination++)
    {
        for(Sign=0;Sign<=LastSign[Destination];Sign++)
        {
            Maud<<"0\nCIRCLE\n8\nSigns"<<"5"<<"\n";
            Maud<<"10\n"<<SignCoord[0][Sign][Destination]<<"\n";
            Maud<<"20\n"<<SignCoord[1][Sign][Destination]<<"\n";
            Maud<<"40\n"<<R[Sign][Destination]<<"\n";
            Maud<<"62\n"<<"5"<<"\n";
        }
    }
    for(Person=0;Person<=LastPerson;Person++)
    {
        if(Active[Person]==1)
        {
            Maud<<"0\nCIRCLE\n8\nPerson "<<"0"<<"\n";
            Maud<<"10\n"<<Coord[0][0][Person]<<"\n";
            Maud<<"20\n"<<Coord[0][1][Person]<<"\n";
            Maud<<"40\n"<<r<<"\n";
            Maud<<"62\n"<<"0"<<"\n";
        }
    }
    Maud<<"0\nENDSEC\n0\nEOF\n";
    Maud.close();
    cout<<"Finished\n";
}
void Rewrite(void)
{
    ofstream Tom("Signs.txt");
    Tom<<LastDestination<<"\n";
    for(Destination=0;Destination<=LastDestination;Destination++)
    {
        Tom<<Destination<<"\n";
        Tom<<LastSign[Destination]<<"\n";
        for(Sign=0;Sign<=LastSign[Destination];Sign++)
        {
            Tom<<Sign<<" "<<Destination<<" "<<SignCoord[0][Sign][Destination]<<" "<<SignCoord[1][Sign][Destination]
               <<" "<<Theta[Sign][Destination]<<" "<<"\n";
        }
```

```
    }
    Tom.close();
    cout<<"Rewrite signs\n\n";
}
void SavePicture(void)
{
    char   ColourRed[2*HalfWidth];
    char ColourGreen[2*HalfWidth];
    char  ColourBlue[2*HalfWidth];
    int pixelx,pixely;
    FILE *tga;               // Pointer to a FILE
    targa_header header;     // Variable of targa_header type
    /*First, set all the fields in the header to appropriate values */
    header.id_len = 0;          /* no ID field */
    header.map_type = 0;        /* no colormap */
    header.img_type = 2;        /* trust me */
    header.map_first = 0;       /* not used */
    header.map_len = 0;         /* not used */
    header.map_entry_size = 0;  /* not used */
    header.x = 0;               /* image starts at (0,0) */
    header.y = 0;
    header.width = 2*HalfWidth;
    header.height = 2*HalfHeight;
    header.bpp = 24;            /* 24 bits per pixel */
    header.misc = 0x20;         /* scan from upper left corner */
    /* Open a file for writing targa data.  Call the file "test.tga" and
      write in binary mode (wb) so that nothing is lost as characters
      are written to the file */
    char FileName[40];
    char  zero[5]="0";
    char   one[5]="1";
    char   two[5]="2";
    char three[5]="3";
    char  four[5]="4";
    char  five[5]="5";
    char   six[5]="6";
    char seven[5]="7";
    char eight[5]="8";
    char  nine[5]="9";
    int NewFileNumber,ThisNumber,TenToThePower,Divisor,Power;
    FileNumber++;
    NewFileNumber=FileNumber;
    strcpy(FileName,"TestNo");
    for(TenToThePower=3;TenToThePower>=0;TenToThePower-=1)
    {
        Divisor=1;
        for(Power=1;Power<=TenToThePower;Power++)Divisor=Divisor*10;
        ThisNumber=NewFileNumber/Divisor;
        NewFileNumber-=ThisNumber*Divisor;
        if(ThisNumber==0)strcat(FileName,zero);
        if(ThisNumber==1)strcat(FileName,one);
        if(ThisNumber==2)strcat(FileName,two);
        if(ThisNumber==3)strcat(FileName,three);
        if(ThisNumber==4)strcat(FileName,four);
        if(ThisNumber==5)strcat(FileName,five);
        if(ThisNumber==6)strcat(FileName,six);
        if(ThisNumber==7)strcat(FileName,seven);
        if(ThisNumber==8)strcat(FileName,eight);
        if(ThisNumber==9)strcat(FileName,nine);
    }
    strcat (FileName,".tga");
    tga = fopen(FileName, "wb");//Write the header information
    writeheader(header, tga);
    for(pixely=2*HalfHeight-1;pixely>=0;pixely-=1)
    {
        glReadPixels(0,pixely,2*HalfWidth,1,GL_BLUE,  GL_UNSIGNED_BYTE,&ColourBlue);
        glReadPixels(0,pixely,2*HalfWidth,1,GL_GREEN, GL_UNSIGNED_BYTE,&ColourGreen);
        glReadPixels(0,pixely,2*HalfWidth,1,GL_RED,   GL_UNSIGNED_BYTE,&ColourRed);
        for(pixelx=0;pixelx<=2*HalfWidth-1;pixelx++)
        {
            fputc(ColourBlue[pixelx], tga);
            fputc(ColourGreen[pixelx],tga);
            fputc(ColourRed[pixelx],  tga);
        }
    }
    fclose(tga);
}
void writeheader(targa_header h, FILE *tga)
{
    fputc(h.id_len, tga);          // Write chars for ID, map, and image type
    fputc(h.map_type, tga);
    fputc(h.img_type, tga);
    fputc(h.map_first % 256, tga); // Write integer, low order byte first
    fputc(h.map_first / 256, tga); // Write second byte of integer, high order
    fputc(h.map_len % 256, tga);   // Another integer
    fputc(h.map_len / 256, tga);
    fputc(h.map_entry_size, tga);  // Write a char - only one byte
    fputc(h.x % 256, tga);         // More integers
    fputc(h.x / 256, tga);
    fputc(h.y % 256, tga);
    fputc(h.y / 256, tga);
    fputc(h.width % 256, tga);     // Even more integers
    fputc(h.width / 256, tga);
    fputc(h.height % 256, tga);
    fputc(h.height / 256, tga);
    fputc(h.bpp, tga);             // Write two chars
    fputc(h.misc, tga);
}
```