SPRINGER BRIEFS IN ARCHITECTURAL DESIGN AND TECHNOLOGY

Machi Zawidzki

Discrete Optimization in Architecture

Architectural & Urban Layout

Springer

SpringerBriefs in Architectural Design and Technology

Series editor

Thomas Schröpfer, Singapore University of Technology and Design, Singapore, Singapore

Understanding the complex relationship between design and technology is increasingly critical to the field of Architecture. The SpringerBriefs in Architectural Design and Technology series aims to provide accessible and comprehensive guides for all aspects of current architectural design relating to advances in technology including material science, material technology, structure and form, environmental strategies, building performance and energy, computer simulation and modeling, digital fabrication, and advanced building processes. The series will feature leading international experts from academia and practice who will provide in-depth knowledge on all aspects of integrating architectural design with technical and environmental building solutions towards the challenges of a better world. Provocative and inspirational, each volume in the Series aims to stimulate theoretical and creative advances and question the outcome of technical innovations as well as the far-reaching social, cultural, and environmental challenges that present themselves to architectural design today. Each brief asks why things are as they are, traces the latest trends and provides penetrating, insightful and in-depth views of current topics of architectural design. SpringerBriefs in Architectural Design and Technology provides must-have, cutting-edge content that becomes an essential reference for academics, practitioners, and students of Architecture worldwide.

More information about this series at http://www.springer.com/series/13482

Machi Zawidzki

Discrete Optimization in Architecture

Architectural & Urban Layout

Springer

Machi Zawidzki
MIT
Cambridge, MA
USA

ISSN 2199-580X ISSN 2199-5818 (electronic)
SpringerBriefs in Architectural Design and Technology
ISBN 978-981-10-1105-4 ISBN 978-981-10-1106-1 (eBook)
DOI 10.1007/978-981-10-1106-1

Library of Congress Control Number: 2016943038

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer Science+Business Media Singapore Pte Ltd.

In memoriam Professor Witold Kosński

Preface

This book concerns the layout design, which is the most fundamental problem in Architectural and Urban Composition. Part I presents the optimization of a building floor plan based on constraint satisfaction combined with coarse grid discretization, and evaluation of the quality of a town square (plaza) based on geometrical properties derived directly from its plan. Part II presents the application of crowd simulations for analytical and creative purposes in Architecture.

This book presents results of the research titled: *Effective computational methods for grid and raster-based modeling of practical problems in architectural and urban design* conducted from December 2013 to November 2015 under the Singapore University of Technology & Design and Massachusetts Institute of Technology Postdoctoral Program.

Warsaw, Poland
May 2016

Machi Zawidzki

Contents

Acronyms

ABM	Agent-based Model
AFLO	Architectural Functional Layout Optimization
AGE	Automated Geometrical Evaluation
AOF	Aggregate Objective Function
CA	Cellular Automaton
CD	Crowd Dynamics
CDC	Crowd Dynamics Component
CSP	Constraint Satisfaction Problem
CZ	Crowd-Z
DF	Distance Field
eR	Error Rate
FFCA	Floor Field Cellular Automaton
FFN	Feed Forward Network
H	Hexagonal (tessellation)
HM	Heat Map
HSE	Human Subjective Evaluation
HVAC	Heating Ventilation Air Conditioning
ME	Moore's neighborhood with Euclidean metric
MFRB	Multi-Family Residential Building
MM	Moore's neighborhood with Maximum metric
MP	Matrix of Plazas
MQ	Matrix of Qualities
NAQ	Normalized Accumulated Quality
ODF	One-directional Flow
OSU	Ordered Sequential Update
P	Plaza
QAP	Quadratic Assignment Problem
RSU	Random Shuffled Update

SPB	St Peters Basilica
SRE	Square Room Evacuation
T	Triangular (tessellation)
vN	von Neumann (neighborhood/metric)

Part I
Layout Optimization & Evaluation

The first and second chapters of this part concern with: the optimization of an architectural layout and evaluation of an urban square, respectively. The former is based on the assumption that *it is practical to impose stringent constraints for finding good functional architectural layout*, and the latter—on the assumption that *the quality of an urban space strongly depends on its fundamental geometrical properties*. Such properties can be derived directly and sufficiently from urban plans.

Chapter 1
Architectural Functional Layout Optimization in a Coarse Grid

Abstract This chapter describes the method for creating optimal architectural functional layouts. The methodology is based on coarse grid and three general steps: i. generation of layouts satisfying requirements given by the designer, ii. selection of the "proper" layouts, and iii. ranking of the "proper" layouts according to multiple objectives. Presented methodology can be used in architectural practice, urban or graphic design, and wherever the allocation of interrelated shapes is to be optimized. For clarity, simplified examples of a single-story two-apartment residential building are shown. Despite this simplicity, presented layouts resemble realistic functional solutions. One example of a practical-size floor-plan of three apartments of total twenty rooms is generated. The material is organized as follows: the concept of space discretization with coarse grid is introduced; the backtrack (depth-first) search algorithm is implemented for the generation of a number "potentially good" layouts. A machine learning method (feed-forward artificial neural network) is implemented for the classification of "proper" and "improper" layouts based on the "corridor criterion". Simple examples of dynamic multi-criterial ranking of "proper" layouts are demonstrated.

1.1 Introduction

According to Ref. [16], *Architecture* can be defined in two thousand ways. This shows the level of ambiguity and difficulty with the nomenclature and proper mathematical formulation of the problems. This is due to the expressive power of the natural language, which according to Ref. [26]: *can easily generate contradictions and paradoxes*. Moreover, the communication in this field is mostly visual, through: diagrams, plans, perspective and parallel projections, etc. Finally, an architectural design can not be formulated as simple algorithmic steps. It is rather a number of interrelated processes involving: sketching of the functional relationships, visualizations of imagined aesthetics appearance, followed by detailed drafting and approval procedures, etc. Usually, there several re-iterations and re-considerations before the project is completed, and rarely the final result satisfies all the personnel involved, which is likely due to:

M. Zawidzki, *Discrete Optimization in Architecture*, SpringerBriefs
in Architectural Design and Technology, DOI 10.1007/978-981-10-1106-1_1

1. The problems in *Architecture* are "hard".
2. The constraints, conditions, and in particular user's requirements for a project are often ill-defined.
3. The "ideal" solution is the sense of optimization has not been found. Such a solution corresponds to the point in the objective space whose coordinates have minimal (maximal) values of the objective functions. It is quite common that such "ideal" solution does not exist—then it is necessary to select one of the permissible solutions (called an "admissible solution"). Since in practice, this decision is also difficult, the final choice is usually made arbitrarily.

1.1.1 Functional Layout: Definition

A functional layout (FL) represents graphically the relationship among the parts of a building of assigned functions. The sizes and dimensions of respective spaces (e.g. rooms) must be represented with proper proportions. In other words, the elements of FL correspond to the final blueprint.

Usually the proper allocation of the functional spaces (considering their relationships and approx. sizes) lead to successful final architectural design. According to Ref. [18]: *experience has shown also that the most effective strategy for the layout problem is the iterative approach*. The architectural design is a complex process and in principle cannot be expressed as a linear procedure. However, it is composed of three roughly independent stages, as visualized in Fig. 1.1.

The problem of FL optimization has been already approached by computer science with various methods. However, according to Ref. [28]: *the resulting schemes, although correct, lacked certain organizational, aesthetic, or identifiable characteristics*. The method described in this chapter, however, produces layouts of topology and geometry resembling realistic architectural floor plans. According to Ref. [10]: *the design of FL of the spaces in the architectural design of a building and industrial facilities has received a lot of attention in the fields of Architecture and Engineering. However, research results from those fields are seldom used in the other*. There are two kinds of limitations which reduce the cardinality of the set of possible solutions:

1. *internal*, that is the user's requirements regarding the project;
2. *external*, that is the situation (lot dimensions, proportions, orientation; climatic conditions such as insolation etc.; structural constraints; daylight, safety and building code requirements, etc.).

The sizes of rooms and the relationships among them are defined by the project requirements (internal constraints). As an illustration let us consider an apartment comprised of: i. master-bedroom, ii. bedroom, iii. toilet (WC), iv. bathroom,

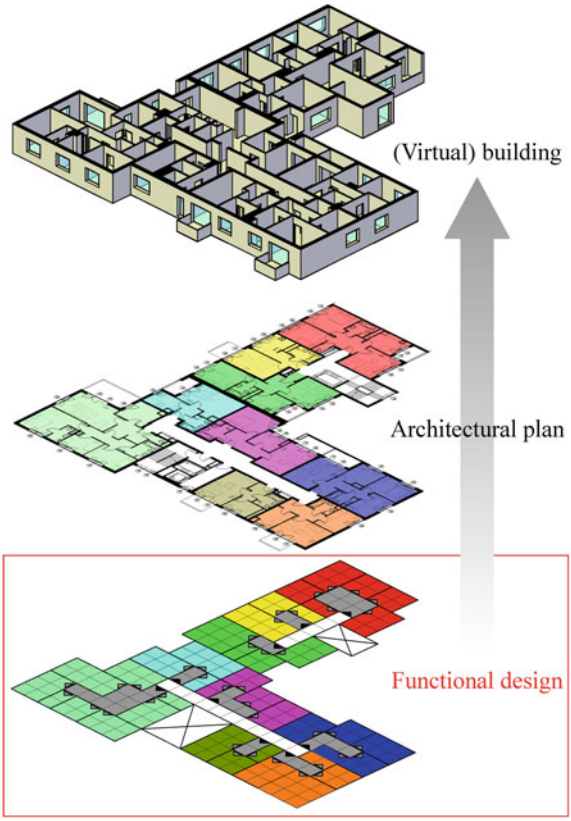

Fig. 1.1 Three stages of an architectural design shown schematically. This chapter is concerned with the functional design (highlighted in *red* on the *bottom*)

v. kitchen, and vi. living-room. Moreover, the following constraints and relationships are defined:

1. None of the bedrooms to face North;
2. Living-room to face South;
3. Living-room and kitchen to be adjacent;
4. Bathroom and master-bedroom to be adjacent.

The conditions listed above extend adjacency diagrams traditionally applied in architectural design [21] by introduction of orientation and negative edges (forbidden adjacencies). An allowable solution must meet these requirements (constraints). The external limitations mentioned above, must not be violated. They have been implemented in the search algorithm as *pruning functions*.

The framework presented here can be used in various "layout" problems. The rest of this chapter, however, focuses on single-story residential buildings (RBs).

The cases where the geometry of the layout is given and the *arrangement* of the facilities is subjected to optimization only, are known in the literature as "facility

layout optimization". Several heuristic methods have been successfully implemented for this type of optimization, e.g.:

- Tabu search strategies (incl. multi-search tabu search strategy) [17];
- "Annealed Neural Network" [31];
- Simulated Annealing [24];
- Genetic Algorithms [27];

Nevertheless, such problems are in fact rare in architectural practice. Usually the objective of the architectural layout optimization is the establishment of the geometry of the layout subjected to several constraints.

1.2 Coarse Grid Discretization

In theory, rooms in RBs can have any sizes and proportions [20]. However, in fact almost always the proportions and sizes of such rooms lay within surprisingly narrow ranges. E.g. a typical room is not elongated more than two squares. Such expert knowledge can substantially limit the potential search space by exclusion of vast amounts of impractical layouts.

Grid and guide systems have been implemented in various kinds of design: graphic, engineering, urban, architectural, etc. In FL optimization, e.g. in quadratic assignment problem, the geometry of a building is simplified in a coarse grid. The relationships among space elements are defined and the space units are assigned to the corresponding locations. Such relationships include both geometry and topology. This distinguishes so called "space layout planning" from the classic "linear assignment problem" [14]. In the architectural FL problem, however, the shapes of the rooms (spaces) and the building lot are constrained and defined beforehand. Moreover, the building outline is often not predefined, and in fact usually becomes one of the major objectives in the architectural design. The analysis of several architectural layouts, in particular of residential multi-unit constructions lead to the following four fundamental assumptions:

Functional layouts (FL) can be solved in discrete space.

For all analyzed floor-plans of RBs, it was possible to schematically transcribe the layouts into square, very coarse grids. Such operations conserved exactly the topological relationships among respective spaces. They also conserved approximately the proportions and sizes of the rooms. In other words, the architectural elements defined in the fine-grid (1×1 cm) commonly used in architectural plans can be transposed to qualitatively coarser functional grids (range of 1×1 m). Figures 1.2, 1.3, 1.4 and 1.5 show a few examples of floor-plans of multifamily residential buildings (MFRBs) from Warsaw, Poland.

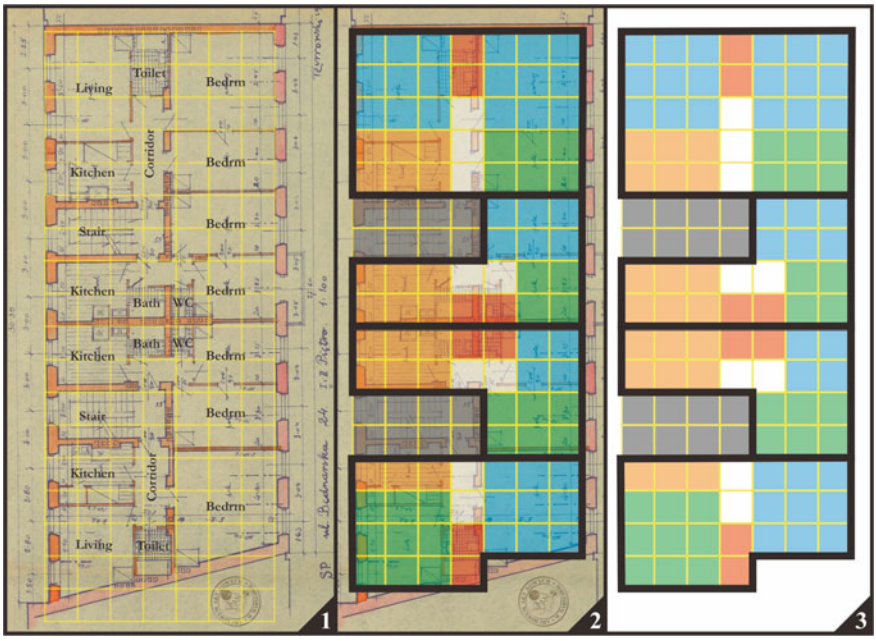

Fig. 1.2 A typical floor of a classical MFRB. **1** The original plan with superimposed coarse grid. **2** The functional layout superimposed over the original plan. **3** Pure functional layout in the coarse grid. Grid size: 1.73 m

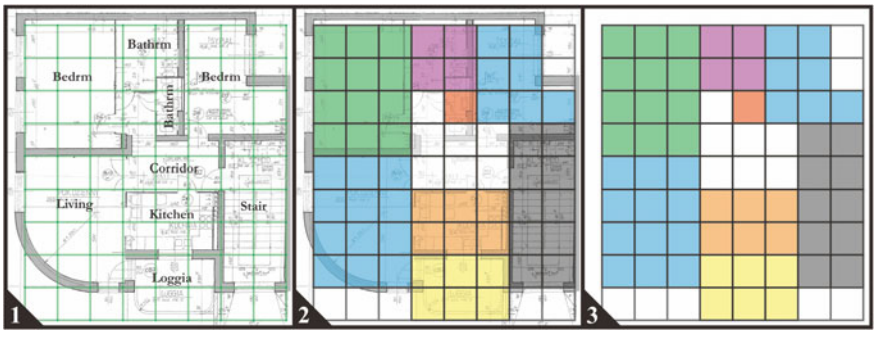

Fig. 1.3 A single apartment in a typical middle-income MFRB. Convention as in Fig. 1.2, grid size: 1.48 m

The simplified graphical representation on the right-hand side of Figs. 1.2, 1.3, 1.4 and 1.5 are FLs as defined above. Final architectural plans are far more complex, as they are subjected to various additional requirements and constraints such as:

- Provision of services such as HVAC, plumbing, etc.
- Placement of load-bearing members.
- Maximal spans of the beams and slabs given by the construction system, etc.

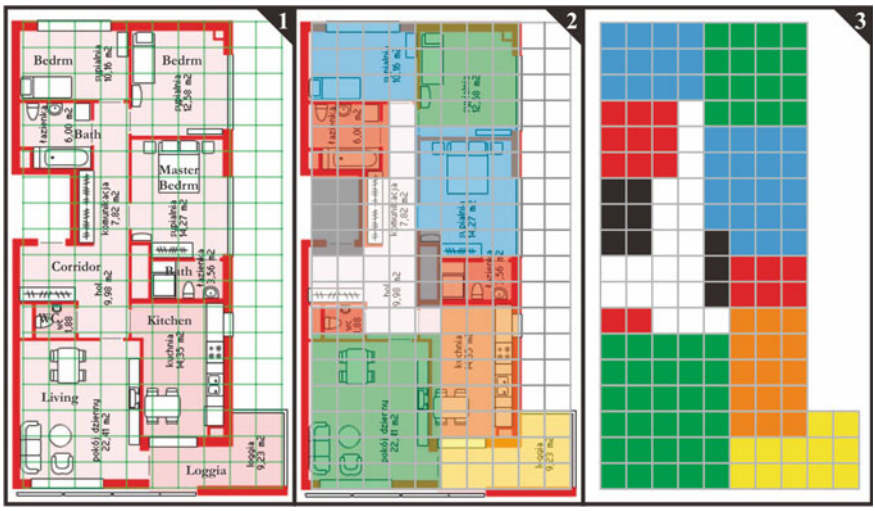

Fig. 1.4 A single apartment in a typical upper-income high-rise MFRB. Convention as in Fig. 1.2, grid size: 0.93 m

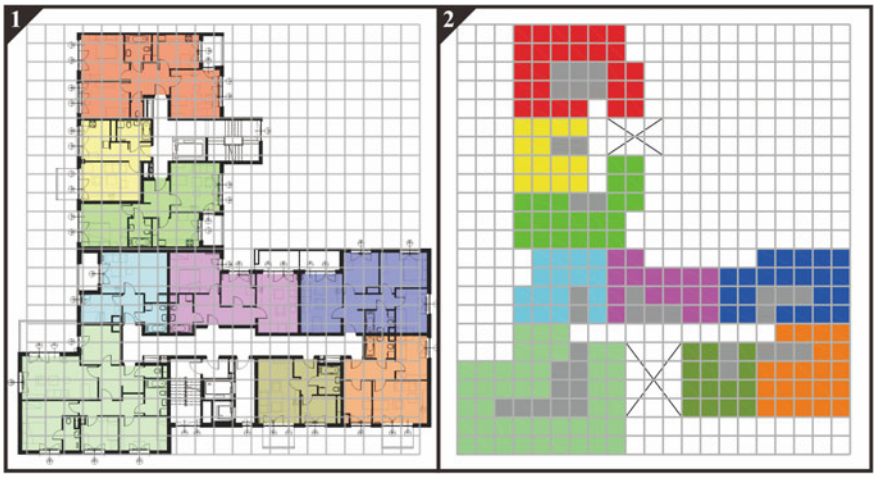

Fig. 1.5 A typical floor of a multi-story upper-income multi-unit residential building; grid size: 1.68 m

For a given architectural FL, the type of coarse grid, proportions and sizes of the spaces must be individually determined to meet to the user's requirements and reflect expert knowledge regarding particular type of a building, building regulations, etc.

Any functional layout (FL) can be transformed into an architectural floor-plan.

In general, architectural floor-plans can be derived from FLs. It is possible, however, that some FLs can not be directly transformed to architectural blueprints due to structural constraints and other construction or service conflicts. In such cases the data about initial constraints must be adjusted and the algorithm—restarted. After the completion of the computations, the solution(s) produced in a coarse square grid need to be interpreted and adapted by a designer to become the final architectural floor-plan (see the middle sub-figure of Fig. 1.1).

For a given type of functional layout (FL), the number of size variations of the given rooms is small.

As mentioned before, the actual architectural blueprints are based on fine, usually 1×1 cm grid. In such grids, the room sizes greatly vary. On the other hand, the coarse functional grid is based on arbitrary units, usually close to the minimum corridor clearance (in the examples from Figs. 1.2, 1.3, 1.4 and 1.5, from 0.73 to 1.73 m). As these Figures indicate, the number of possible room proportions and sizes considerably decreases.

Combinatorial search is feasible in functional design space.

In other words, by provision of sufficient initial data for the rooms (that is required sizes and relationships), and by application of proper coarse grid system, the cardinality of the set of allowable solutions decreases considerably, so the application of exhaustive combinatorial search methods is possible.

In 1963 Buffa and Armour have formulated the assignment of facilities to given shapes in a grid as a "quadratic assignment problem" (QAP). They created a computer program: Computerized Relative Allocation of Facilities Technique (CRAFT, for short) [4]. Subsequently, heuristic methods such as genetic algorithms have been successfully implemented to QAP [14, 18]. Nevertheless, in principle the problems of space planning are substantially more complex than QAP due to the imposition of activity area requirements. According to Ref. [18]: *Since areas required by activities are not necessarily equal, it is not always feasible to match activities and locations on a one-to-one basis*. The approach described in this chapter differs from the classic QAP: i. the outline of the building is unknown, ii. each room is rectangular, iii. the rooms have different predefined sizes. For such a problem, the implementation of genetic algorithm for seems extremely difficult. However, a successful implementation of a genetic algorithm for optimization of the topology of an architectural layout

has been shown in [29]. The problem was defined there as a search under constraints for the best adjacencies between functional spaces among many possible types of spaces.

1.2.1 Tatami-Mat System

At first sight similar, however, principally different compositional system based on **tiling** with rectangular units, called *tatami* has been developed in historical Japan. *Tatami* is a type of mat used as a flooring material in traditional Japanese-style rooms. Originally, *tatamis* were luxury items for the nobles. During the Heian period, that is from 794 to 1185, the palatial rooms were mainly wooden, and *tatamis* were used as seating for the aristocrats of the highest rank, only. In the Kamakura period, that is from 1185 to 1333, *tatamis* were also used by the priests and samurai. In the Muromachi period, that is from approximately 1336–1573, *tatamis* gradually came to be spread over entire floor of the rooms. At first, the floors of small rooms were entirely covered with *tatamis*. Rooms with floors completely tiled with *tatamis* are called *zashiki*, which means "the room spread out for sitting". Gradually, *Tatamis* were popularized and by the end of the seventeenth century reached the homes of commoners.

The size and dimensions of *tatami* is clearly related to human body, in particular in the sleeping position. *Tatami* has proportions of approximately two squares, so called one *ken* × half-*ken*. The length of *ken* differs between different regions in Japan: from 0.955 m in Kyoto area to 0.88 m in Tokyo area. Despite the fact that a traditional tatami-mat layout can be placed on an approximately 1×1 m square grid, it has not been a compositional grid system in the Western sense (compare with Figs. 1.2, 1.3, 1.4 and 1.5). Figure 1.6 shows an example of a traditional Japanese-style floor-plan and its "gridization".

In certain arrangements a half-*ken* × half-*ken* (square) mats were used. However, these were exceptions used when a regular *tatami* alone could not tile a room rectangle, and layouts with more than one such irregular mat are rare in the history of Japanese architecture. On the other hand, in recent years "square tatami" tiling (called *Ryukyu Datami*, which means "tatami of Okinawa") became relatively popular in some regions of Japan. However, they lost all the connection with historical rectangular "sleeping mat" origin and even their size is customizable.

Most importantly, traditional *tatami* is a physical flooring tile, not a virtual grid defining the placement of structural elements. As a result the actual dimensioning of the load-bearing members such as walls and columns is secondary to the *tatami* tiling.

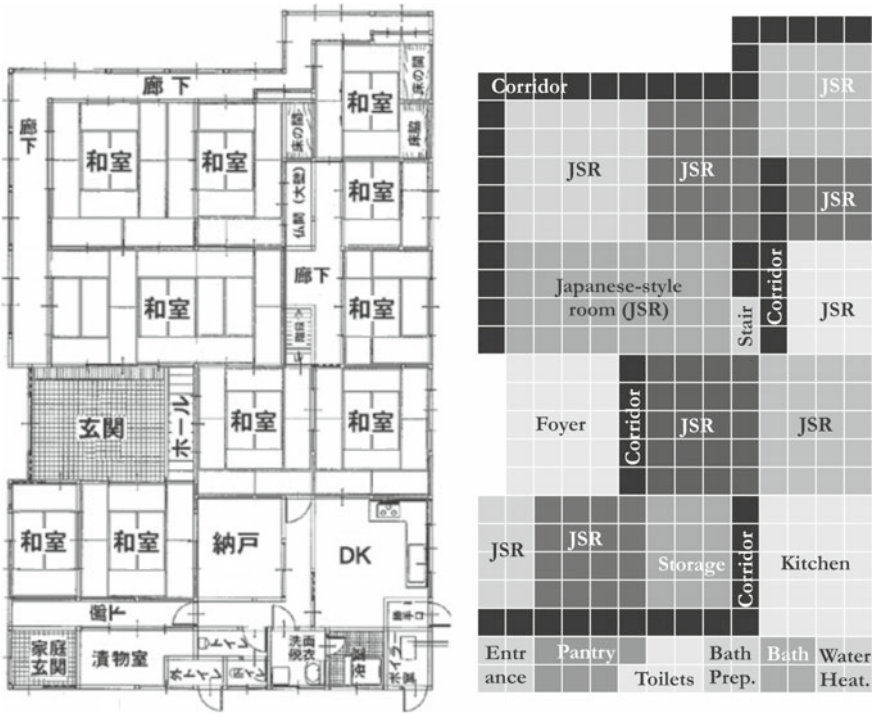

Fig. 1.6 *Left* A floor-plan of a Japanese traditional guest-house. *Right* the same layout transformed into a square grid. JSR stands for "Japanese-style room"

1.3 The Procedure

The room configurations are ranked according to the given criteria. It is a multi-step process:

1. The input of the initial information including: i. the geometrical constraints, in other words sizes of the apartments; ii. the required geometrical properties of the rooms, that is the number, proportions, sizes, number; iii. daylight requirements; iv. the relationships among rooms e.g.: adjacencies, minimal and maximal allowed distances among the given rooms, etc.
2. Generation of the potential solutions. A depth-first backtracking search algorithm is applied for this Constraint Satisfaction Problem (CSP). During this search, only the complete configurations are collected. In other words, each room configuration must not violate any constraint and must include all the required rooms. Although the searching procedure is constrained significantly, the cardinality of the produced set of configurations may range enormously: sometimes returning enormous number of solutions, sometimes—none. Thus this algorithm is very sensitive to the initial input. The procedure can be halted after collection of any

arbitrary number of potential solutions. However, in principle in order to find the *ideal solution* a full search should be performed. According to Ref. [31], instead of getting a single mathematically optimal layout, it is desirable from the designer's perspective to have a choice among a number of solutions.

3. Selecting the *proper configurations* from the set of *potential solutions*. All the initially generated layouts, that is the *potential solutions* meet the criteria and constraints. However, not all of them are *proper* in the architectural sense. In this chapter the internal (corridor) communication is used for the classification criterion. Such classification can be performed "manually" by a designer, however, here a classic machine learning method has been applied.
4. Final ranking of *proper functional layouts*. At this stage additional evaluation criteria are applied to sort the results. E.g.: the size of the internal (corridor) communication, minimal perimeter of the entire floor-plan, the position of a selected room in a given zone (for instance at South elevation). The input for the algorithm generating the initial room configurations can include some of these criteria. However, it is more natural in design practice to consider greater set of allowable solutions and filter-out the best ones according to additional (often variable) criteria. Moreover, other types of expert knowledge can be applied for such ranking at this stage. As mentioned above, it is possible that no solutions jointly meet the initial and final requirements. In such a case, the initial constraints should be relaxed and the process—reiterated.

1.3.1 Creativity and Knowledge

Fuzzy terms are the most natural for expression of the knowledge regarding the requirements for a design. According to Ref. [15] it is possible to perform algebraic operations on fuzzy numbers after so called *defuzzyfication*. Nevertheless, quantification of the knowledge by assignment of numerical values to the levels of satisfaction is still and most likely will remain among the most difficult challenges. The translation of the expert knowledge into the programming language of the algorithm is a stand-alone problem, which is not covered by this chapter. Nevertheless, described here formulation of architectural layout optimization as a discrete optimization problem requires indeed the architectural expert knowledge. E.g. the knowledge on RBs embedded in the algorithm included: setting the proper coarse grid, setting the sizes and proportions of respective rooms in the grid. Furthermore, the architectural expert knowledge is also reflected in the algorithm of positioning rooms, in particular the knowledge on the daylight and internal communication requirements (e.g. from the building code). This algorithm can be further adjusted to include other types of knowledge expressed as geometrical constraints regarding for example: particular structural system, daylighting, passive solar gains, energy conservation, security, plumbing, HVAC, etc. The procedure presented here can be used for various types of layouts. However, the algorithm must always be adapted for particular building type, e.g.: RB, office and commercial buildings, sports facility, swimming pool, etc.

It is important, however, to be aware that designing is an creative act, which remains and most likely will remain a special human activity [23]. Moreover, according to Ref. [7] designing is also a pleasurable activity. It is possible that certain qualities of particular room configurations can not be fully expressed numerically. It is also possible that substantially different layouts will have the same rating according to provided criteria. Nevertheless, despite these limitations the framework described in this chapter is a practical design support tool. It provides designers with a systematical methodology which allows to explicitly address and evaluate designer's constraints and preferences. Finally, the intension is not to replace the human element of the architectural design, but to provide the most efficient and user-friendly design support environment.

1.3.2 The Initial Input

The user provides the initial input which contains the following information: number of apartments, number of rooms, sizes of the apartments and rooms. Individual rooms are encoded as two-level lists. The first sub-list is the general and permanent information about the room, that is: index of the apartment, index of the room, the daylight requirement (yes or no), and the index of the room to which the given room must be adjacent. The second sub-list contains the dimensions of given room. This data is semi-permanent as the dimensions can be flipped due to the rotation of a room.

$$\{\{a_i, r_i, r_i*, d_i\}, \{w_i, l_i\}\} \tag{1.1}$$

where, $a_i = 1, ..., N_A$;
$r_i = 1, ..., N_{Ri}$;
$r_i* = 1, ..., N_r - 1$;
$d_i \in (0, 1)$;
w_i—the width of an ith room (the x dimension);
l_i—the length of an ith room (the y dimension);
N_{Ri}—the number of rooms in the ith apartment;
N_A—the total number of apartments;

As mentioned above, the values in the second sub-list can be flipped dues to the room rotation: $\{l_i, w_i\} \rightarrow \{w_i, l_i\}$. A list of such two-level lists unequivocally encodes the layout of an apartment, and the list of such lists encodes the entire building, as shown in Fig. 1.7.

Fig. 1.7 The initial data for a simple building layout with two apartments, shown in *red* and *green*, respectively. Apartment 1 contains rooms number: 1 and 2, apartment 2 contains rooms number: 3 and 4. The room sizes for apartment 1 and 2 are highlighted in *red* and *green*, respectively. The adjacencies are indicated by *cyan arrows*

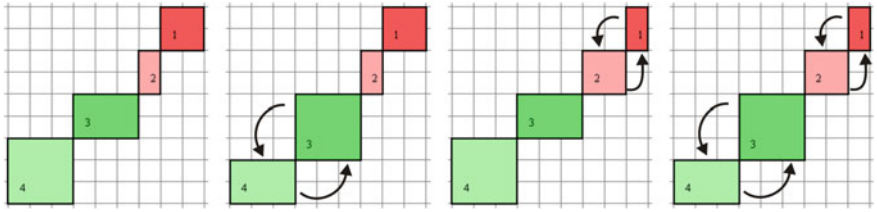

Fig. 1.8 All four permutations of the room sequences of the layout from (Fig. 1.7)

1.3.3 Pre-processing of the Initial Input Data: Room Permutations

The algorithm generating the initial room configurations operates on fixed room lists provided by the user. In the first step, the algorithm permutes the room sequence in each apartment as shown in Fig. 1.8.

The number of room permutations within apartments is expressed as follows:

$$\frac{N_A!}{2}\prod_{i=1}^{N_A} N_{Ri}! \tag{1.2}$$

where, N_A is the number of apartments, $N_A > 1$;
N_{Ri} is the number of rooms in the ith apartment.

Moreover, each non-square room can be placed at two orientations.

1.3.4 *Potential Solutions: Generation of Room Configurations as a Constraints Satisfaction Problem (CSP)*

The placement of rooms is limited to allowable positions. Therefore the creation of a FL can be considered as a CSP. A number of computer programs based on this approach have been implemented for a layout optimization defined as a rectangle partitioning: WRIGHT [2], SEED [12], HeGeL [9], LOOS/ABLOOS [11].

The styles of RB vary substantially worldwide. However, it can be said, that regardless of local building traditions and conditions, room locations are always somewhat limited. Proper mathematical formulations of such constraints are usually the most challenging tasks for the following reasons:

1. Transcription of the architectural knowledge and client's requirements expressed in natural language into mathematical formulae is a difficult communication challenge.
2. The algorithm is sensitive to the initial constraints. Thus setting overly relaxed constraints results in enormous sets of potential solutions. Conversely, overly stringent constraints produce no results.
3. Moreover, setting highly stringent initial constraints causes omission of potentially very interesting and exceptional solutions which could not have been preconceived by a designer.
4. To a certain degree some constraints are contradictory.

For most RBs, substantial part of the architectural knowledge and principles can be formulated as a set of geometrical constraints. Below is the list of rules implemented in the algorithm presented here:

- The size of the building is constrained and required as input.
- The sizes of apartments are constrained and required as input.
- The sizes of rooms are required as input.
- Rooms are rectangular. This seems as a major limitation for the architectural creativity. However, in fact it applies to the vast majority of the existing building stock. An approach for layout problems not limited to orthogonal rooms has been investigated in [3, 8].
- Main rooms, that it rooms which require daylight constitute *chains*. It means that each main room is adjacent to other main room, etc. In principle, voids in the chains, that is gaps between any rooms are not allowed.
- *Main* rooms such as: bedrooms, living room, kitchen, study rooms must be located on the building perimeter, as they require daylight. Thus buildings with atria are considered as having multiple perimeters.
- Conversely, *auxiliary* rooms such as toilets, closets, bathrooms, pantries, etc. do not require daylight, thus they can be located also "deep inside" the building.
- Two corner-to-corner rooms are not considered adjacent. In other words, two adjacent rooms share wall of at least one-grid length.

- The distance (e.g. minimal and maximal) between any pair of rooms can be constrained and given as input.
- The corridors are not predefined since their shape is usually not important. Nevertheless, they are crucial for the layout but usually should be as small as possible.

Further requirements and constraints to be implemented in the algorithm are provided by the user. E.g.: the lot size and shape; number and sizes of apartments; number, sizes and types of rooms; preferred relative and locations of rooms; preferred orientations of rooms, etc. Most of these geometrical requirements are translated to as logic statements defining the *pruning functions*. These functions control the allowable positions of the rooms during the generation of potential solutions. Fuzzy logic is considered the most natural for expression of architectural, that is spatial relationships [5]. However, since the potential size of the search space is enormous the bivalent logic statements have been implemented in the algorithm. Finally, most of architectural knowledge to be implemented to the algorithm is specific to the type of the optimized building.

1.3.5 Depth-First Search by Backtracking

The cardinality of the set of all possible layouts grows extremely fast with the growing number of rooms and their sizes. The number of all possible room positions in a coarse square grid for a given sequence (chain) of rooms (see Fig. 1.7) can be computed according to the following formula:

$$2^{k-1} \prod_{j=1}^{k-1} (x_j + x_{j+1} + y_j + y_{j+1}) \tag{1.3}$$

where, k is the total number of rooms;
j is the room index;
x_j is the x dimension of the jth room;
y_j is the y dimension of the jth room.

Although in principle, corner-to-corner neighborhood and room overlapping are not allowed in room layouts, they are also included in the formula above. Moreover, as mentioned before, each non-square room can be placed at two alternative orientations (see Fig. 1.9).

Thus the number of all possible room configurations for a given *chain* must be multiplied by 2^r, where r is the total number of non-square rooms in the list. The final formula for calculating all possible room configurations:

$$2^r \times \frac{N_A!}{2} \times \prod_{i=1}^{N_A} N_{Ri}! \times 2^{k-1} \prod_{j=1}^{k-1} (x_j + x_{j+1} + y_j + y_{j+1}) \tag{1.4}$$

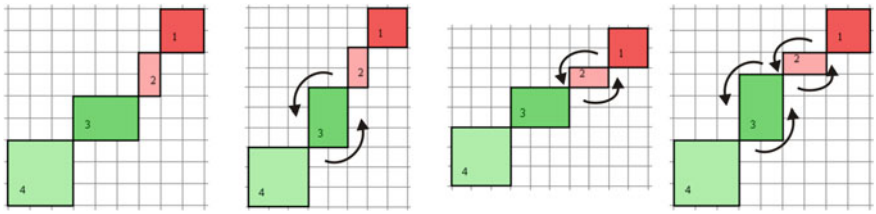

Fig. 1.9 Non-square rooms have two alternative orientations

where, r is the total number of non-square rooms;
N_A is the number of apartments, $N_A > 1$;
i is the apartment index;
N_{Ri} is the number of rooms in the ith apartment;
k is the total number of rooms;
j is the room index;
x_j is the x dimension of the jth room;
y_j is the y dimension of the jth room.

Therefore, in this very simple four-room layout the number of all possible room configurations is: $\frac{2!}{2} \times 2! \times 2! \times 2^3 \times (2+1+2+2) \times (1+3+2+2) \times (3+3+2+3) \times 2^2 = 78848$.

As the numbers above indicate, the use of so called "brute force" search methods here is impractical. As Fig. 1.10 indicates, the width of the search tree is much broader than the depth. Moreover, only complete room chains, that is configurations containing all the required rooms are valid. For these reasons a classic *depth-first* search method, so called backtracking has been implemented. In this procedure the subsequent rooms from the initial input list are being added until all of them are successfully allocated in the coarse grid.

Backtracking method has already been applied for rectangle partitioning in tackling so called Unequal Area Layout Problem in programs: WRIGHT [2], and HeGel [1]. In this chapter the backtracking is implemented as follows: if a pth room can not be added without violating any constraints or due to pruning functions, such branch of the search tree is omitted, the location of previous room $(p-1)$ is modified and process further continues. The data for the algorithm is given in the form illustrated in Fig. 1.7. Examples of additional constraints implemented to the algorithm are presented and explained in Table 1.1.

The algorithm generates the potential solutions as lists which contain geometrical information (position and orientation) for all rooms. Each room is encoded uniquely by a list containing three sub-lists:

1. The first sub-list is copied from the original input data and contains the general description of the room: the index of the apartment, the index of the room, the index of the room to which this room is adjacent, the requirement for daylight.

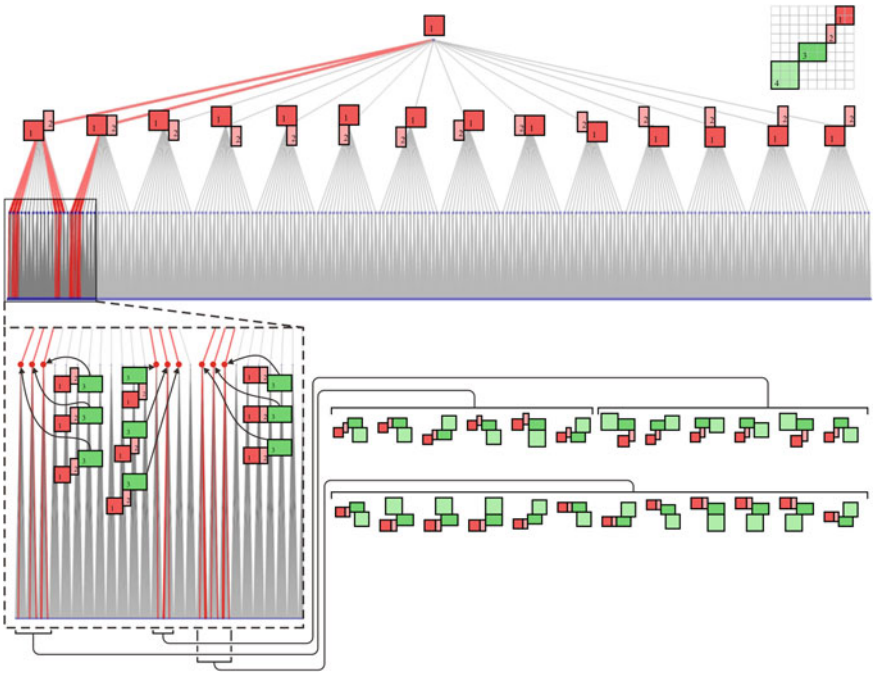

Fig. 1.10 The tree-graph illustrating complete layout search for the room chain shown in the *right top corner* of the Figure. The tree has 4928 *leaves*. The paths leading to the twenty-four potential solutions are highlighted in *red*. The corresponding room configurations are shown for each path. The relevant part of the tree is magnified on the *bottom left*. This method is relatively efficient, as the majority of the nodes (which lead to infeasible solutions) is omitted. This tree also illustrates the dependence of this method on the constraints: the allowable configurations are localized and scarce

Table 1.1 The table with complete initial data defining the simple building layout of two-room apartments

A sublist of the initial input list	Explanation
{{1,1,X,1}, {3,3}}, {{1,2,1,1}, {3,2}}, {{2,3,2,1}, {1,2}}, {{2,4,3,1}, {2,2}}}	The principal geometrical data regarding the rooms. This information also includes the access to daylight and relationships to other rooms
{7,6}	The size and proportions of the lot: 7 × 6 units
{{1,{6,5}}, {2,{3,3}}}	The sizes of the apartments: Apartment #1: 6 × 5 units Apartment #2: 3 × 3 units
{{3,1,1,2}, {4,1,1,2}, {4,2,1,3}}	Three additional constraints for relative room positions. The middle example: the minimal and maximal allowable distances of room #4 from room #1 are 1 and 2 units, respectively

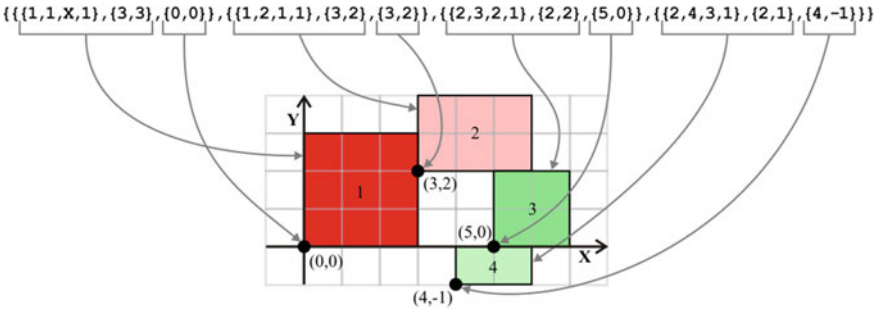

Fig. 1.11 A potential (allowable) solution for the simple two-room apartments layout

Fig. 1.12 The algorithm produced many poor layouts due to uncontrolled relative distances among rooms

2. The second sub-list provides the updated (due to possible rotation) room dimensions.
3. The third sub-list contains the position of the bottom left corner of the room on the grid (see Fig. 1.11).

A number of additional parameters e.g.: the plot and apartment sizes and shapes can further efficiently reduce the search domain. Tuning such parameters requires certain skills and trial-and-error experimentation. Figure 1.12 shows a number of solutions generated at constrained: i. size of the entire floor-plan, ii. sizes of the apartments. The total number of solutions is very large and the quality of layouts are poor due to the room clustering.

Proper communication network within a functional layout means that each apartment has own sub-corridor allowing for access to each room. Moreover, there must be a main corridor which connects the sub-corridors and can be accessed from the exterior. A communication network is inefficient in one of the following conditions:

- Any room is inaccessible from a sub-corridor.
- The communication network is overly large and complex.

Adding more reasonable constraints improves the performance of the algorithm. E.g. Fig. 1.13 shows some examples generated at the following additional constraints: i. there must be at least one grid-cell clearance between rooms #1 and #2 and rooms #3 and #4

Under these constraints, the algorithm has returned 695 solutions. For the computational efficiency it is crucial that the solutions are unique, in other words that the algorithm does not generate configurations which are redundant. Here, also any pair of solutions which differ only by rotation by π are not considered unique. Figure 1.14 shows selected examples.

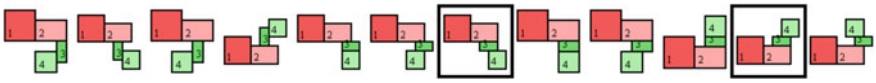

Fig. 1.13 Improved layouts as a result of additional geometrical constraints. Room configurations framed in *black* are in fact classified as "proper" according to so called "corridor criterion"

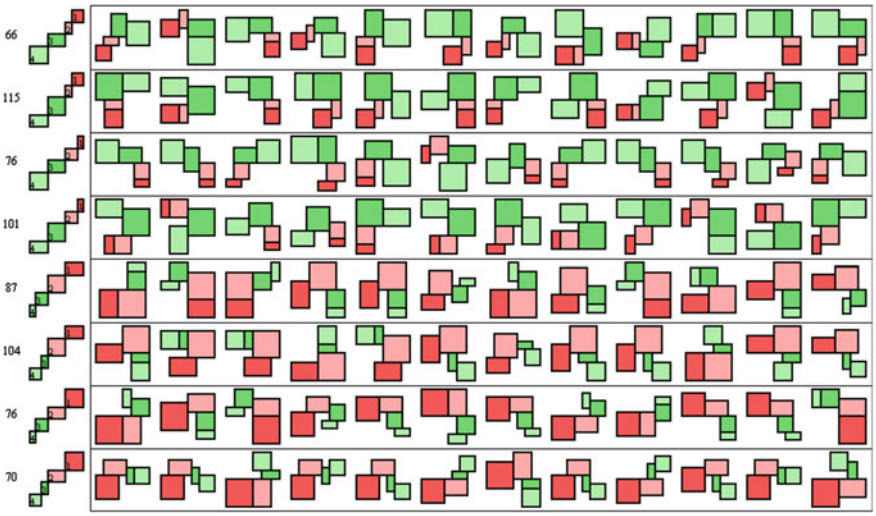

Fig. 1.14 Selected potential solutions generated under increased number of constraints. For each *chain* permutation the number of potential solutions is shown on the *left*

35 % that is 244 out of 695 of the potential solutions are proper. The classification methodology is described below.

1.3.6 Classification of Potential Solutions According to the Internal Communication Criterion

The network of corridors embodies the internal communication in FLs. Corridors, unlike the rooms do not have predefined sizes and shapes. In a nutshell the function of the corridor network is to allow access to each room, as shown in Fig. 1.15. In a case where a certain corridor or its part must have predefined size or shape it can be included as an additional "non-daylight-requiring" room with defined relationship to other rooms in the *chain*.

The classification of potential solutions to two disjunctive classes of "proper" and "improper" is relatively complex and difficult to express as a sequence of simple algorithmic steps. It takes a while even for a trained user. The requirements used here for the classification:

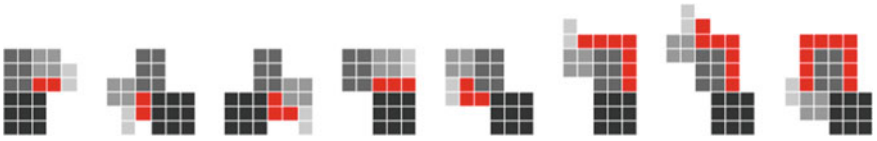

Fig. 1.15 Examples of simple apartment layout comprised of four rooms. The corridors are highlighted in *red*. The size and complexity of corridors grow from *left* to *right*

Fig. 1.16 18 potential solutions. Only one (indicated by a frame) is proper according to the criteria defined above. *Magenta* and *cyan* indicate the sub-corridors for the "red" and "green" apartments, respectively. *Yellow* indicates the main corridor which connects both sub-corridors. The total number of grid-cells occupied by the communication is four

- There is a sub-corridor in each apartment.
- The main corridor connects the apartments' sub-corridors.
- The total number of corridor grid-cells must not exceed four.
- Sub-corridors should be proportional to the apartments' sizes.
- The main corridor, if possible, should be larger than the apartments' sub-corridors.

Figure 1.16 shows a proper configuration among improper room configurations.

Such classification is relatively simple for small examples, and can be performed simply by a visual inspection. Solving such problems is rather natural to humans, as it requires simultaneous evaluation of many pieces of graphical information. Nevertheless, in realistic cases with dozens of rooms and thousands of potential solutions such task becomes very difficult, if realistic at all. On the other hand, as mentioned before the straightforward automation seems impossible. For these reasons a classic machine learning method based on artificial neural network has been applied.

1.3.7 Classification of Rooms Configurations with Feed-Forward Neural Network

The number of potential solutions in the above example is relatively modest (695) and the classification could be done "manually". However, for illustrative purposes a machine learning method has been used. The Feed-forward Neural Network (FNN) has been selected to these reasons:

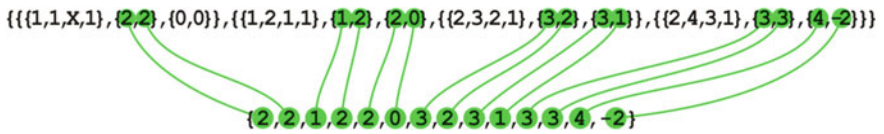

Fig. 1.17 On the *top* the original encoding. On the *bottom* the "compressed" 14-dimensional vector for FNN

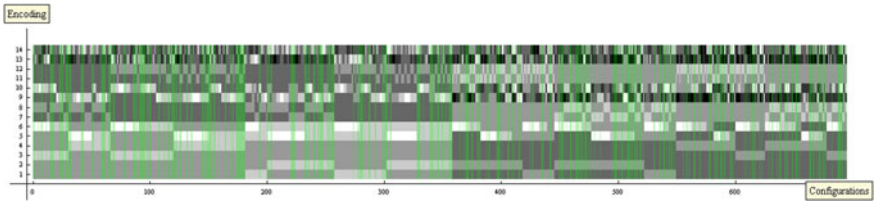

Fig. 1.18 695 potential solutions (room configurations) encoded as 14-dimensional vectors. The *gray* corresponds to the value of vectors' coordinates: *white* and *black* for minimal and maximal values, respectively. *Green* indicates the "proper" room configurations

- FNN is a classic supervised machine learning method.
- It is commonly applied for various classification problems.
- The criteria for evaluation are difficult to express in strict mathematical formulae.
- However, in the problem considered it is possible to unequivocally determine whether any configuration is improper or proper.
- It is also possible in this example to provide enough samples for the training data. In this example more than $15 \times 15 = 225$ training vectors are needed; 621 have been provided.
- Data represents all classes rather equally.
- The training samples are similar.

For efficiency, the original encoding has been simplified. Any redundant data has been removed, e.g.: apartment and room indices as shown in Fig. 1.17.

Figure 1.18 visualizes all 695 potential solutions which have been encoded as 14-dimensional vectors. The first 621 served as the training set, the remaining 74—as the validation data for FNN.

FNN with the following parameters has been initialized: 14-input, 1-output, 1 hidden layer, 14-neurons, and Sigmoid activation function. Training has been performed with Levenberg-Marquardt algorithm. The initial weights have been generated randomly. Two alternative terminating conditions have applied: maximum number of iterations has been set to 1000 and "stop if five consecutive iterations of λ produce no change"; where λ is the width of the basis function for each neuron described by a one-component vector. Figure 1.19 illustrates with the by root mean square deviation ten trials of the training process of this FNN. Figure 1.20 compares the performance of FNN at ten training trials. A threshold function has been applied. As a result over 89 % of the validation samples have been correctly classified by the best FNN (trained in the 1st trial). Figure 1.21 shows the details of the validation.

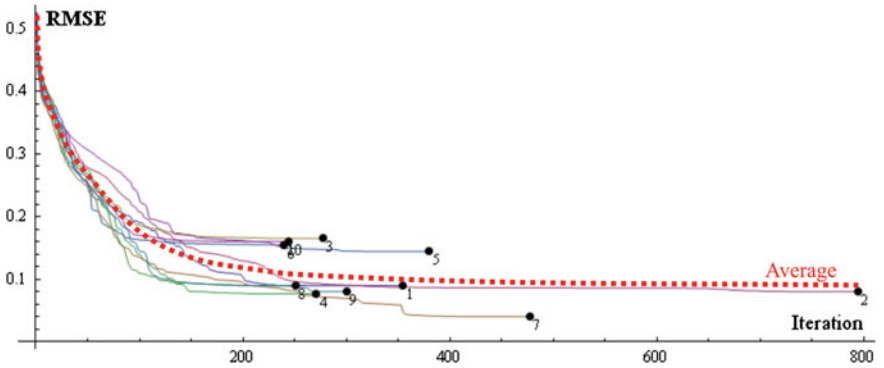

Fig. 1.19 Ten trials of the FNN training. The number of iterations completed in each training session varies and depends on the progress of training. The *red dotted line* shows the mean of the ten training *curves*. The trial indices are shown at the corresponding *curves*

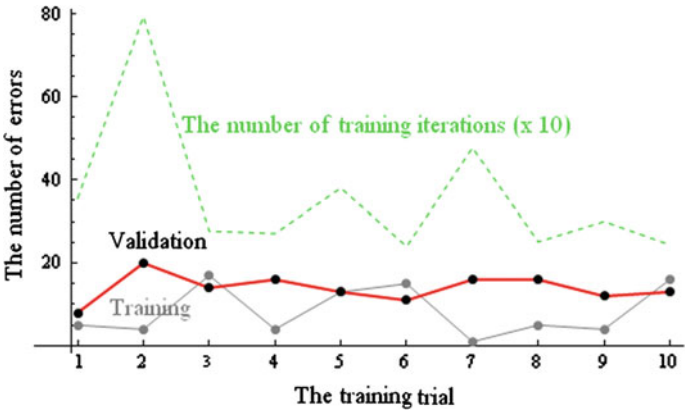

Fig. 1.20 The response errors for all ten trained FNNs. The training errors and validation errors are shown in *gray* and *red*, respectively. The performance for the validation data of the first-trial FNN was the best (number of samples: 74, number of errors: 8)

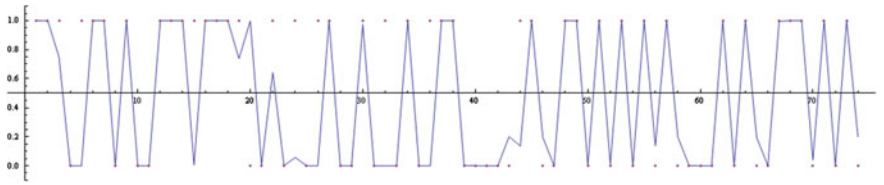

Fig. 1.21 The detailed validation of the best FNN. The classification performed by FNN filtered by the threshold function and the correct answers are shown by the *line* and *dots*, respectively. The accuracy: 89.2 %

The classification method is required to be accurate. It is not obvious whether 89.2 % accuracy can be considered satisfactory. Most importantly, however, is that such errors occur by "over-appreciation" improper solutions than exclusion of proper ones. Interestingly, the ranges of sizes and shapes of these corridors are very narrow. The smallest and largest are: a unit square {1,1} and rectangle {1,2} or {2,1}, respectively. Therefore such corridors could have been included as an additional "room" in the room chain. Nonetheless, in more realistic FLs the variety of sizes and shapes of possible corridors is enormous, and considering them as additional rooms would be impractical.

1.4 Sorting of "Proper" Configurations According to Additional Criteria

"The best" room layout can be found by ranking of the proper solutions according to additional criteria. E.g.:

- The geometrical simplicity of the entire layout or particular apartments;
- The size of the internal communication;
- Preferred relative position of particular rooms;
- The length of the layout perimeter, etc.

However, in the reality of architectural practice however, client's requirements can not be completely and precisely expressed as mathematically and implemented in the algorithm. Therefore it seems practical to generate a larger number of "very good" solutions and choose among them later on. E.g. Figure 1.22 shows certain room configurations of various layout perimeter lengths. Shorter perimeter means greater compactness of a layout. In architectural practice building compactness is an important factor in energy conservation [25] defined as the surface area to volume ratio.

Preferences regarding orientation of certain rooms are common in architectural design. It can be easily implemented as an additional criterion for FL ranking. Figure 1.23 shows an example where the preferred orientation of room #3 is the north-west corner of the building.

The sizes of corridors are another intuitive criteria, as shown in Fig. 1.24.

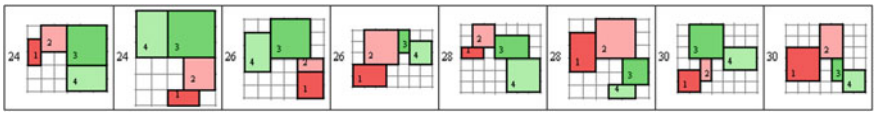

Fig. 1.22 Some examples of room configurations with external perimeters of different lengths. The perimeter lengths are shown on the *left* at each FL. Since the internal communication is not included in this computation, the corridor cells are hidden

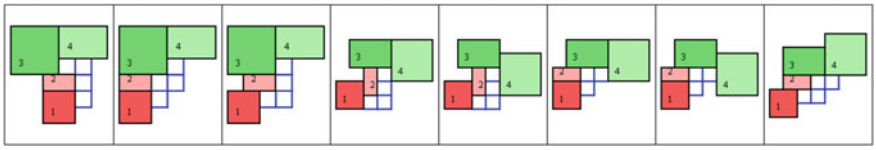

Fig. 1.23 A few FLs with room #3 positioned at the north-west corner of the building (*top left* in the plan)

Fig. 1.24 Some room configurations with the smallest possible corridors. Only the apartments' corridors are shown, the main corridors are hidden

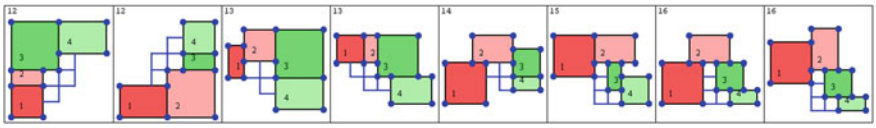

Fig. 1.25 Selected examples of FLs with decreasing geometrical simplicity. For each FL the value of simplicity parameter is shown in the *top left corner*

Geometrical simplicity is another rather intuitive quality of a FL. It is defined here as the number of unique coordinates of every corner of every room. In other words, when two corners of different rooms occupy the same position, that is overlap, they are counted as 1. Lower value of this parameter indicates less complex geometry of a FL, as shown in Fig. 1.25.

The corridors can also be included for the calculations of the simplicity parameter. Figure 1.26 shows the same FLs with additional simplicity parameter defined by the number of non-shared, that is independent nodes. It is quite common that the high geometrical simplicity of rooms results in low geometrical simplicity (in other words, high geometrical complexity) of the corridors and the other way around.

The simplicity parameters for rooms and corridors can be combined. The relative importance of these values can be adjusted by an additional parameter (weight). Figure 1.27 shows examples of FLs where the these simplicity parameters are added and equal 15.

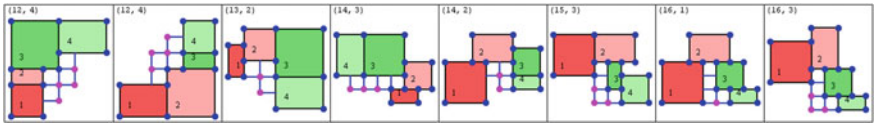

Fig. 1.26 Geometrical simplicity parameter additionally applied to the same FLs. The values of geometrical simplicity for the rooms and internal corridors are shown for each FL. FLs are sorted primarily according to the increasing corridor simplicity, which is measured by counting unique nodes (indicated in *magenta*)

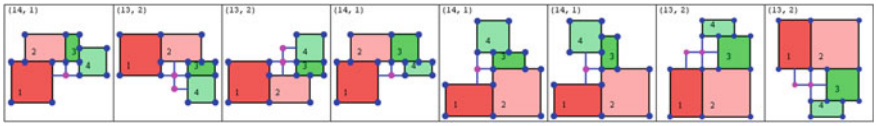

Fig. 1.27 FLs with the lowest value (15) of the combined simplicity parameters of the rooms and corridors

1.4.1 Final Ranking of the Functional Layouts According to Multiple Criteria

The final ranking of FLs according to multiple additional criteria can be formally expressed as a problem of multi-objective optimization:

$$\underset{\chi \in D}{\text{Minimize}}\, \mu(\chi) = \begin{pmatrix} \mu_1(\chi) \\ \mu_2(\chi) \\ \vdots \\ \mu_n(\chi) \end{pmatrix} \tag{1.5}$$

where, D is the feasible region in the decision space;
$\mu_1(\chi)$, ..., $\mu_n(\chi)$ are the coordinates of the image of χ in the objective space;
n is the number of objectives.

As an illustrative example, let us consider the following objectives for this optimization:

1. The preference regarding position of selected room. In this case, room #3 must be located as far in the South direction a possible;
2. The surface area of the corridors to be minimal;
3. The overall geometrical simplicity of the room layouts and corridors to be maximal. In other words, the complexity of these elements to be minimal.

Figure 1.28 illustrates these criteria in a three-dimensional space by assigning them to the individual coordinate axes.

In the architectural practice, the selection of the best FL is usually difficult. It is caused by the fact that not all design requirements can be or have been defined precisely. In such cases the ultimate decisions are made arbitrarily. So called Aggregate

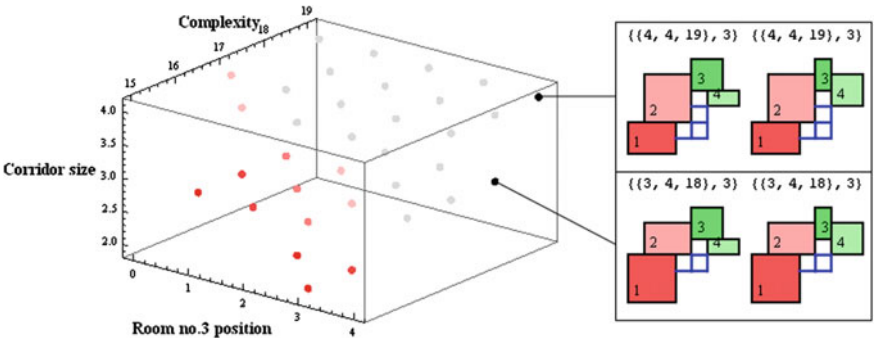

Fig. 1.28 The visualization of the objective function domain of the three-objective FL optimization. A single point in this space can correspond to several FLs, as illustrated on the *right*. The best solutions are shown in *red*. Saturation of *dots* decreases proportionally to the decreasing quality of the solutions

Objective Function (AOF, for short) is a straightforward method of combining the objective sub-functions in a weighted sum:

$$AOF = w_1 F_1 + w_2 F_2 + \cdots + w_n F_n \tag{1.6}$$

where, n: the number of objectives;
$F_1 \ldots F_n$: objective sub-functions;
$w_1 \ldots w_n$: parametric weights assigned to the objective sub-functions.

The main advantage of AOF is that the importance of the objective sub-functions can be fine-tuned by parametric weights. It is a straightforward method, however, when the number of objective increases, the individual manual adjustment of these weights becomes problematic [19]. In this example AOF combines the following objectives:

- The distance of room #3 to be located at the south edge of the FL: $4 > d_{R3} > 0$;
- The corridor size: $4 \geq cor_n \geq 2$
- The total geometrical simplicity (rooms and corridors): $19 \geq S_{FL} \geq 15$.

The above parameters have been normalized. Figure 1.29 shows the histogram of AOF with all three weights set to 1 for this three-objective optimization.

In the actual architectural problems, the determination of the weight values is rather difficult and requires several trial-and-error experiments. Furthermore, the exact evaluation of a real FL is far more unclear and arbitrary. Most importantly, the relative importance of the objective sub-functions is not only difficult to begin with, but often changes at the final decision-making stage. Since there are only three independent criteria, the objective function domain can be intuitively visualized in three-dimensional space, as shown in Fig. 1.30.

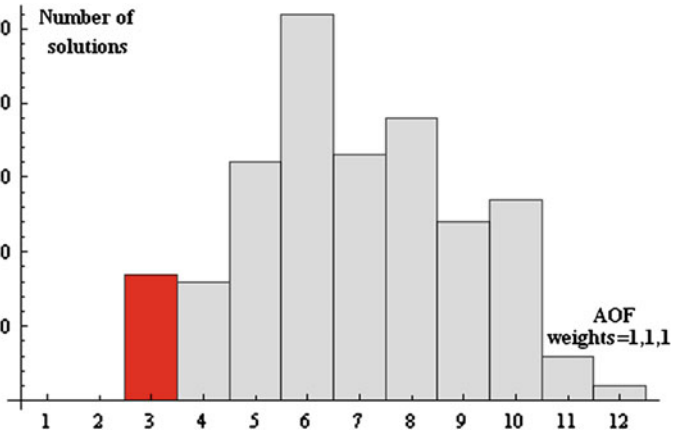

Fig. 1.29 *Red* indicates AOF $= 1 + 1 + 1 = 3$. There are seventeen such "ideal" FLs

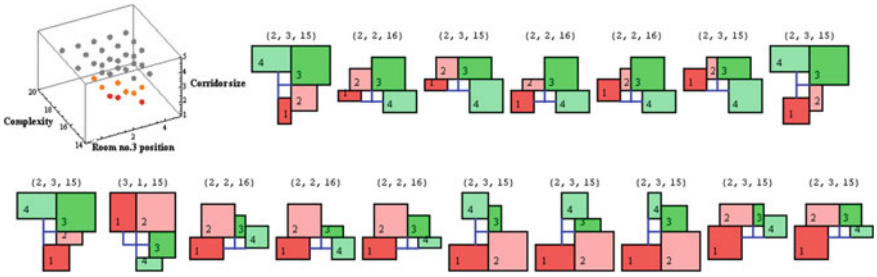

Fig. 1.30 The three-dimensional objective function domain is visualized in the *top left corner*. The best FLs are close to the coordinate system's origin. The coordinates of the best three FLs are: (3, 1, 15), (2, 2, 16), and (2, 3, 15). The individual values of the parameters, that is: i. the size of corridors, ii. proximity of room #3 to the south edge of a FL, and iii. the layout complexity are shown for each FL. The *dots* corresponding to the "second best" FLs are highlighted in *orange*

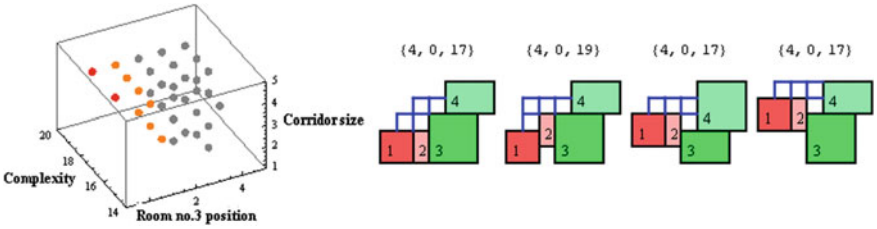

Fig. 1.31 In this example the location of room #3 at the south edge of the FL is dominant. Therefore the ranking of FLs runs along the corresponding axis. The center value (0) in the triplets indicates that in each case room #3 is as far south as the lot limits permit. Four FLs occupy two positions in the three-dimensional visualization indicated by *red dots*. The *dots* corresponding to the "second best" FLs in this ranking are highlighted in *orange*

The weights of the the independent parameters in AOF reflect the importance attributed by a designer. Figure 1.31 shows an example where the highest priority is given to the location of room #3 at the south edge of the FL.

1.5 A Realistic Case Study

For simplicity, the algorithm was illustrated here with extremely small examples. However it performs well for moderate size residential buildings (RBs). It is worth, however, to mention the scale of computation. Let us calculate the number of all possible FLs for a three-apartment RB having six, eight, and six rooms. Since there are fifteen non-square rooms the formula (1.3) must be multiplied 2^{15}. Therefore the total number of all possible configurations for a this three-apartment case with a sequence of twenty rooms is $2.7496 \times 10^{23} \times 2^{15} \approx 9.01 \times 10^{27}$. Despite this astronomical number, the algorithm operating on a single-core PC with 1.86 GHz CPU produced 24 potential solutions in approximately 25 min. Figure 1.32 shows all the results.

In this example, the initial constraints were very stringent and resulted in a relatively "narrow" search domain. The final selection among the 12 "proper" FLs shown in Fig. 1.32 was based on the following preferences:

1. The smallest apartment to be located at the North side;
2. The largest apartment to be located at the East side,
3. The north elevation to be simple, that is having minimal geometrical complexity.

The FL finally selected according to the above criteria is shown in Fig. 1.33.

This LF has been adjusted according to further technological requirements and served as a floor plan for a project of a multi-family RB, as shown in Fig. 1.34.

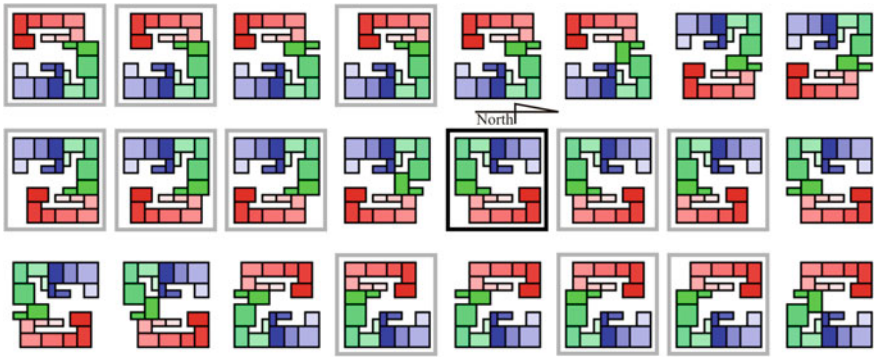

Fig. 1.32 The initial constraints given for this example were very stringent. Nevertheless, 50 % of the results are "proper" according to the internal communication criteria (indicated by frames). The finally selected FL is indicated by the *black frame*

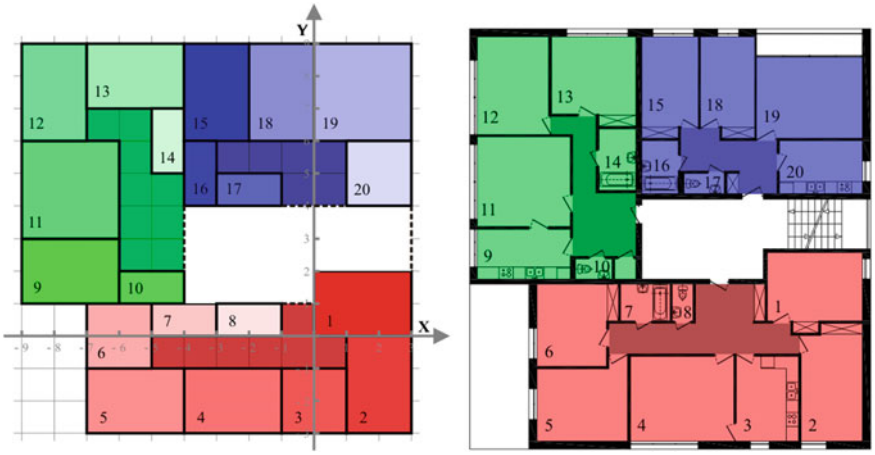

Fig. 1.33 On the *left* the optimal FL for the given layout problem. On the *right* an architectural floor-plan based upon this FL

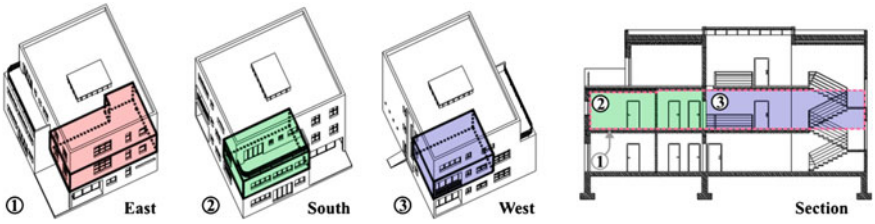

Fig. 1.34 On the *left* three axonometric projections of the RB design with indicated apartments based on the generated FL. On the *right* the longitudinal section. The *dotted line* indicates the location of apartment #1 which is not visible since it is in front of the plane of this section

The algorithm based on the Constraint Satisfaction Problem approach are usually very sensitive to the initial input. In practice, adjusting these initial constraints is rather difficult. Nevertheless, although in the present stage the algorithm operation is relatively time-consuming and additionally the parameter-setting requires some experimentation, such automatically generated FLs are acceptable from the architectural perspective.

1.6 Estimation of the Dimensions of Search Domain

In principle, the size of the search domain grows astronomically with the number of apartments and rooms. Therefore it is rational to estimate the potential number of solutions beforehand.

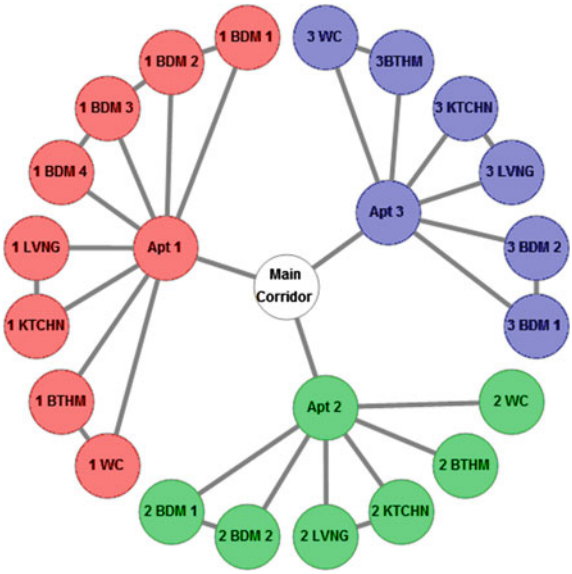

Fig. 1.35 A graph representing the relationships declared for the "realistic case study". This graph is planar, thus a solution for this FL problem may exist. If such a graph does not have at least one planar embedding, it means that the problem defined by corresponding relationships can not be solved. Node colors correspond to the colors of apartments

The numbers computed by formula (1.3) are often enormous. However, this formula also includes infeasible FLs e.g. the layouts with overlapping rooms. It also does not consider several other constraints which in fact greatly reduce the search domain. Therefore the formula (1.3) can not be considered satisfactory for accurate estimation of the search space size. However, preliminary determination whether the constraints are not too stringent can be performed by a planarity check of the required relationships, as shown in Fig. 1.35.

For a solution to exist, the condition of planarity is necessary but not sufficient. There are available algorithms for planarity examination. However, in a case of non-planarity of the graph representing the initial requirements, two problems occur: so called crossing minimization on almost planar graphs [13] and the decision which relationships (graph edges) to be removed. Graph-theoretic approach has already been implemented in a commercial software FactoryOPT [22] for optimization of a layout of a rectangular plan. This restriction is too strong, and presented in this chapter methodology does not limit the contours of buildings to rectangles. Moreover, it also uses other geometrical constraints which go beyond the relationships which can be directly expressed as a graph. Nevertheless graph-theoretic methods seem promising for preliminary validation whether initial constraints and requirements are reasonable (see Fig. 1.36) and for the feasible search domain approximation.

Figure 1.36 relates a graph, which is an abstract notion to "real" geometrical relationships of the elements of a functional architectural layout. Moreover, the real floor-plans of buildings are based on strict topological connections and notions of "interior" and "exterior" are fundamental. On the other hand, the dimensionless graphs usually have several geometrical interpretations, called embeddings. In

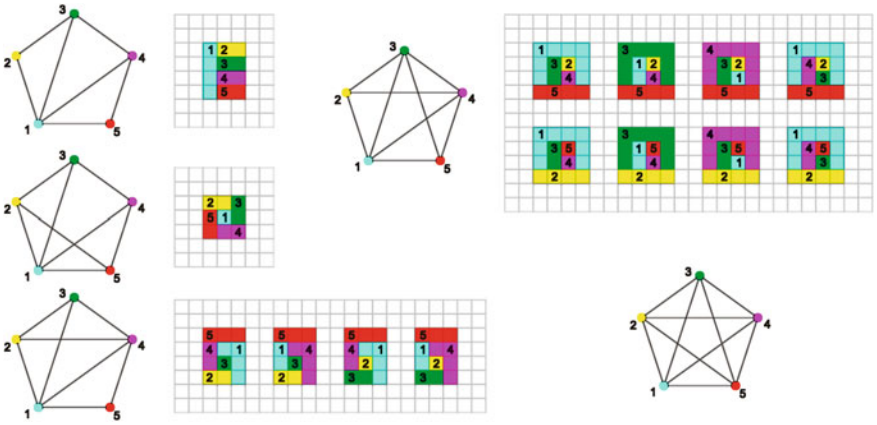

Fig. 1.36 Adjacencies of rooms represented by a graph and the corresponding all possible smallest FLs. Voids in FLs are not permitted. A non-planar graph (K_5) obviously does not correspond to any FL

summary, finding the smallest possible functional embedding for a given layout problem expressed as a set of relationships can help estimating feasibility of the search, the dimensions of the search domain. Alternatively, it could be a good start for application of a heuristic method.

1.7 Conclusions

- Architectural design belongs to creative human activities. It is and most likely will remain intuitive and arbitrary at least to a certain degree.
- The method described in this chapter approaches the problem of creation of an optimal architectural functional layout in an objective and systematical way. The main concept presented here is to sort allowable solutions according to variable, however, clearly defined criteria.
- The framework presented here is universal and customizable. The objectives, parameters, requirements, constraints and can be adapted to any layout problem.
- In the approach presented here the connections among the architectural spaces are confined to the two-dimensional plane. In real world, however, many spatial problems have three-dimensional nature [3]. Physical space is indeed three-dimensional, however, architectural problems due to their anthropocentric nature are strongly anisotropic in a sense that the vertical and horizontal movements are qualitatively different. In any building with multiple floors there are issues involving vertical transportation e.g.: pedestrian transportation via stairs, ramps, escalators or elevators; but also piping, ducts etc. However, it is not necessary to approach them as fully three-dimensional problems. A "2.5-D" approach appears the most rational, pragmatic and in effectively the most practical.

There are, however, several issues still to be addressed:

- Although the present version of the algorithm have performed rather satisfactorily, it can not be considered as user-friendly. Moreover, it should be improved for efficiency and interactivity.
- The classification method for selecting "proper" functional layouts to be also further improved. Possibly the machine learning method can be replaced e.g. by fuzzy logic [6], or at least enhanced to perform with higher predictability.
- Implementation of fuzzy logic and operations seem as the natural step in the further development of the algorithm. Some of the input data may naturally remain non-fuzzy e.g.: the numbers of apartments and even rooms, as well as the lot sizes and shapes. However, other requirements e.g.: regarding certain adjacencies, sizes and proportions of rooms have rather "fuzzy" nature.
- Moreover, the algorithms unlike a human designer performs in a few distinct steps. The application of fuzzy logic has potential of unification and simplification of this process as suggested in [30].
- At the present stage decision variables are adjusted by trial-and-error. More systematical approach is desirable.
- Development of reliable, robust and quick method for estimation of the solution space beforehand is also desirable.

References

1. Akin O, Dave B, Pithavadian S (1992) Heuristic generation of layouts (hegel): based on a paradigm for problem structuring. Environ Plann B: Plann Des 19(1):33–59
2. Baykan CA (1992) Formulating spatial layout as a disjunctive constraint satisfaction problem
3. Bier H, De Jong A, van der Hoorn G, Brouwers N, Heule M, van Maaren H (2008) Prototypes for automated architectural 3d-layout. In: Virtual systems and multimedia. Springer, pp 203–214
4. Buffa ES, Armour GC, Vollmann TE (1964) Allocating facilities with CRAFT. Harvard University Boston, MA
5. Chang PT, Lee JH, Hung KC, Tsai JT, Perng C (2009) Applying fuzzy weighted average approach to evaluate office layouts with feng-shui consideration. Math Comput Model 50(9):1514–1537
6. Chi-Hsing H, Jiang B, Lee E (1999) Fuzzy neural network modeling for product development. Math Comput Model 29(9):71–81
7. Cross N (1999) Natural intelligence in design. Des Stud 20(1):25–39
8. Damski JC, Gero JS (1997) An evolutionary approach to generating constraint-based space layout topologies. In: CAAD futures 1997. Springer, pp 855–864
9. Dave B, Pithavadian S et al (1987) Heuristic generation of layouts (hegel) based on a paradigm for problem structuring
10. Del Río-Cidoncha M, Iglesias J, Martínez-Palacios J (2007) A comparison of floorplan design strategies in architecture and engineering. Autom Constr 16(5):559–568
11. Flemming U, Baykan CA, Coyne RF, Fox MS (1992) Hierarchical generate-and-test vs constraint-directed search. Springer
12. Flemming U, Coyne R, Woodbury R (1994) Seed: a software environment to support the early phases in building design. In: Proceedings of IKM94. pp 5–10
13. Hliněnỳ P, Salazar G (2007) On the crossing number of almost planar graphs. In: Graph drawing. Springer, pp 162–173

14. Jo JH, Gero JS (1998) Space layout planning using an evolutionary approach. Artif Intell Eng 12(3):149–162
15. Kosiński W, Weigl M (1997) General mapping approximation problems solving by neural and fuzzy inference networks. Syst Anal Model Simul 30(1–2):11–28
16. Leśniakowska M (1996) Co to jest architektura? Kanon
17. Liang LY, Chao WC (2008) The strategies of tabu search technique for facility layout optimization. Autom Const 17(6):657–669
18. Liggett RS (2000) Automated facilities layout: past, present and future. Autom Const 9(2):197–215
19. Messac A, Puemi-Sukam C, Melachrinoudis E (2000) Aggregate objective functions and pareto frontiers: required relationships and practical implications. Optim Eng 1(2):171–188
20. Michalek J, Choudhary R, Papalambros P (2002) Architectural layout design optimization. Eng Optim 34(5):461–484
21. Mitchell WJ (1990) The logic of architecture: design, computation, and cognition. MIT press
22. Montreuil B, Venkatadri U, Donald Ratliff H (1993) Generating a layout from a design skeleton. IIE Trans 25(1):3–15
23. Saridakis KM, Dentsoras AJ (2008) Soft computing in engineering design-a review. Adv Eng Inform 22(2):202–221
24. Sharpe R, Marksjö BS (1986) Solution of the facilities layout problem by simulated annealing. Comput Environ Urban Syst 11(4):147–154
25. Smeds J, Wall M (2007) Enhanced energy conservation in houses through high performance design. Energ Build 39(3):273–278
26. Sowa J (2008) Conceptual graphs/john. In: Sowa F, van Harmelen F, Lifschitz V, Porter B (eds) Handbook of knowledge representation. Elsevier, pp 213–237
27. Tam KY (1992) Genetic algorithms, function optimization, and facility layout design. Eur J Oper Res 63(2):322–346
28. Terzidis K (2006) Algorithmic architecture. Routledge, p. 44
29. Wong SS, Chan KC (2009) Evoarch: An evolutionary algorithm for architectural layout design. Comput-Aided Des 41(9):649–667
30. Xiong Y, Rao SS (2004) Fuzzy nonlinear programming for mixed-discrete design optimization through hybrid genetic algorithm. Fuzzy Sets Syst 146(2):167–186
31. Yeh IC (2006) Architectural layout optimization using annealed neural network. Autom Constr 15(4):531–539

Chapter 2
Evaluation of the Quality of an Urban Square

Abstract As shown in the previous chapter, the evaluation of a building layout is difficult. The same applies to an urban layout. However, the basic evaluation of the quality of a plaza (*P*), that is the basic element of urban composition, is relatively straightforward. This chapter presents a method for an Automated Geometrical Evaluation (AGE) of a *P*. Firstly, nineteen plazas from various countries have been evaluated by twenty respondents in so called Human Subjective Evaluation (HSE). Secondly, a preliminary investigation of HSE including the identification of categories which are redundant is demonstrated. Thirdly, three normalized properties derived directly from a plan of *P* are introduced: smallness (*S*), enclosure (*E*), and regularity (*R*). Finally, the evaluation method of *P* based on these properties is discussed. AGE based on *S*, *E*, and *R* (NP_{SER}) shows good agreement with HSE. The quality rating based on NP_{SER} of *P*, namely: fair, good, and excellent is presented. Some outlying cases are shortly discussed.

2.1 Introduction

Urban design is a multifaceted discipline involving many engineering and designing fields such as urban composition, planning, development, architecture, landscape architecture, transportation, economics, law and finance, among others. Public space (*PS*) has been defined in several ways based on: use, ownership, access, use, etc. Here, it is defined according to Ref. [5] as follows: "*publicly accessible places where people go for group or individual activities*". According to Ref. [11] a good *PS* has the following qualities: it is meaningful, democratic, and responsive. In the same Ref. so called *public space index* has been introduced. It assesses the quality of *PS* by observational assessment of the following qualities: pleasurability, comfort, safety, meaningfulness, and inclusiveness. Moreover, Ref. [15] has implemented geospatial analysis for automatic detecting of:

- Neighborhoods;
- Neighborhood centers;
- Edges among urban neighborhoods;

M. Zawidzki, *Discrete Optimization in Architecture*, SpringerBriefs
in Architectural Design and Technology, DOI 10.1007/978-981-10-1106-1_2

- So called "mixes", that is spatial distributions of various kinds of land uses in the given area;
- Proximity, which is defined as: the number and quality of places which can be reached and in general the ability to reach urban places.

This chapter concerns with urban composition—a field of design dealing with human perception of space. Although urban composition is not necessary for a city to function, as shown in Fig. 2.1(1), it is often considered valuable.

Modern cities probably serve the citizens overall better than the old ones, however, the quality of urban composition seems to worsen drastically with the advent of so called International Style in 1920s. Other totalitarian movements such as fascism and communism have also contributed to this deterioration. It seems like the public spaces became perhaps more impressive, mostly in terms of size, but lost connection with the human scale, which resulted in their lesser overall appreciation. Figure 2.1(2), (3) show an example of a Renaissance city and International Style-inspired district from 1970s. Although the sizes and overall form seems similar, there are also dramatic differences. The principal elements of classical urban composition are: dominants (monuments and high-profile buildings such as: churches, palaces, city halls, etc.), street frontages and plazas (*P*). These elements suffice to compose clearly defined diverse spaces, which most importantly relate well to the human scale: the street frontage (usually with commercial functions) contrasts with larger, but still enclosed *P*s. The example shown in Fig. 2.1(3) neglects these rudimentary urban tools. As a result the hierarchy of spaces is lost, and the buildings indeed become merely a (...) *magnificent play of masses brought together in light* [7], as devised in 1923 by the protagonist of the International Style, Charles-Édouard Jeanneret alias Le Corbusier.

Although the definition of a dominant and a street frontage seems fairly straightforward (see Fig. 2.1(2)), the proper design of a *P* is substantially more difficult. Space syntax [8] is a well established set of theories and tools used for spatial morphological analysis with particular applications in urban science. The original research started in 1970s. An integration of space syntax into geographic information system (GIS) for modeling of urban spaces has been proposed in [10]. This chapter, however, concerns with a single *P*—an isolated node of the network and ignores the connectivity with other urban elements. According to [16], more important consideration in urban design than legibility is the aesthetics. The field study conducted in Dresden, Germany has investigated the relationship between urban form and the aesthetic experience [9], and argued that picturesque urban form may be more aesthetically appealing than other urban spatial arrangements. Here, however, the aesthetic value of *P*s is ignored and the legibility of space is considered only.

Traditional evaluation, whether a certain urban square qualifies as a "good one" is difficult and most importantly—rather intuitive and arbitrary. This chapter instead of speculating on urban composition presents a straightforward phenomenological approach.

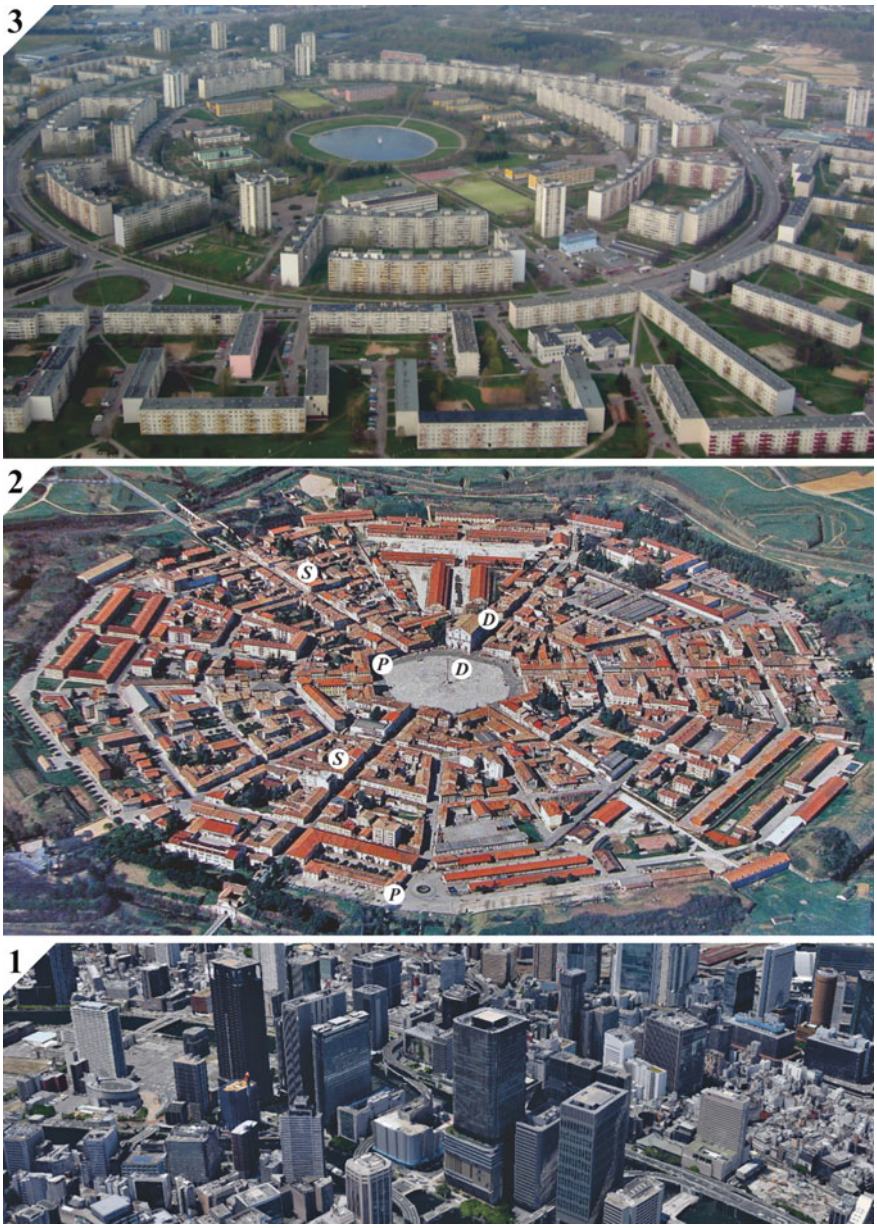

Fig. 2.1 **1** Osaka, Japan—no urban composition (present); **2** Palmanova, Italy (1593–1813), a city realized on utopian concept of Renaissance Ideal City. **3** Väike-Õismäe—a residential district in Tallinn, Estonia (1970). The sizes of Palmanova and Väike-Õismäe are comparable. *P*, *D*, and *S* stand: for plaza, urban dominant, and street frontage, respectively

2.2 Nineteen Plazas Subjected to Human Subjective Evaluation (HSE)

Figures 2.2 and 2.3 show: nine *P*s from Warsaw, Poland and ten worldwide *P*s, respectively. All *P*s have been evaluated by twenty respondents.

Each of the nineteen worldwide *P*s has been evaluated individually by twenty respondents. The evaluation has been done with a slightly modified questionnaire as used in [12]. Table 2.1 collects fifteen bipolar adjectival pairs and the semantic scale used in the questionnaire.

For aggregation of the values given by respondents, the responses have been transposed from −3–3 scale to 0–6. Further in text, only the "positive qualities", as indicated by boldface in Table 2.1 are used. They are called simply *qualities* from now on. Presented in this chapter approach differs from the methodology used in [12] in the following aspects:

- The main objective of this investigation is on the spatial perceptions of spaces. If possible, the aesthetic aspects were intentionally ignored.
- Instead of evaluation of photographs, respondents were exploring selected *P*s virtually by *Google Earth*.
- The group of respondents was smaller, namely twenty as oppose to three hundred persons.

The responses have been aggregated, averaged, and tabulated into the "matrix of plazas" (*MP*), that is nineteen (the number of *P*s) fifteen-dimensional (the number of HSE *qualities*) vectors. The transposition of *MP* forms the "matrix of qualities" (*MQ*). In order to find redundant *qualities*, the correlations among all *MQ* vectors have been investigated. Interestingly indeed, some responses have been practically equivalent. Namely, pairs: "Cozy"–"Comforting", and "Natural"–"Relaxed" showed 0.86 and 0.88 rate of correlation, respectively. Therefore "Comforting" and "Relaxed" have been omitted in further analysis. The values of responses for every *P* have been averaged, summed-up, and normalized to the range (0, 1). Therefore, this numerical value of measured assumed *quality* of *P* from now on is called "normalized accumulated quality", NAQ for short. The evaluations done by the respondents of all *P*s have been ranked by NAQ, as shown in Fig. 2.4.

2.3 Automated Geometrical Evaluation (AGE) of an Urban Square

The main concept presented in this chapter is to evaluate the spatial quality of *P*s solely by analyzing their plans. Such analysis is named here: the Automated Geometrical Evaluation of a *P*, AGE for short. Despite the opinion presented in Ref. [16] that: "*aesthetics is a more important consideration in urban design than legibility*", the focus of the investigations presented here is exactly on the "legibility" of a *P*.

Fig. 2.2 The nine *P*s from Warsaw, Poland: **1** Old Town Market, **2** Mariensztat, **3** Savior Sq., **4** Dabrowski Sq., **5** Politechnika Sq., **6** Union of Lublin Sq., **7** Constitution Sq., **8** Bank Sq., **9** Crossroads Sq

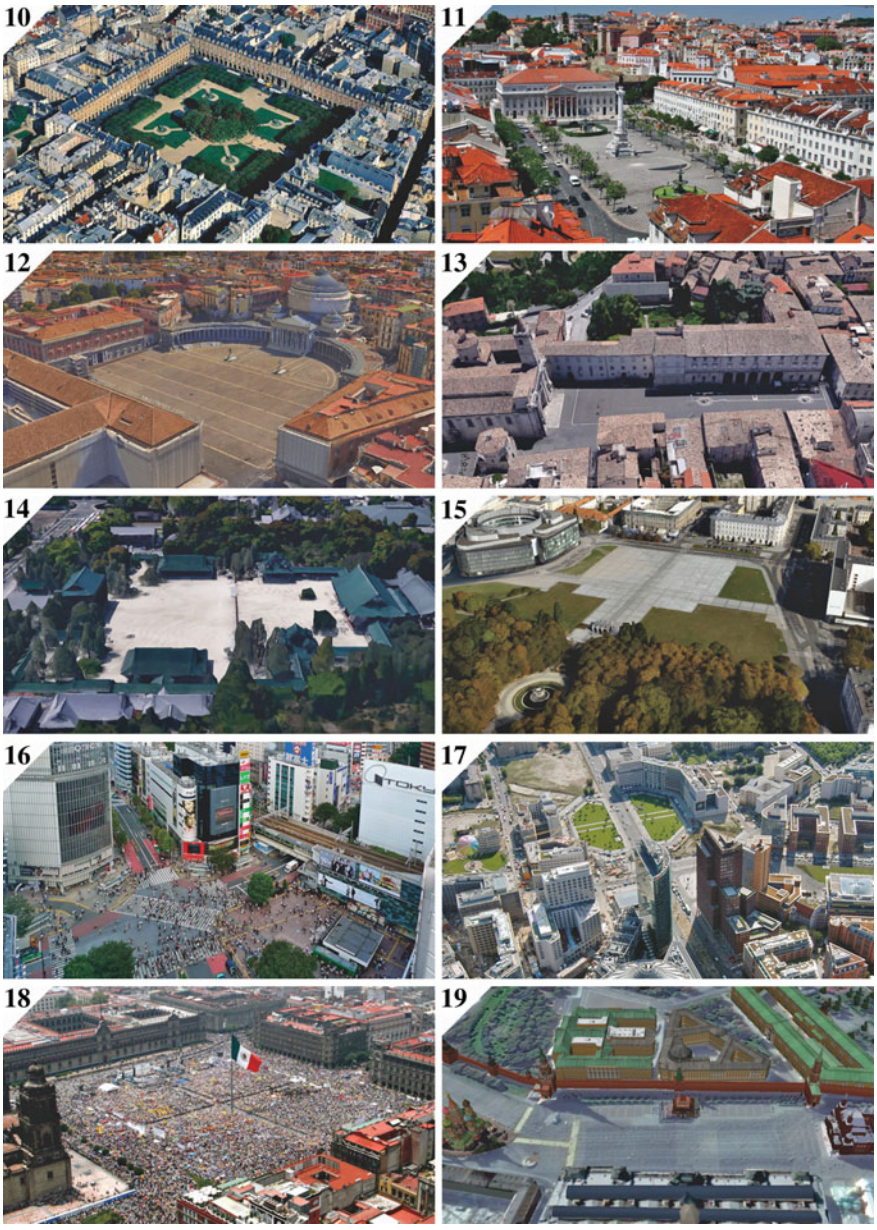

Fig. 2.3 **10** Place des Vosges, Paris, France; **11** Praça Dom Pedro IV, Lisbon, Portugal; **12** Piazza del Plebiscito, Naples, Italy; **13** Piazza Arringo, Ascoli Piceno, Italy; **14** Heian Jingu, Kyoto, Japan; **15** Pilsudski Sq., Warsaw Poland; **16** Shibuya, Tokyo, Japan; **17** Potsdamer Platz, Berlin, Germany; **18** Zocalo, Mexico City, Mexico; **19** Red Square, Moscow, Russia

Table 2.1 The questionnaire for human subjective evaluation (HSE) of plazas

1	Dull	−3	−2	−1	0	1	2	3	**Dynamic**
2	Repelling	−3	−2	−1	0	1	2	3	**Attractive**
3	Chaotic	−3	−2	−1	0	1	2	3	**Arranged**
4	Artificial	−3	−2	−1	0	1	2	3	**Natural**
5	Boring	−3	−2	−1	0	1	2	3	**Interesting**
6	Disturbing	−3	−2	−1	0	1	2	3	**Comforting**
7	Cacophonous	−3	−2	−1	0	1	2	3	**Harmonious**
8	Dysfunctional	−3	−2	−1	0	1	2	3	**Functional**
9	Tense	−3	−2	−1	0	1	2	3	**Relaxed**
10	Unsocial	−3	−2	−1	0	1	2	3	**Social**
11	Harsh	−3	−2	−1	0	1	2	3	**Cozy**
12	Uninspiring	−3	−2	−1	0	1	2	3	**Inspiring**
13	Plain	−3	−2	−1	0	1	2	3	**Diverse**
14	Flimsy	−3	−2	−1	0	1	2	3	**Sound**
15	Inaccessible	−3	−2	−1	0	1	2	3	**Accessible**

Conversely, the aesthetic, in other words, architectural aspects of *P*s are being disregarded, in accordance with Camilo Sitte: (...) "*just as there are furnished and empty rooms, so one might also speak of furnished and unfurnished plazas*"*(...)* [14]. Three geometrical properties (called simply *properties* from now on, which match HSE have been identified: *smallness*, *enclosure* and *regularity*.

2.3.1 Smallness

One of the fundamental properties of a *P* is its size. According to the classic book on urban design [14] by Camilo Sitte, "*A plaza that is too small usually does not give due effect to monumental buildings; on the other hand, one that is too large is, obviously, still more awkward (...). Such giant squares of vast dimensions occur in modern cities almost solely as drill grounds*". This quote indicates that size is a natural property of *P*, and that it has always been considered by urban planners. Most importantly, however, in principle, the relations between the sizes and NAQs of *P*s are not obvious. Surface area is a straightforward measurement of the size of a *P*. According to the responses in the questionnaires on the investigated nineteen worldwide *P*s, with sizes ranging from very large to medium, the "vastness" is not perceived positively. The surface area of each *P* has been re-scaled in relation to the extreme-sized *P* within the set. The smallest and largest *P*s are: Mariensztat (P_2, shown in Fig. 2.2(2)), and Zocalo (P_{18}, shown in Fig. 2.3(18)), respectively. The surface areas of P_2 and P_{18} are: approx. 4,500 m^2, and approx. 57,600 m^2. The former is within suggested in Ref. [2] range for "ideal plazas"; the latter has been selected as a reference, because despite its

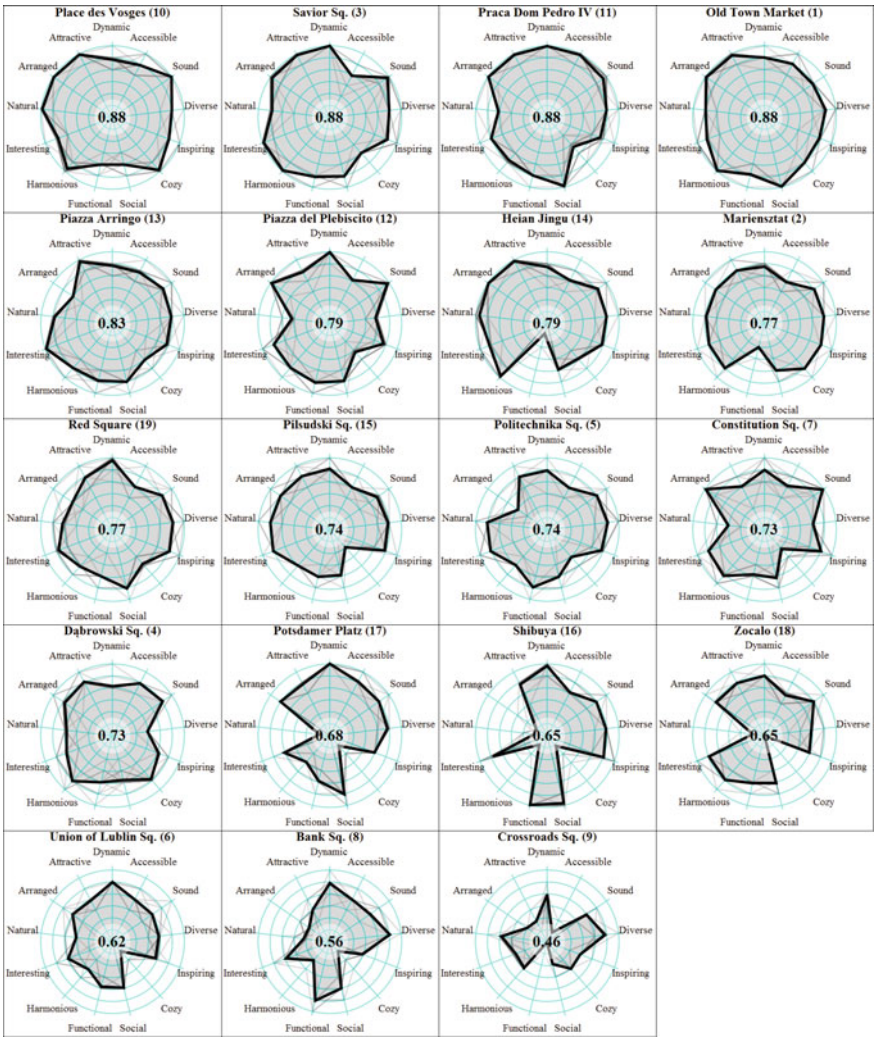

Fig. 2.4 Human subjective Evaluation. *P*s ranked according to the values of NAQ, shown in the center of every graph. All twenty individual evaluations of *P*s and their means are shown as gray lines and gray filling, respectively

enormous size, it has legible, well-defined form. There are even larger *P*s worldwide, however, they do not provide the sensation of enclosed space anymore. They can be designed by strict geometric relationships visible in their plan. However, they do not create perceptible barriers due lacking vertical elements in the perimeters of such *P*s proportional to their horizontal dimensions. In other words, such enormous *P* can be considered as an "open space" rather than an "urban space". Such gigantic *P*s are

listed here: [17]. Thus the notion of *smallness* (S), as a positive property has been defined, as follows:

$$S(P) = 1 - \frac{A_P - A_{P_2}}{A_{P_{18}} - A_{P_2}} \tag{2.1}$$

where A_P is the surface area of a given plaza P;

A_{P_2} and $A_{P_{18}}$ are the surface areas of: Mariensztat (P_2: 4,500 m^2) and Zocalo (P_{18}: 57,600 m^2), respectively.

2.3.2 Enclosure

In the same book [14], Sitte writes explicitly: "*the main requirement for a plaza is the enclosed character of its space*". So called, enclosure (E) is a property derived directly from P's plan. The calculation of its numerical value is explained in Fig. 2.5.

The enclosure is naturally normalized and the range of possible values is from 0 to 1, for completely open, and completely enclosed spaces, respectively.

2.3.3 Regularity

Qualitative evaluation of regularity is very easy for human perception. However, the quantitative evaluation of the degree of regularity is much more difficult. Moreover, a strict mathematical evaluation of the regularity of given shapes is also problematic. There are algorithms capable of detecting regular polygons implemented for recognizing of traffic signs. Refs. [4, 13] present the application of these algorithms

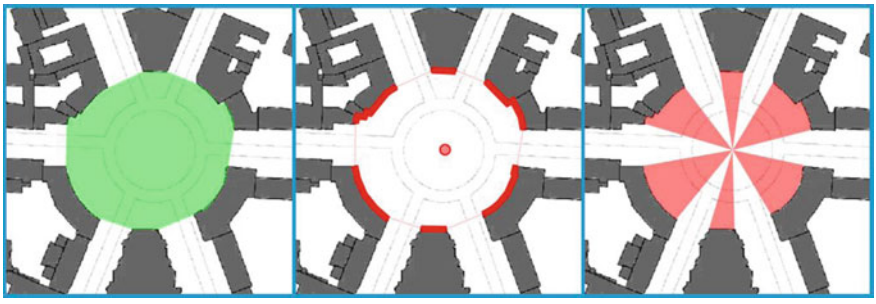

Fig. 2.5 Three steps of enclosure calculation: **1** Identification of the entire surface area occupied by a P (shown in *green*). **2** Identification of the vertical elements (usually building facades) which surround the P and the centroid of the P (shown as thick *red lines* and a *red dot*, respectively). **3** The surface area of the sectors projected from the centroid on the surrounding vertical barriers is calculated. Finally, the latter area is divided by the total area of the P

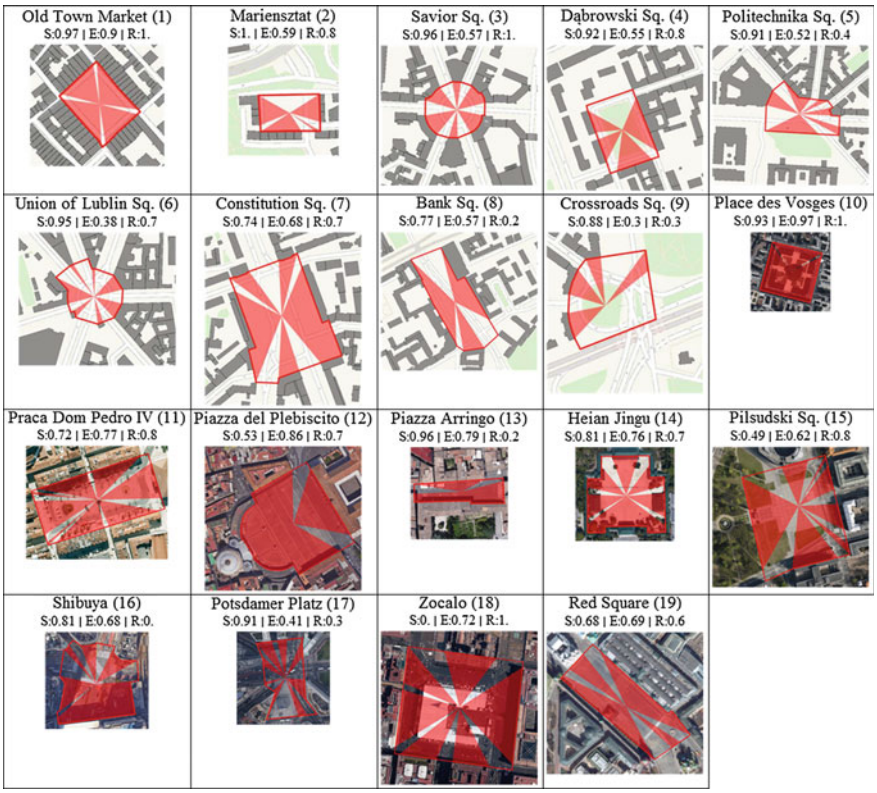

Fig. 2.6 All *P*s with calculated: smallness (*S*), enclosure (*E*), and regularity (*R*)

for analyses of photographic images. Traffic signs, however, are convex, therefore the research on regularity focuses on this class of geometrical shapes [3, 6]. In this chapter the assessment of regularity has been done arbitrarily based on intuition of each respondent. The assigned values range from 5 to 0, for perfectly regular (for example P_{10} shown in Fig. 2.3(10)) to extremely irregular (for example P_{16} shown in Fig. 2.3(16)), respectively. Figure 2.6 shows the plans of all nineteen *P*s with numerical values calculated for the three *properties*.

2.4 Correlation Between Automated and Human Evaluation of Plazas

Figure 2.7 shows correlation between the "normalized accumulated quality" (NAQ) pf *P*s assessed in the "human subjective evaluation" (HSE) to the three geometrical *properties* defined above. As Fig. 2.7 indicates, the three *qualities*, namely: "diverse",

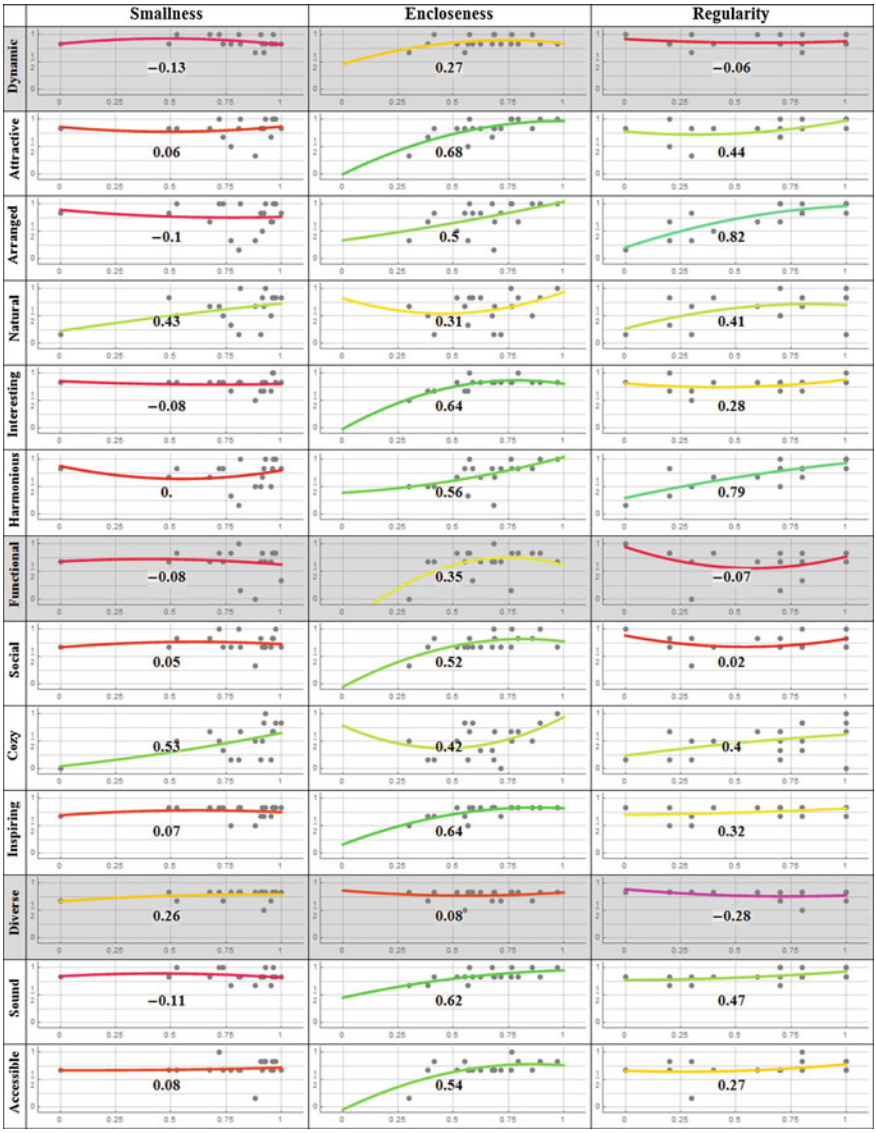

Fig. 2.7 Correlations among NAQ values of the subjective *qualities* (shown as rows) and geometrical *properties* derived directly from *P*s plans (shown in columns). Color of the *fitting line* corresponds to the correlation level. *Cyan* and *red* represent the strongest and weakest correlations, respectively. *Gray background* indicates the cases where there is no correlation between the *qualities* and *properties*

"functional", and "dynamic" show practically no correlation to the *properties*, thus these *qualities* have been removed from the mathematical model.

The NAQs of *P*s have been compared to the normalized totals of the respective *properties*, as shown in Fig. 2.8.

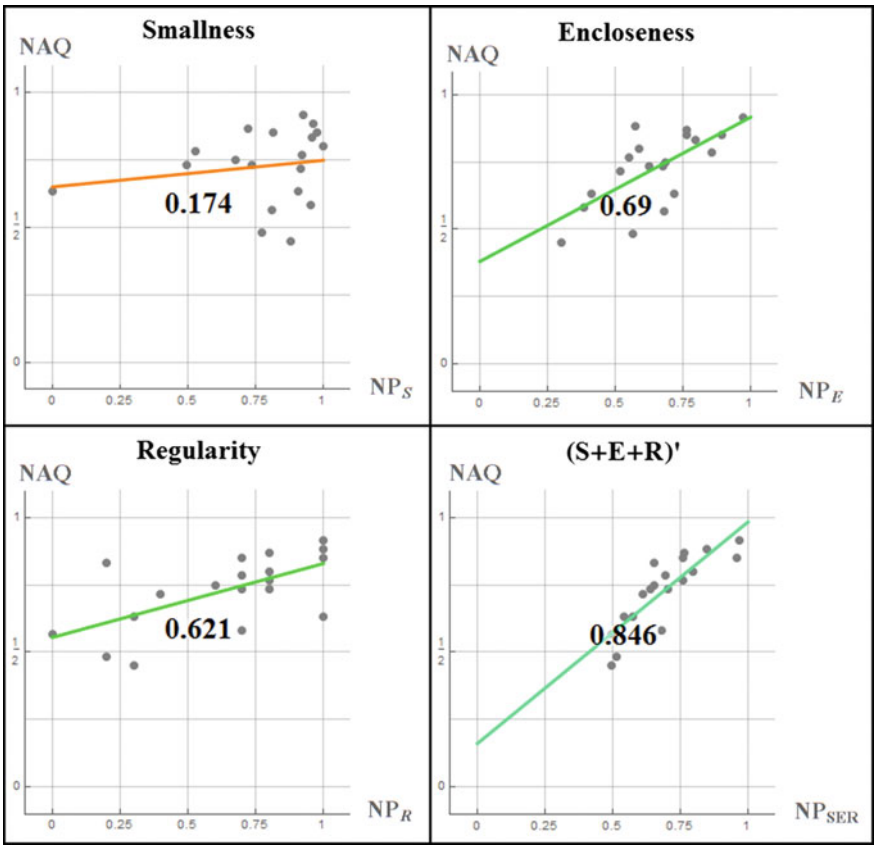

Fig. 2.8 Correlations Between NAQ and individual and combined and normalized geometrical *properties* of *P*s. The color convention of the *fitting lines* as in Fig. 2.7

As Fig. 2.8 indicates, the correlation of normalized geometrical *properties* of *P*s based solely on: smallness, enclosure and regularity show poor, good and good correlation to NAQ, respectively. Combining these three *properties* to a single normalized sum (NP_{SER}) improves the correlation with NAQ.

2.4.1 AGE and HSE Correlation

Finally the correlation between the automated geometrical evaluation (AGE) based on NP_{SER} with the human subjective evaluation (HSE) based on NAQ has been investigated. Figure 2.9 compares the two types of evaluations: NP_{SER} and NAQ for all nineteen *P*s. *Ps*. The sequence of *P*s in this Figure follows the sorting based on NAQ. As Fig. 2.9 indicates, there is a good agreement between NAQ values and the

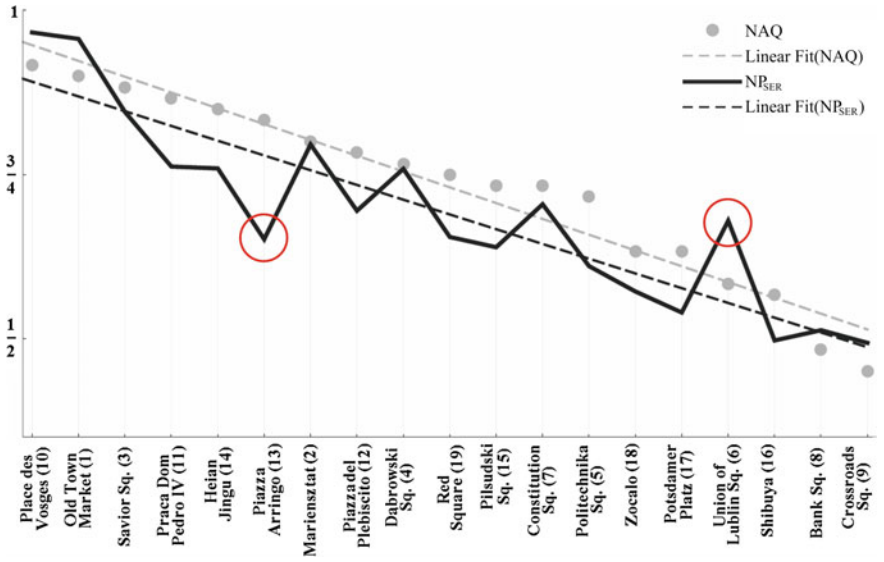

Fig. 2.9 Nineteen worldwide plazas sorted according to NAQ and subjected to the automated geometrical evaluation (AGE) by NP_{SER}. *Red circles* indicate two exceptional cases: one under-rated and one over-rated

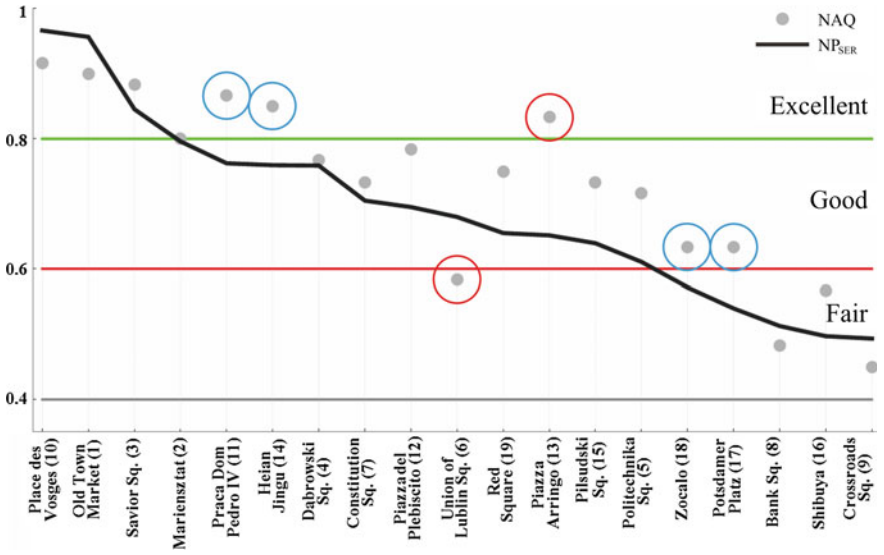

Fig. 2.10 Rating of plaza quality according to AGE divided into three arbitrary classes: excellent, good and fair. *Red circles* indicate the same exceptional cases as in Fig. 2.9. *Cyan circles* indicate four cases which have been "under-rated" by AGE

Table 2.2 The exceptional cases

P index	**NAQ**	**NP_{SER}**	**Comments**
6	0.58	0.68	Union of Lublin. This P is quite well defined. Two tollhouses indicate that it used to be the administrative border of the city. This results in P_6 being a major communication node with heavy vehicular traffic. Despite the urban attractiveness, these circumstances make it less pleasant for the visitors.
13	0.83	0.65	Piazza Arringo. Due to the proportions in plan, this P can viewed as a widened street. It does not form a well defined coherent space. However, due to the beauty and harmony of the surrounding architecture, as well as the usage of urban compositional elements (e.g. two fountains), P_{13} is an exquisite public space.
11	0.87	0.76	Praça Dom Pedro IV. This place has been one of the main squares in Lisbon since the Middle Ages. It is very large, which resulted in lower NP_{SER} rating. However, the size fits well the population demand which is not included in AGE. Moreover, P_{11} is a masterpiece of classical monumental urban space.
14	0.85	0.76	Heian Jingu. The plan of P_{14} is somewhat complicated. However the urban composition, placement of plants and presence of wooden architecture makes this place very attractive.
18	0.63	0.57	Zocalo. P_{18} is extremely large. In this study it is the maximum size reference for rescaling the property of smallness. Nevertheless it is the main square in central Mexico City, one of the most populated cities in the World. Thus, its size seems quite relevant.
17	0.63	0.54	Potsdamer Platz. P_{17} in the present form has been completed in the late 1990s. Despite its under-defined form it is an attractive social place in Berlin.

Red and green backgrounds indicate the case where a P has been: “over-rated” and “under-rated” by AGE, respectively

linear model of NP_{SER}. Figure 2.10 shows the values of the automated geometric evaluation (AGE) of all *P*s, and divides them in three classes: excellent, good and fair. The sequence of *P*s in this Figure follows the sorting based on NP_{SER}.

As Fig. 2.10 indicates, six *P*s have been misclassified by AGE. These errors are qualitatively rather small.

Five *P*s have been rated lower by AGE than HSE, once AGE has "over-rated" a *P* in comparison to HSE.

The cases of "under-rating" by AGE, in other words *P*s "over-rated" by HSE can be explained by the fact that either these *P*s are situated in exceptionally attractive locations and (or) these *P*s are exceptionally aesthetic. Although in this survey the aesthetic values have been purposely ignored, it may indicate that it is not fully achievable by human respondents, and most likely they have been subconsciously biased towards aesthetically appealing *P*s. On the other hand, one case where AGE has given higher value than HSE can be explained by heavy vehicular traffic at this location which most likely is repelling to the respondents, causing them to view it lower. All the exceptional cases are commented in detail in Table 2.2.

2.5 Conclusions

- Many of the plazas considered in this chapter are among the best in the world, thus the quality rating proposed here may be viewed as rather "harsh". Even the plazas here rated "fair" can most likely be considered as "elite".
- The number of users assumed for a given plaza is a principal parameter in urban design. However, since presented here model of AGE focuses solely on the geometrical properties derivable from the plans, this parameter has not been included in the model. Nevertheless, it could be easily implemented if the relevant data was available.
- The definition of *smallness* presented here seems over-simplified. It simply gives linearly better values for smaller plazas. This proportionality is based on two plazas selected arbitrarily. It is rational to relate the smallness of the plaza to the number of its potential users.
- AGE presented here deliberately excludes the aesthetics. Despite this major limitation, it gives meaningful results. Most importantly, it is not intended as an "absolute evaluation" tool. However, it provides a reliable and straightforward evaluation methodology for identification of areas for potential improvements of a given plaza. Nevertheless, as Table 2.2 indicates, aesthetics of surrounding buildings may strongly affect the perception of a plaza.
- It is desirable that the evaluation of plaza's regularity was also done not in an arbitrary, but in an automatic way. This issue, however, seems particularly difficult. Alternatively, if a fully automated method would not be possible, more systematic and rigorous should be considered.
- the Human Subjective Evaluation (HSE) model is very simple as it adds a number of responses and normalizes the value to the range (0, 1). It would be interesting to

further investigate the inter-dependencies among the *qualities*. AGE model could be fine-tuned to more systematical HSE model.

- The model od AGE presented here requires some amount of "manual" labor, which to a certain degree could be automated. However, the idea of a fully automated AGE based on aerial photography, although very attractive, seems very difficult. It is mostly due to the fact that even advanced aerial photography recognition methods [1] are not fully reliable. It is usually caused by the radiometric similarity between the image background and roofs of the buildings.

References

1. Ahmadi S, Zoej MV, Ebadi H, Moghaddam HA, Mohammadzadeh A (2010) Automatic urban building boundary extraction from high resolution aerial images using an innovative model of active contours. Int J Appl Earth Obs Geoinf 12(3):150–157
2. Alexander C, Ishikawa S, Silverstein M (1977) A pattern language: towns, buildings, construction, vol 2. Oxford University Press
3. Aloupis G, Cardinal J, Collette S, Iacono J, Langerman S (2009) Detecting all regular polygons in a point set. arXiv:09082442
4. Barnes N, Loy G, Shaw D, Robles-Kelly A (2005) Regular polygon detection. In: Tenth IEEE international conference on computer vision, 2005. ICCV 2005, vol 1, pp 778–785. IEEE
5. Carr S (1992) Public space. Cambridge University Press
6. Chalmeta R, Hurtado F, Sacristán V, Saumell M (2013) Measuring regularity of convex polygons. Comput Aided Des 45(2):93–104
7. Corbusier L (1931) Towards a new architecture. Courier Corporation
8. Hillier B, Hanson J (1984) The social logic of space. Cambridge University Press
9. Isaacs R (2000) The urban picturesque: an aesthetic experience of urban pedestrian places. J Urban Des 5(2):145–180
10. Jiang B, Claramunt C (2002) Integration of space syntax into gis: new perspectives for urban morphology. Trans GIS 6(3):295–309
11. Mehta V (2014) Evaluating public space. J Urban Des 19(1):53–88
12. Perovic S, Folic NK (2012) Visual perception of public open spaces in Niksic. Proc Soc Behav Sci 68:921–933
13. Piccioli G, De Micheli E, Parodi P, Campani M (1996) Robust method for road sign detection and recognition. Image Vis Comput 14(3):209–223
14. Sitte C (1986) City planning according to artistic principles. Rizzoli
15. Talen E (2011) The geospatial dimension in urban design. J Urban Des 16(1):127–149
16. Taylor N (2009) Legibility and aesthetics in urban design. J Urban Des 14(2):189–202
17. Wikipedia: list of city squares by size. http://en.wikipedia.org/wiki/List_of_city_squares_by_size

Part II
Crowd Simulation

Crowd simulation (CS) is a well-established field of research, which is used for various purposes, e.g., organization of mass events, planning escape routes in buildings, etc. Chapter 3 describes Crowd-Z (CZ): a user-friendly environment for CS in any floor plans provided. In Chap. 4, CZ is used for experiments investigating the reliability of certain crowd simulation setups. Chapter 5 presents a creative implementation of CZ for optimization of a floor plan in respect of a crowd flow.

Chapter 3
Crowd-Z

Abstract This chapter presents Crowd-Z (CZ): a user-friendly framework for crowd simulations (CS) in any floor-plans. The crowd dynamics component of CZ is a straightforward agent-based model. Such CSs can be carried out at every stage of architectural or urban design process: from early sketches to the final blue-prints. Most importantly, CZ accepts the initial input in practically any form, e.g.: pre-processed drawings produced by Computer-Aided Design (CAD) software, digital images, free-hand drawings, etc. Selected methods of acquisition of the CS environment are demonstrated and illustrative with practical examples. Finally CZ is evaluated against commercially available software and with some "classic" CS experiments.

3.1 Traditional Grid and Guideline Systems

Most, if not all architectural layouts are based on some kind of a grid system. There are three regular, also called "Platonic" tessellations. The practical use of square, hexagonal and triangular grids in architectural and urban design and CS are: extremely common, relatively rare and very rare, respectively. "Platonic" tessellations divide Euclidean planar space into congruent units of the same shape and surface area. The symmetry group of regular tilings is transitive on the tiles. They are homogeneous with respect to vertexes, tiles and edges and are strongly edge-homogeneous [5]. This is equivalent to an edge-to-edge tiling by congruent regular polygons. The use of this property has long history in various kinds of design. In early seventeenth century, Kepler has given it the first rigorous mathematical consideration in [10].

3.1.1 Architecture

Architectural projects are designed and realized in Euclidean space. There are a number of reasons for applying grid and guideline systems, especially for the planar (2D) space of architectural sketches and drawings:

M. Zawidzki, *Discrete Optimization in Architecture*, SpringerBriefs
in Architectural Design and Technology, DOI 10.1007/978-981-10-1106-1_3

1. composition aid
2. modularity constraints
3. relation to the human scale
4. structural constraints.

Various grid and guideline systems have been implemented in architectural and urban design practice since antiquity. The most common has been and still is one of three regular tessellations: the square grid. Figure 3.1 shows two examples, from Renaissance (completion in 1487) and Structuralism (1960).

As Fig. 3.1(1) indicates, the grid forms a system of constraints where the axes of primary structural elements such as columns and walls are to be placed. Figures 3.2 and 3.3 show floor-plans based on the two other regular grids: triangular and hexagonal, respectively.

The sizes of architectural grids vary substantially, as they can be set arbitrarily. Moreover, grids do not have to be applied uniformly over entire floor-plan. An example of a compound rhombic grid is shown in Fig. 3.4.

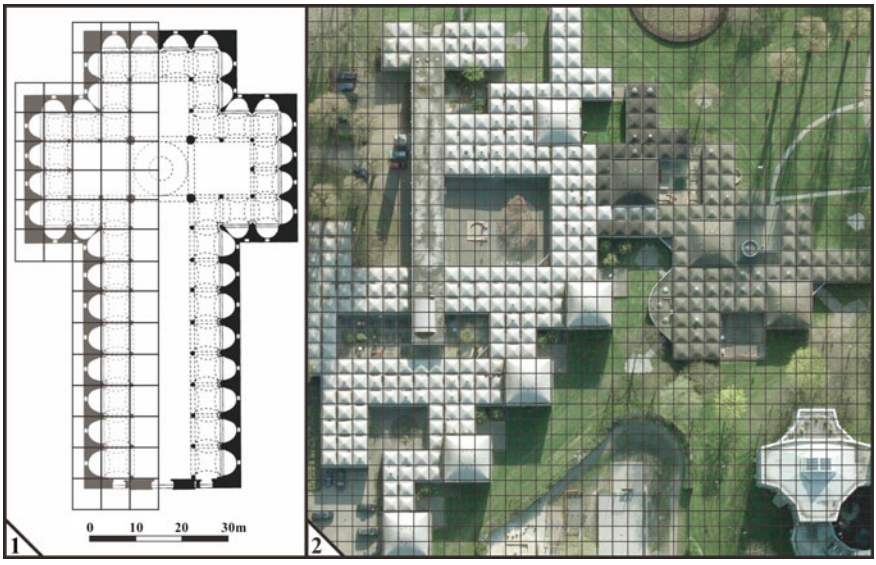

Fig. 3.1 **1** Plan of Santo Spirito, Florence, Italy. Design by Filippo Brunelleschi in 1434-6, construction commenced in 1446, completion in 1487). **2** An aerial photograph of Aldo van Eyck's Amsterdam Orphanage (completed in 1960). Both buildings are shown in the same scale. The grid has been superimposed to emphasize the formal rigor of the compositions

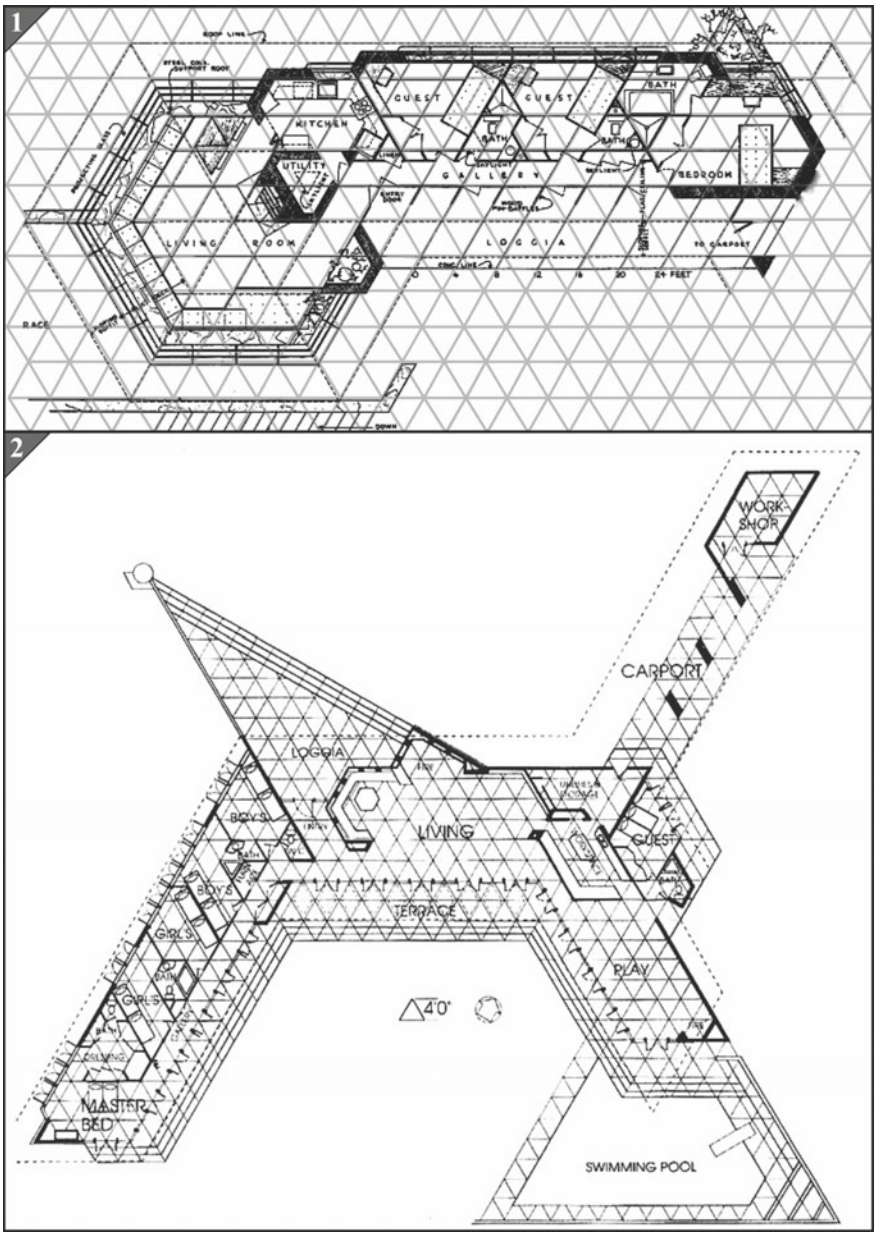

Fig. 3.2 Floor-plans of residential buildings based on triangular tessellation. Design by Frank Lloyd Wright: **1** Walker Residence (1948), **2** Fawcett house (1955)

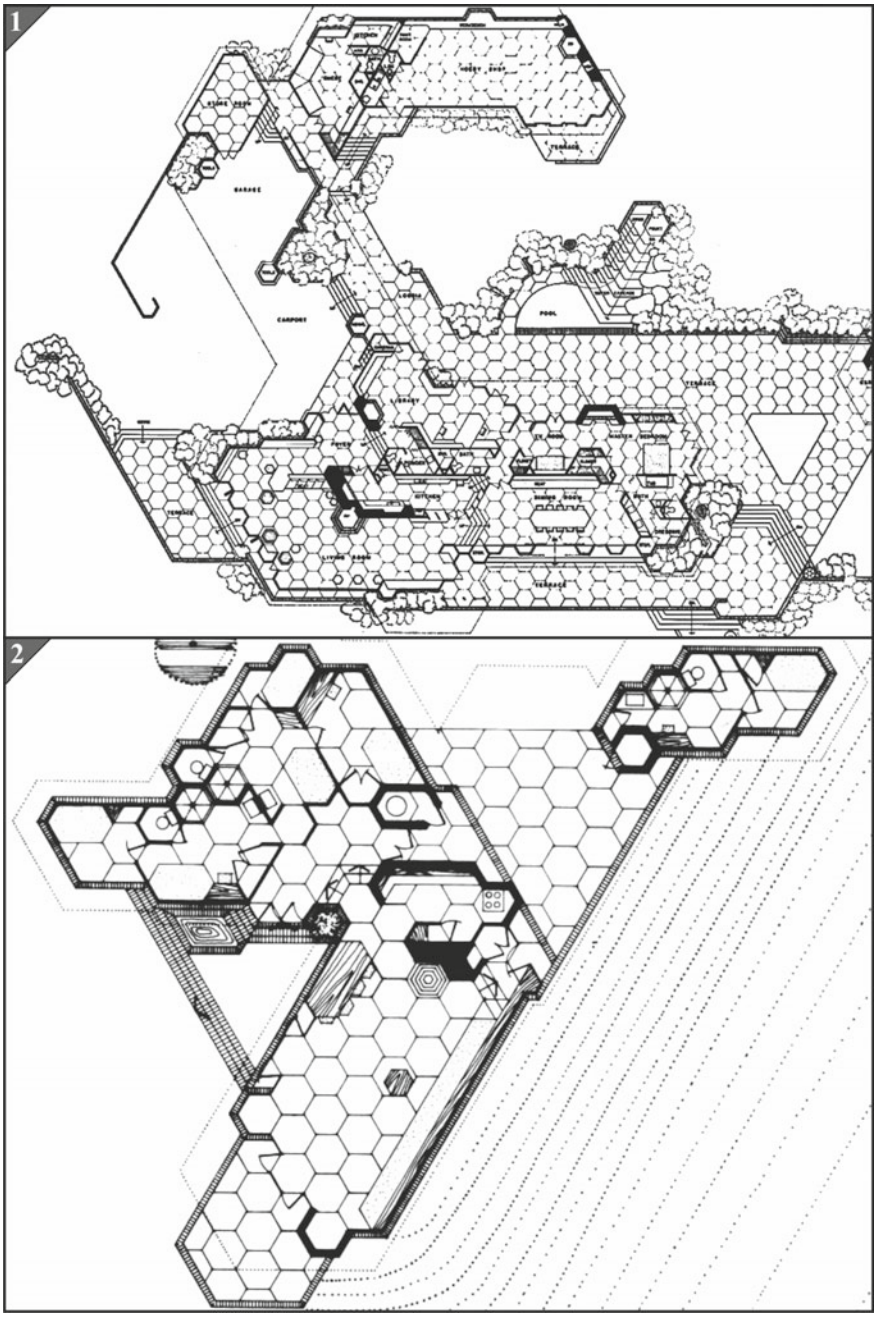

Fig. 3.3 Floor-plans of residential buildings based on hexagonal tessellation. Design by Frank Lloyd Wright: **1** Hanna-Honeycomb house (1937), **2** Bazett house (1939)

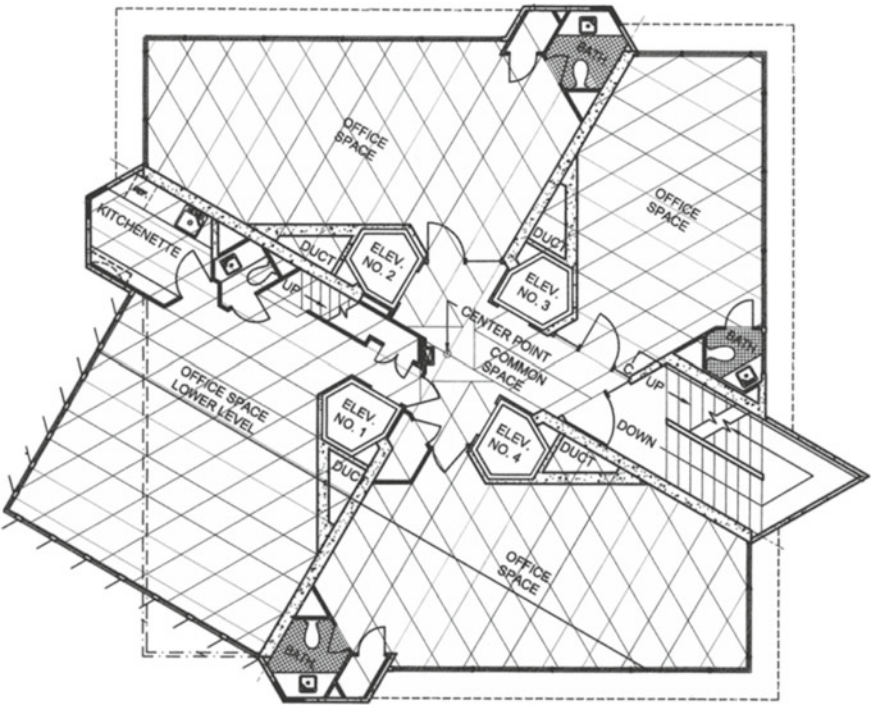

Fig. 3.4 Price Tower—typical floor. A multi-story office building; design by Frank Lloyd Wright, completion in 1956

3.1.2 Urban Design

A number of so-called *planned cities* have been laid-out relatively complex hierarchical systems with nested guidelines depending on the scale of the urban composition. Figure 3.5 shows an example of Detroit, USA. The rigorous regular grid seems to be less suitable for urban composition than for architectural design. E.g. in 1906 Schulze-Naumburg criticized the rectangular grid as impractical [26], as shown in Fig. 3.6.

Nevertheless, a number of cities have been laid-out on such rigid grid systems. Figure 3.7 shows urban compositions based on strict rotational grids. Figure 3.8 shows urban compositions based on rigid rectangular grids.

As Figs. 3.7 and 3.8 indicate, most of such compositions are boring and overwhelming. There are only a few exceptions, e.g. Palmanova in Italy and Barcelona in Spain, where the quality of the architectural content and detail offset the monotony of the composition. In general, however, such approach can be considered *inhumane*.

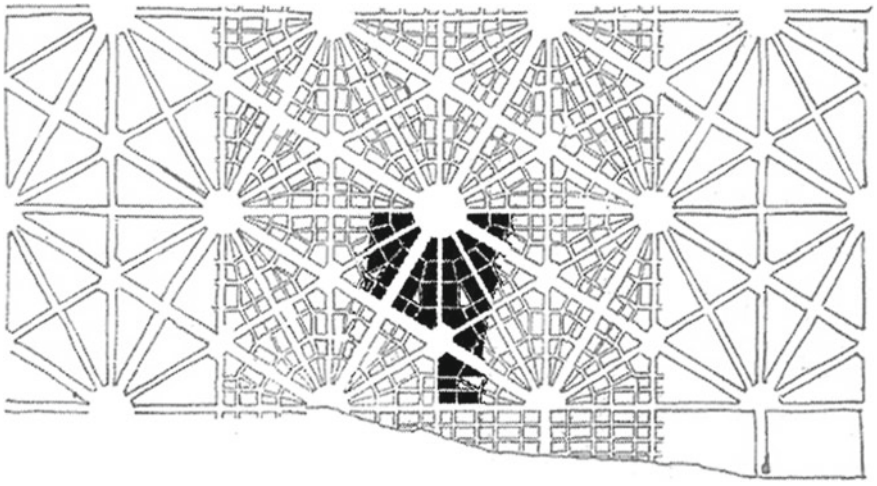

Fig. 3.5 Detroit, USA; the original urban composition designed by Augustus B. Woodward in 1805. The completed parts are filled in *black*

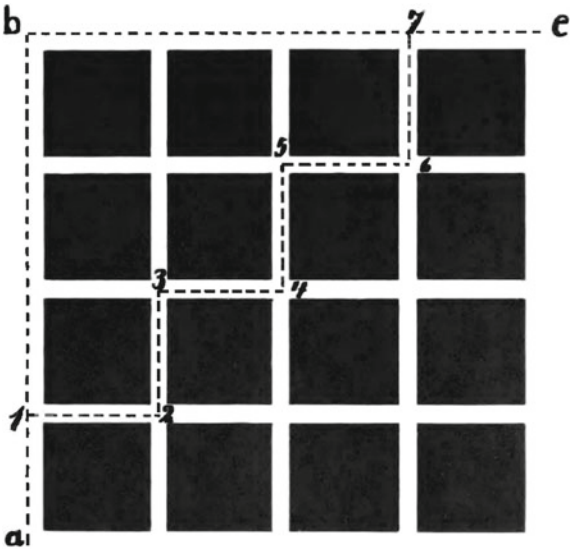

Fig. 3.6 Paul Schulze-Naumburg, *Kulturarbeiten IV*, (1906) p. 65: "*An example of impractical arrangement of square street network. The connection between a and c has always the length of a–1–2–3–4–5–6–7–c, which is equal to a-b-c*". © https://archive.org. Such arrangement is also equivalent to "*Taxicab geometry*", which is often used in computer simulations

Fig. 3.7 Satellite photographs: **1** Palmanova, Italy. **2** Plaza Del Ejecutivo, Mexico City, Mexico. **3** Burning Man, Black Rock City, Nevada, USA. **4** Glyfada, Athens, Greece.
Daily overview | Satellite images © 2016, DigitalGlobe, Inc

3.2 Crowd Simulation

Several mathematical models of the dynamics of pedestrian movement and interactions have been developed in the past. The main areas of applications are: organization of mass events, planning escape routes in buildings, etc. In principle, the models of crowd dynamics fall in one of the two categories:

1. space-continuous models based on Newtonian approach [6, 8, 9, 31]
2. rule-based models with quantified time and space discretized in a regular grid, by means of transition probabilities [2, 7, 15, 22].

Fig. 3.8 Satellite photographs: **1** Placa de Tetuan Barcelona, Spain. **2** La Plata, Buenos, Aires, Argentina. **3** Santosh Park-Uttam Nagar, Delhi, India. **4** Tianguis Mexico City, Mexico. **5** Central Park, New York City, USA. **6** Jeongwang-dong, Ansan, South Korea. Daily Overview | Satellite images © 2016, DigitalGlobe, Inc

So called agent-based models (ABMs) [1, 11, 28] belong to the second category and are particularly relevant to the content of this chapter. Most of these mathematical models describe the quality and quantity of the crowd dynamics (CD) rather satisfactorily. Moreover, a number of commercially available programs have been based on these models, e.g.: PedGo [13], ASERI, [25], VISSIM [17], etc. Unfortunately, such programs are not commonly used to support the design process in architectural and city planning offices. This is mostly due to the following issues: i. architects and planners do not see the benefits of such experiments for the design, and ii. the prices, the steepness of the learning curve, and simply—"unfriendliness" are discouraging. This chapter describes Crowd-Z (CZ), a user-friendly platform where untrained users can experiment on their floor-plans with CSs.

3.3 The Crowd Dynamics Model in Crowd-Z

In a nutshell, the purpose of creating Crowd-Z (CZ) was to provide an algorithmic environment where anyone could intuitively perform CSs, namely:

1. To help understanding the CDs in floor-plans provided. In other words, for simulations of possible usage scenarios.
2. To investigate undesired phenomena such as local over-crowding, or underused of spaces.
3. To support dynamic interaction between CSs and geometries of the analyzed layouts. Specifically, to support on-the-fly simulations while performing alterations i.e. creation, deletion, displacement etc. of the elements of the environment such as walls, pillars, exits, entrances, etc.
4. To investigate the influence of the behavioral profiles of the agents on the collective behavior of the crowd.
5. To allow not only for the analytical, but most importantly—for creative applications of CSs in architectural and urban design.

The objective of CZ is not to mathematically model the CD as precisely as possible, but rather to allow for maximal interactivity between the agents and the simulation environment. Although available advanced CD models simulate CD more realistically, CZ at the present stage offers a straightforward and reliable model which produces acceptable results. Most importantly, the Crowd Dynamics Component (CDC) of CZ is an independent module, which can be further developed to capture more nuances of CD. Nevertheless, since the intention was to provide an intuitive framework even for persons untrained in this field of research, the present CDC is controlled by only a handful of variables.

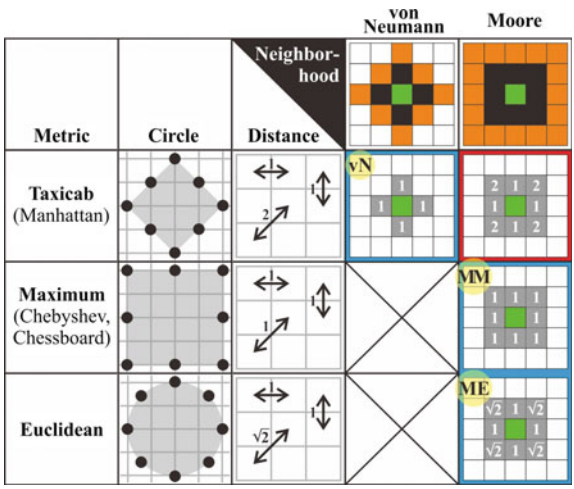

Fig. 3.9 Basic combinations of the neighborhoods and metrics. Combinations used here for CSs are indicated by *blue* background. The neighborhoods can have ranges larger than one. Radii 1 and 2 are shown in *black* and *orange*, respectively

3.3.1 Neighborhood and Metric

In plane geometry, if the points p and q have Cartesian coordinates (x_1, y_1) and (x_2, y_2), their distances are:

- In the Euclidean metric: $D_E(p, q) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$;
- In the Taxicab metric: $D_T(p, q) = |x_1 - x_2| + |y_1 - y_2|$;
- In the Maximum metric: $D_M(p, q) = Max\ (|x_1 - x_2|, |y_1 - y_2|)$.

The von Neumann's neighborhood and Moore's neighborhood are the classic definitions of the sets of cells surrounding a given cell for potential interactions. In CSs the most commonly employed are: the "Taxicab", Maximum and Euclidean metrics. Figure 3.9 illustrates the combinations of these neighborhoods and metrics.

The following neighborhood/metric combinations are employed in presented here CSs: von Neumann @ Taxicab metric (vN, for short), Moore's @ Maximum metric (MM), and Moore's @ Euclidean metric (ME). The unused combination Taxicab @ Moore is indicated by red background in Fig. 3.9. Its impracticality is due to the computational complexity being higher than vN and equal to ME, while being not as realistic.

3.3.2 Distance Potential Field

A number of mathematical models describing the pedestrian strategy of the movement towards an exit have been analyzed in the past [14, 19]. Here the CDC implements the most straightforward approach, that is following the shortest pathways.

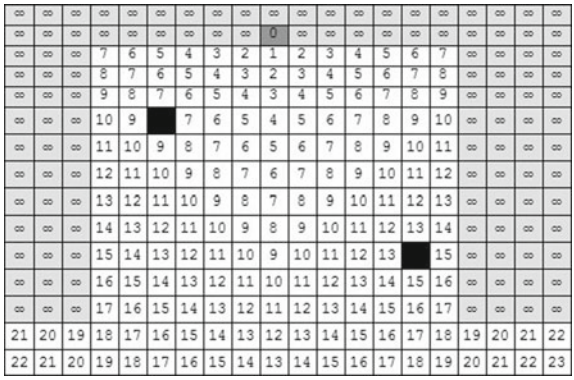

Fig. 3.10 A sample DF based on von Neumann neighborhood. 0 and ∞ correspond to the exit and wall cells, respectively. *Black squares* indicate other agents. *White*, *light gray* and *dark gray* background indicate "walkable" cells, walls, and the exit, respectively

They are defined for every location in the grid of the discretized layout by minimal distances to the exits. The wall cells and exits are given values of ∞ and 0, respectively. So called distance potential field (DF) tabulates the complete distance information for a given layout. CZ uses statical DF which is not influenced by the crowd movement. Values of DF_{ij}^{E} are proportional to the distance from an exit E to a grid position (i, j). The distance is defined by the metric e.g.: Euclidean or "taxicab" [30] and the kind of neighborhood e.g.: von Neumann or Moore, etc., as illustrated in Fig. 3.10.

In layouts with more than one exit, the DF^k is calculated individually for every exit k. Next, they are combined to a single DF_{ij}^{*}, by using the lowest value of all DFs for every cell (i, j):

$$DF_{ij}^{*} = \min_{k=1,\cdots,n} DF_{ij}^{k}, \tag{3.1}$$

where n is the number of exits.

Reference [18] compares different ways of computing DFs. In presented here ABM both the space and time are discrete. Time is divided into time-steps, and at each time-step the agents proceed sequentially. Such sequences can constructed differently, e.g.: they can be random—so called random shuffled update, RSU for short [17], or fixed as in so called ordered sequential update (OSU). For investigation on other types of pedestrian strategy see [14, 19], in particular on the update schemes—see [22]. In the "global scale", every agent follows the information provided by DF and consequently, proceeds to the nearest available exit. Since no more than one agent may occupy any grid cell at once, in order to avoid collisions the movement of the agents in the "local scale" follows this simple procedure:

- If there is no empty cell in the neighborhood due to occupation by other agents or wall cells—halt.
- Otherwise, look up the values of the available cells in the neighborhood provided by the DF and move according to the *perkiness* (explained in the following subsection).

Table 3.1 Three basic pedestrian strategies in CZ

Perkiness	Operation
Lazy (L)	L only proceeds to an available neighbor cell if the new position corresponds to a **lower** value in the DF
Conservative (C)	C only proceeds to an available neighbor cell if the new position corresponds to a **lower or equal** value in the DF
Perky (P)	P proceeds to **any available** neighbor cell

3.3.3 Perkiness

Perkiness is the parameter which determines the tendency of agents to displace. Table 3.1 lists the basic modes implemented presently in CDC.

Moreover, the *perkiness* can be dynamical and change according to various CD aspects. E.g. in order to reduce the congestion the willingness of agents to move can change from initially set *Lazy* to *Perky*. This kind of adapting behavior can be applied to individual agents (e.g. as a function of their effective velocities) or globally to the entire set of agents (e.g. depending on the global characteristics of the crowd such as flow rate). Whenever an agent can move to more than one equipotent destinations, the selection is random. Moreover, presented here CDC allows for varied velocities of agents. The implementation is straightforward—the positions of particular agents can be updated several times during a single time-step. In other words, an agent can proceed by multiple steps in one move. Complex phenomena such as the consequences of crowd inhomogeneity [24] can be analyzed in CZ by assignment of various maximum velocity to the agents. Previous research has shown that simulations with 0.05 rate of both fast and slow walking agents have the best agreement with empirical experimentations [3]. CZ allows for tracking and recording the instantaneous velocity of each agent. It is therefore possible to adjust the effective movement of any individual agent.

3.4 Illustrative Examples

Nowadays visual information is usually managed by well-established methods of digital imaging. The appropriation of such regular grid-based data for ABM is relatively easy. CZ is particularly user-friendly in this regard. The following subsections illustrate three basic methods of providing digital images to CZ: i. direct user's input, ii. upload, and iii. import from a Computer-Aided Design (CAD) software.

3.4.1 Direct User's Input

A computer scan of a hand-drawn sketch on paper or hand-drawing with a computer mouse seem the most intuitive input methods for most designers. Figure 3.11(1) shows a scanned sketch of an early draft. For a meaningful CS the scale of the experimental environment must be proper. In the literature the empirical maximum density of human crowd is 6.25 Persons/m^2 [29]. Consequently, the minimum space required for an agent is 0.16 m^2. This is equivalent to a $40 \times 40\,cm^2$ square cell in a grid. This is the common size of an agent usually assumed in CS of human pedestrians in discrete models [24]. In the presented here visualizations one pixel corresponds to a single agent. Figure 3.11(2) shows the simulation environment scaled accordingly.

In this example 250 agents visit three, two or one exhibit and exit. The number and sequence of visited exhibits is random. Initially the visitors are distributed also randomly. In this CS this scenario takes approx. 1 min. Instantaneous velocity of each visitor is dynamically visualized during the experiment, as shown in Fig. 3.12(1).

So called heat map (HM) is a common method of representing intuitively the behavior of crowds. It records the frequency of agents visits in every cell, and in the visualization renders saturation proportional to this frequency. CZ also dynamically displays the HM, as shown in Fig. 3.12(2). Moreover, CZ supports dynamic

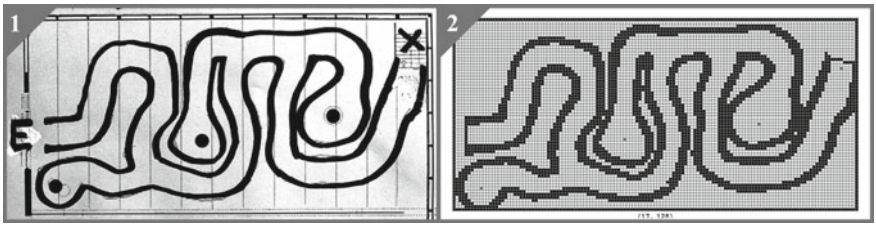

Fig. 3.11 An early draft of an exhibition space. **1** The sketch of the plan shows free-form partition walls. Letters X and E indicate the exit and entrance, respectively. **2** CZ user-interface includes a panel for interactive edition of the CS environment. A "mouse-click" inverts the value of a grid cell

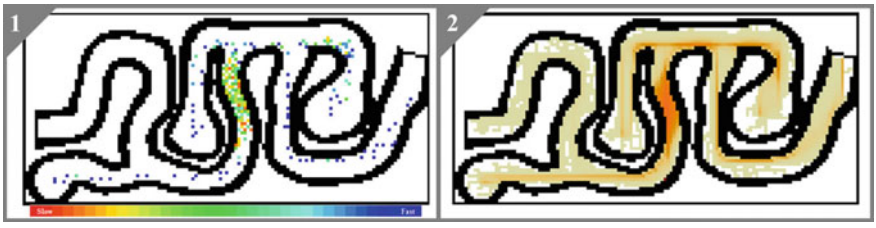

Fig. 3.12 Screenshots of the CZ user interface: **1** Instantaneous velocity of each agent is dynamically displayed. In the middle section local velocities are clearly lower due to congestion. **2** Dynamically displayed corresponding HM confirms the over-crowding of the middle section of the plan

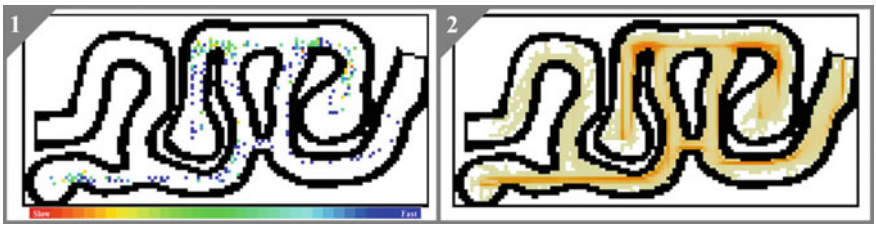

Fig. 3.13 Screenshots of the CS after modification of the plan, which reduces the crowd congestion. Sub-figures **1** and **2** show the visualization of agents' local velocities and the HM, respectively

interaction with the simulation environment. E.g. the problem of congestion identified in Fig. 3.12(2) can be immediately alleviated by creation of a pathway to the neighboring corridor, as shown in Fig. 3.13.

3.4.2 Evacuation from Saint Peter's Basilica

This subsection describes an evacuation study of a larger floor-plan, represented here by the Papal Basilica of Saint Peter in the Vatican, SPB for short. A digital image of SBP can be easily acquired from the Internet. Figure 3.14(1) shows the floor-plan rescaled to the proper grid size. Figure 3.14(2) visualizes DF^*. It combines 15 DFs computed for each individual assumed exit according to Formula 3.1.

In this study randomly distributed 10^4 agents proceed to 15 exits. This setup was also intended to test the computational robustness of CZ, therefore the number of agents in comparison to the experiment presented in [21] is nearly tripled. Moreover, each exit comprised of a single cell, only. Thus this CS takes approx. eight hours in desktop PC (CPU Intel Xeon W5590). Figure 3.14(3) shows an intermediate screenshot of HM after partial evacuation. Figure 3.14(3) also indicates qualitatively some congested areas around certain exits. Figure 3.15 shows quantitative data regarding the numbers of agents using particular exits.

As Fig. 3.15 indicates, in this scenario more than half of the crowd proceeds to the exit #1. As a result it is overcrowded and the neighboring area—congested. Most of the exits: #3, #4, #5, #6, #7, #8, #14 and #15 are practically unused. This information is fundamental for evacuation safety and the analysis of the space use. Such CSs can be easily performed in CZ even by users unfamiliar with crowd modeling.

Another experiment has been performed in the same environment to investigate the relationship among crowd density, "error rate", agents' *perkiness*, and evacuation time. The results have been compared with simulations based on so called Floor Field Cellular Automaton (FFCA) model [4, 12].

Crowd density (ρ) measures the average number of persons in a surface area unit; in this experiment ρ: 0.045 (50 agents in SPB) $< \rho \leq$ 0.965 (9750 agents in SPB).

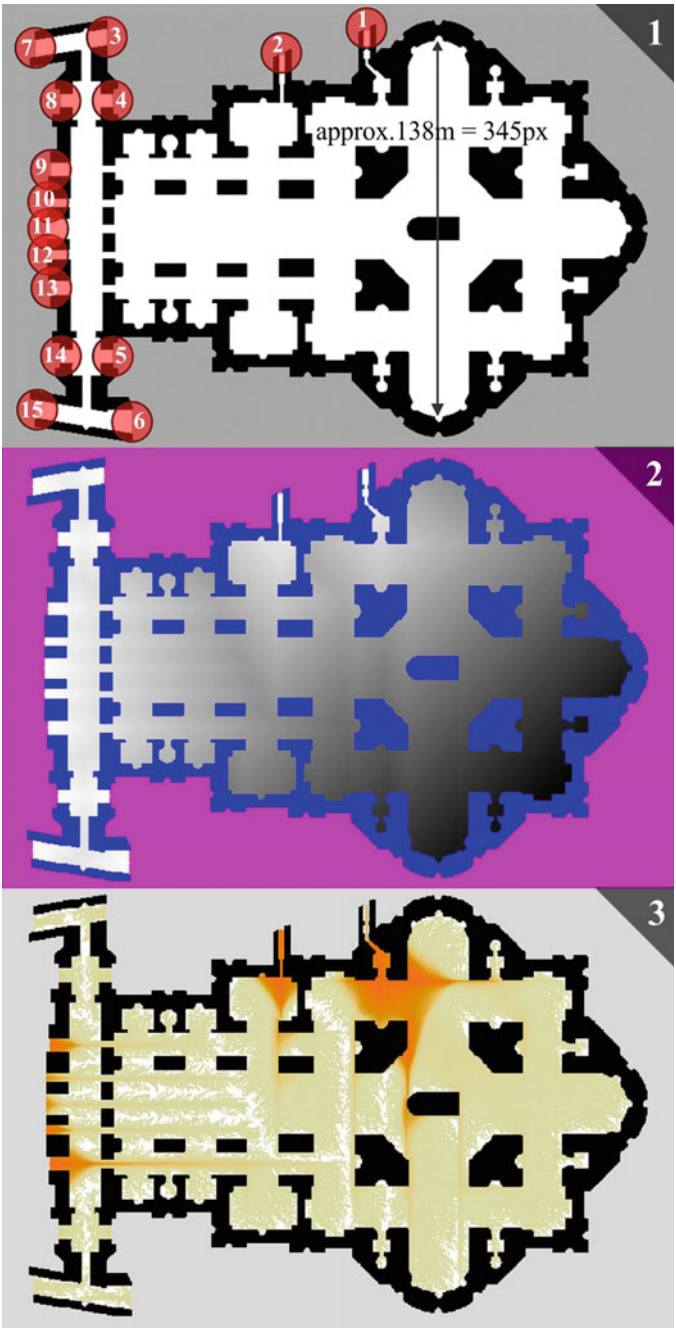

Fig. 3.14 SPB: **1** The floor-plan with highlighted 15 assumed exits. The transverse dimensions is approx. 138 m which correspond to 345 pixels. **2** A visualization of DF^*; the lightness of cells is proportional to the proximity to the exits. **3** Screenshot of HM after the evacuation of approx. half of the agents

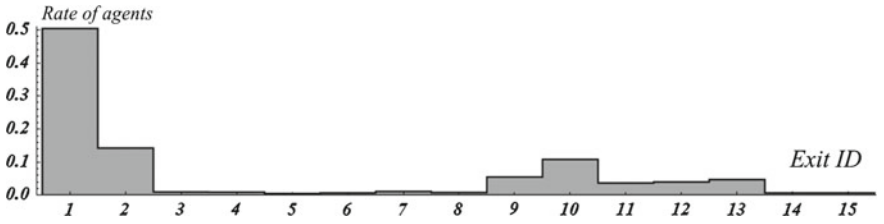

Fig. 3.15 The distribution of agents among 15 exits. The results of 100 trials with 10^3 *conservative* agents at OSU have been averaged

In principle, agents walk extremely efficiently, in other words do not make any mistakes. In many cases, however, such hyper-efficiency may be rather unnatural in simulating human pedestrian movement. Error rate (eR) is an additional parameter in range [0,1] which introduces the element of randomness to the model. For $eR = 1$ and $eR = 0$: agents perform purely "random walk", and there is no randomness to the agents' movement, respectively.

The simulations have been performed for all three types of *perkiness*. In the preliminary trials: the *conservative* crowd has evacuated from SPB in the fewest number of time-steps, the evacuation time of *lazy* and *perky* crowds were comparable. Therefore the *conservative* mode of movement has been applied in the main simulations. Several CSs have been preformed for increasing number of agents from 50 to 9750, as shown in Fig. 3.16.

In this setup the *conservative* crowd evacuates from SPB in less then half of the time used by the FFCA crowd. This is due to the fact that FFCA models more subtle interactions in the crowd. Despite the discrepancy in the evacuation speed, the description of the nature of the phenomenon is the same: there is a linear increase of

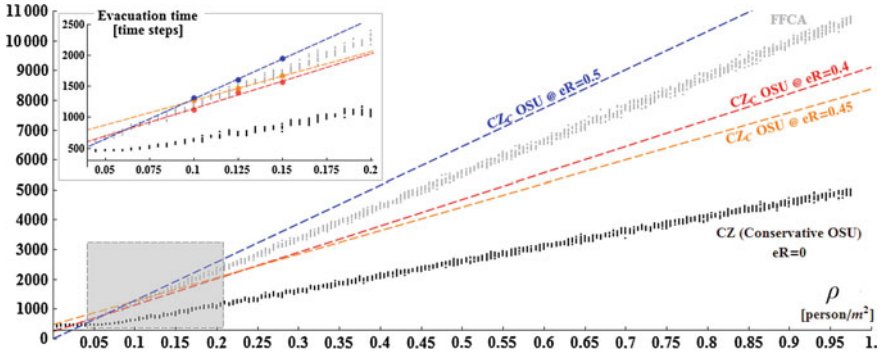

Fig. 3.16 The relationship between the density of crowd and the evacuation time. Ten trials of CZ with *conservative* crowd @ $eR = 0$ compared to the FFCA model. In addition, rough extrapolation from three simulations @ eR : 0.5, 0.45, 0.4 are shown in: *blue*, *orange* and *red*, respectively. The highlighted in *gray part* of the main plot is enlarged above

evacuation time at increasing crowd density. Moreover, the "efficiency" of agents' finding the exits can be fine-tuned with parameter eR, as shown in Fig. 3.16.

3.4.3 *Import from CAD*

Usually the final floor-plans are produced in CAD programs and have vector form as shown in Fig. 3.17(1). Regular architectural drawings contain information which may be confusing for the CS. Therefore they should be pre-processed in a CAD environment by:

- Removal of all auxiliary data such as dimensioning, door (and other opening) markings which can be interpreted as barriers;
- Filling of all obstructing objects such as furniture with a uniform solid hatch (USH);
- Filling of unaccessible areas by USH;
- Additionally, for improved legibility, it is recommended to fill the walls with USH.

Such pre-processed draft should be exported as a raster image and then uploaded to CZ. Next, such image is rescaled to the proper size as shown in Fig. 3.17(2).

As Fig. 3.17(2) indicates, the transverse dimension of this floor-plan is approx. 20 m. This is an equivalent of only 50 cells. Since it is a very coarse grid, in order to maintain maximum detail of the floor-plan, the rescaling should be "supervised". Firstly, the floor-plan is tessellated into square grid corresponding to the size of an

Fig. 3.17 **1** A CAD draft of a floor-plan of a residential building with three apartments. **2** The same floor-plan pre-processed and properly rescaled so a single pixel has the size of an agent

agent, as shown in Fig. 3.18(1). Secondly, the average value in every grid cell is calculated, as shown in Fig. 3.18(2).

Thirdly, the floor-plan is subjected to binarization at threshold t. In other words it is converted to a binary image by replacing all values below t with 0 and others with 1, as shown in Fig. 3.19(1). The best level of detail has been achieved at $t = 0.73$, that is: i. wall lines are interrupted, and ii. each room can be accessed. Nevertheless, since not every corridor and opening has approximately the same size, certain

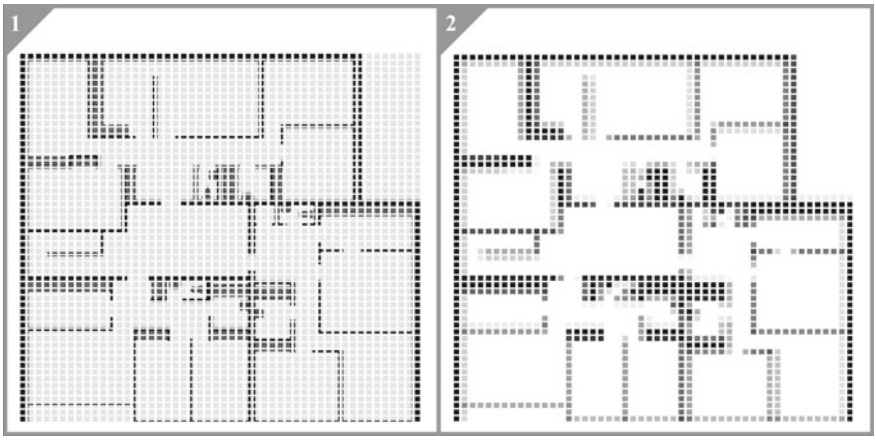

Fig. 3.18 **1** The original floor-plan tessellated into square grid corresponding to the size of an agent. **2** Averaged values in every grid cell

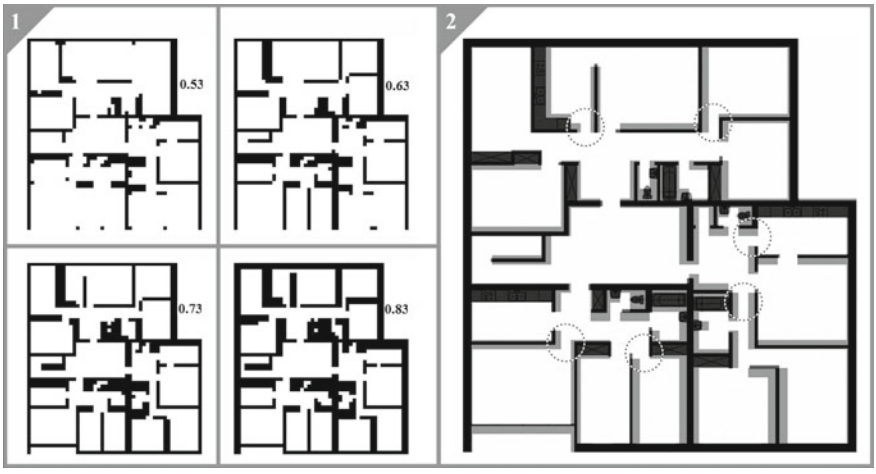

Fig. 3.19 **1** The quality of binarized floor-plan is sensitive to the value of threshold t. **2** Binarization at $t = 0.73$ produces the best detail. For comparison the result is shown as semi-transparent with the original floor-plan shown in *black*. *Circles* indicate overly narrow parts of corridors and door openings which require "manual" alterations

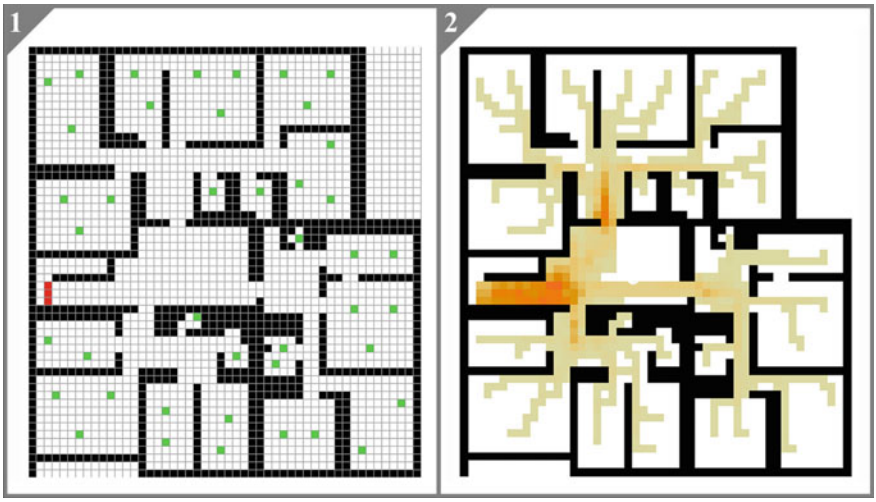

Fig. 3.20 **1** The final CS environment with "manually" adjusted floor-plan. The initial positions of agents and exit cells have been also placed "manually" and are shown in: *green* and *red*, respectively. **2** Although HM indicates certain congestion at the exit, this floor-plan seems acceptable regarding safe evacuation

"manual" correction has been applied, as shown in Fig. 3.19(2). As mentioned above, CZ supports interactive "manual" adjustments to the floor-plans. These include: i. placement of the agents, ii. placement of the exits, iii. alterations wall-to-void and vice versa. Figure 3.20(1) shows the final CS environment with "manually" adjusted floor-plan, placed agents and exit. Although the agent number is rather exaggerated for a residential building, the size of the exit is proper. Figure 3.20(2) shows the HM illustrating the evacuation paths.

3.5 Validating CZ with Three Crowd Simulations

This section presents the validation of CDC implemented in CZ by three experiments and comparison against: i. the empirical data, ii. commercial CS software, and iii. classic model described in the literature.

3.5.1 Empirical Validation: Bottleneck Evacuation

The most valuable validation of a numerical model is always the comparison with empirical data. So called *bottleneck* is a localized congestion of traffic due to a specific physical condition. The modeling and empirical experimentation of bottleneck

scenarios are classic problems of CD described in the literature. This subsection presents the quantitative validation of CZ against the empirical data presented in [20]. The experimental setup is shown in Fig. 3.21. The area beyond the bottleneck is added to potentially include so called "backward effect" caused by the interactions among the pedestrians who have already left the bottleneck and the ones still in the bottleneck.

The corresponding CZ simulation setup is shown in Fig. 3.22.

The distance $h = 4.4\,\mathrm{m}$ between the holding area and the bottleneck corresponds to 11 grid cells in the CZ setup. Initial crowd density ρ is controlled by the length of the holding area. The crowd flow J through the bottleneck is calculated as follows:

$$J = \frac{N}{\Delta t} \tag{3.2}$$

$$\Delta t = t_{\mathrm{last}} - t_{\mathrm{first}} \tag{3.3}$$

where N is the number of pedestrians, Δt is the time-gap between the last (t_{last}) and first ($-t_{\mathrm{first}}$) pedestrians crossing the bottleneck measurement line.

Figure 3.23 collects and compares the results.

As Fig. 3.23 indicates, the flow rate of the *conservative* agents at $eR = 0$ increases approximately proportionally with the width w of the bottleneck. Thus the agreement

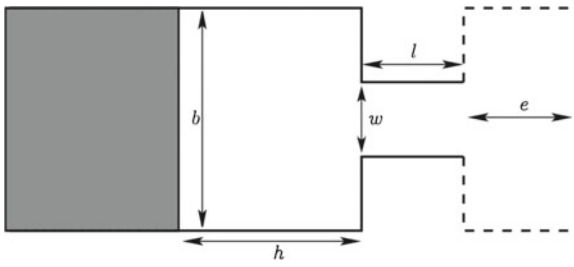

Fig. 3.21 The setup for the bottleneck experiment. Every pedestrian starts from the holding area (shown in *gray*) and proceeds to exit through the bottleneck; $b = 4\,\mathrm{m}$, $\mathrm{h} = 4.5\,\mathrm{m}$, $\mathrm{l} = 2.8\,\mathrm{m}$, $\mathrm{e} = 2\,\mathrm{m}$; w is variable

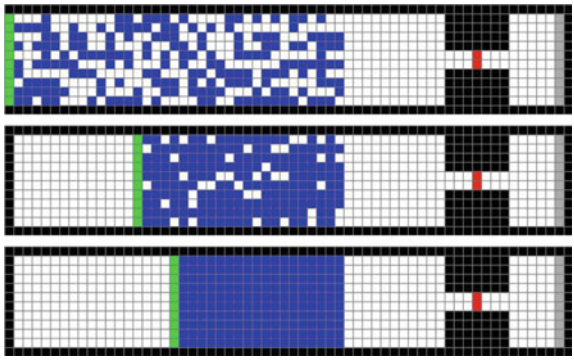

Fig. 3.22 Three examples of CZ simulation setup. *Top*, *middle* and *bottom* sub-figures correspond to: $\rho_0 = 0.5\,\mathrm{m}^{-1}$, $\rho_0 = 0.8\,\mathrm{m}^{-1}$ and $\rho_0 = 1.0\,\mathrm{m}^{-1}$, respectively. The measurement line, initial agent positions, and exit are shown in: *red*, *blue* and *gray*, respectively

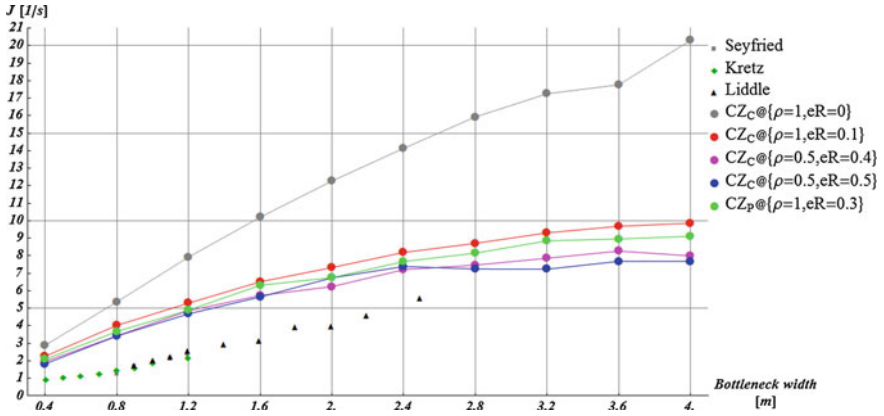

Fig. 3.23 The averaged results of ten trials of simulated crowd flow through the bottleneck with respect to its width w. $N = 180$ agents at various initial crowd densities, eR values and types of *perkiness*. Comparison with empirical data from: Seyfried [27], Kretz [16], and Liddle et al. [20]

with empirical observation is poor. However, increasing the error rate ($eR = 0.15$) for the same type of *perkiness* indicates the stagnation at certain width w of the bottleneck, which coincides with the empirical data. Figure 3.23 also shows that in principle, the crowd flow measured empirically is lower than in the simulations. Nevertheless, the simulated flow rate can be decreased by rising the eR value.

3.5.2 Case Study with PedGo

In this subsection a CS performed in CZ is compared with PedGo—a commercially available program evacuation simulations. The simulation environment is a floor-plan of school similar in size to the empirical study described in [13]. The layout has been imported from a CAD program and 600 agents in three equal groups have been distributed in different parts of the building. Their objective is to proceed to the closest exit, as shown in Fig. 3.24.

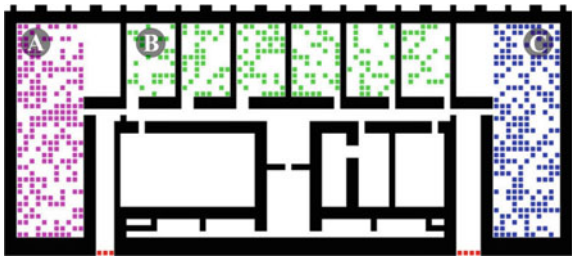

Fig. 3.24 *Magenta*, *green* and *blue* indicate the agents assigned to sections: A, B and C, respectively. Exit and wall cells are indicated by *red* and *black*, respectively

Figures 3.25, 3.26, and 3.27 show the screenshots of two CZ panels displaying the local velocities and HMs of three types of agents: *lazy*, *conservative* and *perky*, respectively.

500 trials for six setups: *lazy*, *conservative* and *perky* agents at OSU and $eR = 0$ and $eR = 0.25$ have been performed in CZ. According to Ref. [24] the standard pedestrian velocity is 1.2 m/s. In CZ simulation, the maximum velocity at one time-step is one grid cell, that is 0.4 m. Thus three discrete time-stems are equivalent of 1s. Figure 3.28 presents the quantitative analysis of the evacuation time in comparison to the PedGo model.

As Fig. 3.28 indicates, in principle the CZ-modeled crowd @ $eR = 0$ evacuate form the school building almost two times faster than the PedGo crowd. Interestingly, in the experiment presented in [13], the schoolchildren also "outperformed" the "standard simulation agents" by a similar factor. Nonetheless, the evacuation time of the CZ crowds can be fine-tuned by proper combination of the *perkiness* and

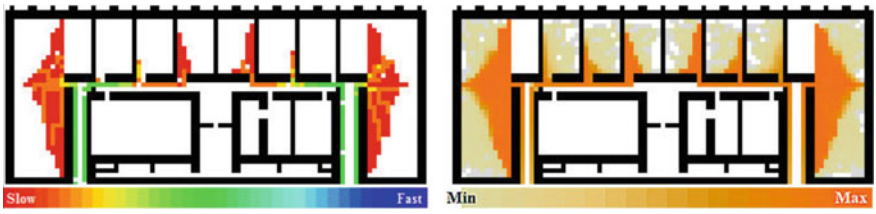

Fig. 3.25 The screenshots of CZ panels at the 60th time-step of the simulation of the evacuation of *lazy* agents. On the *left* and *right*: the positions with visualized local velocities of the agents and the HM, respectively

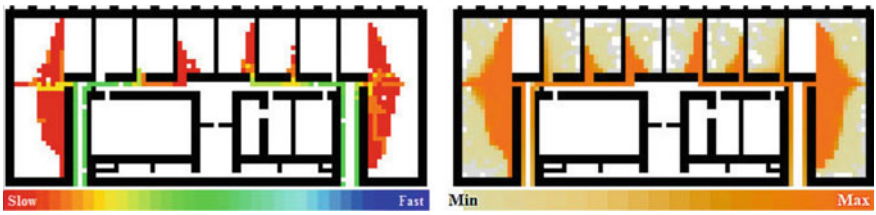

Fig. 3.26 The 60th time-step of the CS with *conservative* agents. The convention as in Fig. 3.25

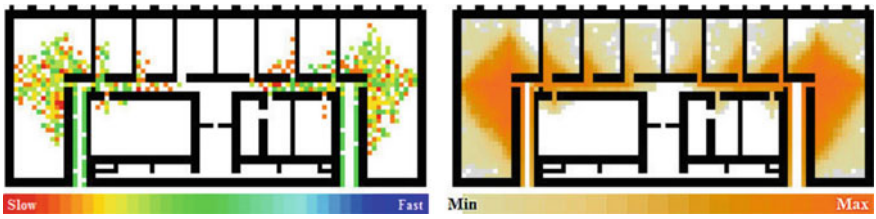

Fig. 3.27 The 60th time-step of the CS with *lazy* agents. The convention as in Fig. 3.25

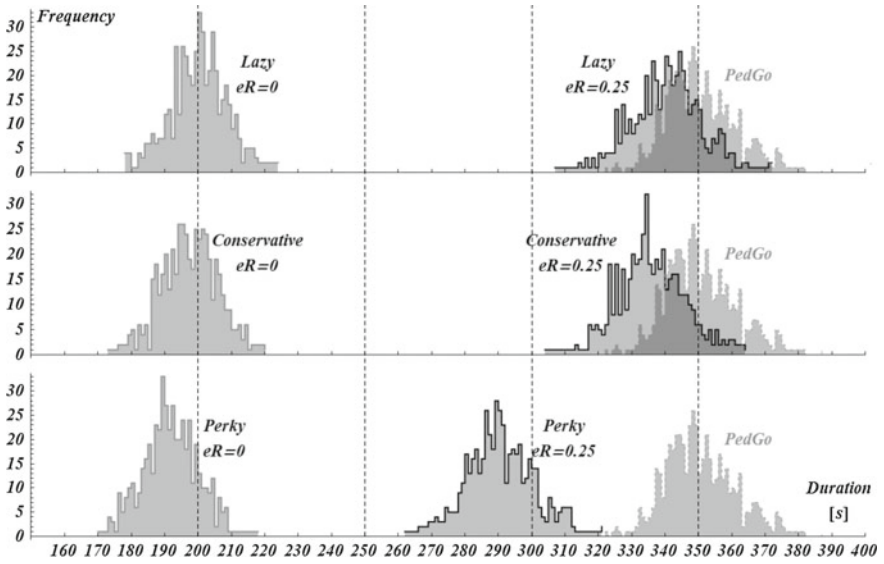

Fig. 3.28 Histograms of evacuation times of 500 trials of the school evacuation modeled by CZ and PedGo. In the *top*, *middle* and *bottom*: *lazy*, *conservative*, and *perky* agents, respectively. CZ set to the *lazy* crowd at $eR = 0.25$ shows the best quantitative agreement with PedGo

parameter eR. Figure 3.28 also shows that setting these parameter values to match the results produced by PedGo is relatively straightforward. Perhaps it is possible by adjusting the eR parameter to qualitatively emulate the PedGo model with other types of *perkiness*, that is *lazy* and *conservative*. In such cases, however, the histograms, that is quantitative distributions may not match. Furthermore, the sensitivity of crowds of different type of *perkiness* to the eR parameter varies. It is rather intuitive, that since random component is the highest in the behavior of *perky* crowd, it is the least affected by the eR parameter. On the other hand, the eR parameter alters the behavior of the *lazy* crowd the most heavily, since the movement of this type of agents is the most stringent.

3.5.3 Bottleneck Evacuation Study from the Literature

Two alternative update schemes are implemented in CDC: "random shuffled update" (RSU) and "ordered sequential update" (front to back) (OSU). In the experiment described in this subsection, the influence of the update scheme on the overall crowd behavior has been investigated. The results have been compared to [23]. In this CS the crowd of 200 agents evacuates from a 25×25-cell room through a 3-cell narrow exit being effectively a bottleneck. The locations of agents at the 60th and 100th time-step are shown in Fig. 3.29.

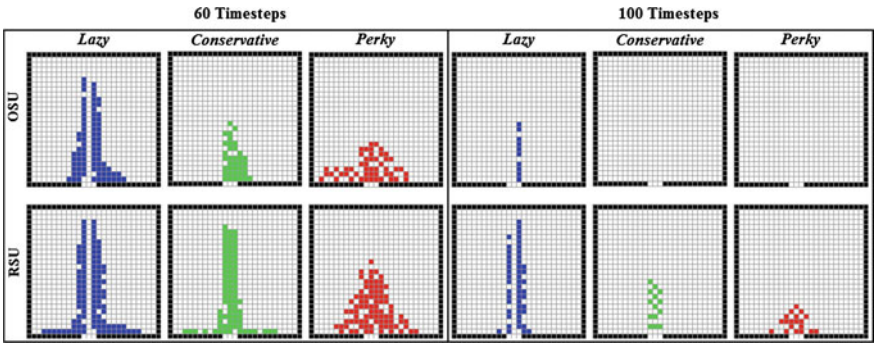

Fig. 3.29 The screenshots for crowds of three types of *perkiness* at the 60th and 100th time-step at the update schemes: OSU and RSU are shown in: the *top* and *bottom*, respectively. *Perky*, *conservative* and *lazy* crowds are indicated by *red*, *green* and *blue*, respectively

As Fig. 3.29 indicates, in the case of homogeneous crowd, the evacuation at OSU of *conservative* agents is quicker than *perky* agents. The evacuation of the *lazy* crowd takes the longest time and shows the same queuing behavior as in [23]. This clearly reflects the *lazy* movement strategy as they "would rather wait in queue than move unnecessarily". Increasing the value of eR further prolongs the evacuation time. Figure 3.30 shows that by simple adjustment of eR the evacuation time can be extended to match the results presented in [23].

Figure 3.30 also shows agreement with intuition, that more ordered behavior (e.g. *conservative* agents @ OSU) helps to evacuate the room more efficiently. The same type of agents @ RSU move out of the room slightly slower than the mixed crowd and comparably to the *perky* agents.

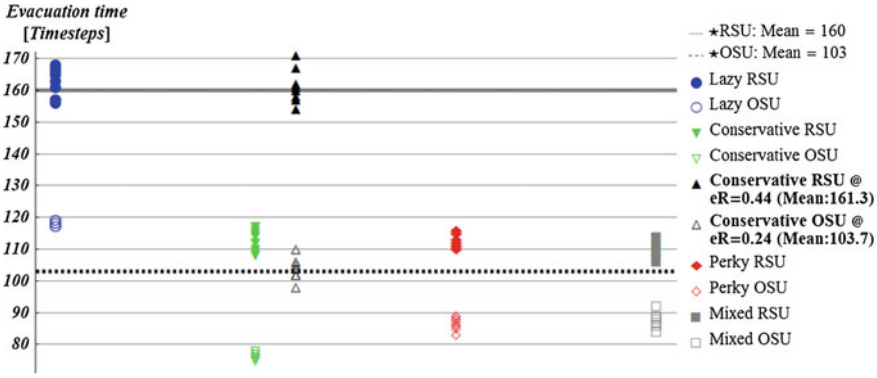

Fig. 3.30 10 trials of experiments shown individually for each combination of *perkiness* and update scheme. In the legend, the results from [23] are indicated with *stars*. In addition, the results for equinumerous mixes of *perky*, *conservative* and *lazy* agents are shown in *gray*

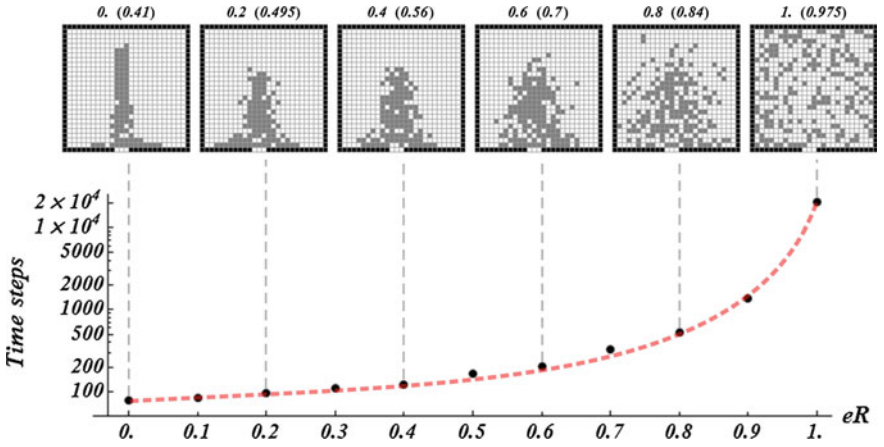

Fig. 3.31 The evacuation time of the *conservative* crowd @ RSU from the room at increasing eR. On the *bottom*: the log-plot relating the value of eR to the number of time-steps required for the evacuation. On the *top*: the screenshots of the room at the 60th time-step at given eR. The value of eR is displayed above each corresponding sub-figure. The rate of remaining agents is shown in *brackets* next to the value of eR

The next experiment investigates the effect of eR parameter on the quantitative behavior of crowd. A series of simulations of the same scenario has been performed at increasing value of eR. The results are shown in Fig. 3.31.

As Fig. 3.31 indicates, the evacuation time increases with increasing eR. For approx. $eR > 0.5$ the evacuation rate decreases immensely.

3.6 Conclusions

The main contributions of Crowd-Z (CZ):

- It is a user-friendly, highly compatible and practical crowd simulation (CS) framework.
- It is highly intuitive and can be employed by users unfamiliar with CS.
- The CZ user interface works smoothly and dynamically displays the CSs.
- CZ supports dynamic interaction with the users by a few intuitive parameters and "manual" input.
- Most of the simulations presented here are illustrated with interactive demonstrations available at [32].
- CZ records the complete information on individual agents e.g.: mean and instantaneous velocity, walking path, etc. Such data can be used for all types of qualitative and quantitative analysis.

The future challenges include:

- Although the "crowd dynamics component" (CDC) of CZ models CS rather satisfactorily, it should be improved for more realistic crowd modeling.
- Additional attributes can be assigned to individual agents. Investigation of the influence of such realistic factors like: tiredness, age, social hierarchy, etc. on the collective crowd behavior seems interesting. Moreover, this would allow for the more subtle analysis of crowds including social priority, (spontaneous) grouping etc.

References

1. Bandini S, Vizzari G (2007) Regulation function of the environment in agent-based simulation. In: Environments for multi-agent systems III, Springer, pp 157–169
2. Blue V, Adler J (1999) Cellular automata microsimulation of bidirectional pedestrian flows. Transp Res Rec J Transp Res Board 1678:135–141
3. Blue VJ, Adler JL (2002) Flow capacities from cellular automata modeling of proportional splits of pedestrians by direction. In: Pedestrian and evacuation dynamics, pp 115–122
4. Burstedde C, Klauck K, Schadschneider A, Zittartz J (2001) Simulation of pedestrian dynamics using a two-dimensional cellular automaton. Physica A 295(3):507–525
5. Chavey D (1989) Tilings by regular polygons II: A catalog of tilings. Comput Math Appl 17(1–3):147–165
6. Chraibi M, Wagoum AUK, Schadschneider A, Seyfried A (2011) Force-based models of pedestrian dynamics. NHM 6(3):425–442
7. Gipps PG (1987) Simulation of pedestrian traffic in buildings. Technical report
8. Helbing D, Molnar P (1995) Social force model for pedestrian dynamics. Phys Rev E 51(5):4282
9. Hirai K, Tarui K (1977) A simulation of the behavior of a crowd in panic. Syst Control, Japan
10. Kepler J (1938) Harmonice mundi (linz, 1619). English edition: Harmonies of the world, Book 5
11. Kerridge J, Hine J, Wigan M (2001) Agent-based modelling of pedestrian movements: the questions that need to be asked and answered. Environ Plann 28(3):327–342
12. Kirchner A, Schadschneider A (2002) Simulation of evacuation processes using a bionics-inspired cellular automaton model for pedestrian dynamics. Physica A 312(1):260–276
13. Klüpfel H, Meyer-König T, Schreckenberg M (2003) Comparison of an evacuation exercise in a primary school to simulation results. In: Traffic and Granular Flow?01, Springer, pp 549–554
14. Kretz T (2009) Pedestrian traffic: on the quickest path. J Stat Mech Theory Exp 2009(03):P03012
15. Kretz T, Schreckenberg M (2006) The fast-model. In: Cellular Automata, Springer, pp 712–715
16. Kretz T, Grünebohm A, Schreckenberg M (2006) Experimental study of pedestrian flow through a bottleneck. J Stat Mech: Theory Exp 2006(10):P10014
17. Kretz T, Hengst S, Vortisch P (2008) Pedestrian flow at bottlenecks-validation and calibration of vissim's social force model of pedestrian traffic and its empirical foundations. arXiv:08051788
18. Kretz T, Bönisch C, Vortisch P (2010) Comparison of various methods for the calculation of the distance potential field. In: Pedestrian and evacuation dynamics 2008, Springer, pp 335–346
19. Kretz T, Große A, Hengst S, Kautzsch L, Pohlmann A, Vortisch P (2011) Quickest paths in simulations of pedestrians. Adv Complex Syst 14(05):733–759
20. Liddle J, Seyfried A, Klingsch W, Rupprecht T, Schadschneider A, Winkens A (2009) An experimental study of pedestrian congestions: influence of bottleneck width and length. arXiv:09114350
21. Löhner R (2010) On the modeling of pedestrian motion. Appl Math Model 34(2):366–382

22. Rajewsky N, Santen L, Schadschneider A, Schreckenberg M (1998) The asymmetric exclusion process: Comparison of update procedures. J Stat Phys 92(1–2):151–194
23. Rogsch C, Schadschneider A, Seyfried A (2009) Simulation of human movement by cellular automata models using different update schemes. In: Proceedings of the 4th International Symposium on Human Behaviour in Fire 2009, 13–15 July 2009, Interscience Communication Ltd, London, 2009. ISBN: 978-0-9556548-3-1. -S. 543–548. http://juser.fz-juelich.de/record/6763, record converted from VDB: 12.11.2012
24. Schadschneider A, Klingsch W, Klüpfel H, Kretz T, Rogsch C, Seyfried A (2009) Evacuation dynamics: empirical results, modeling and applications. In: Encyclopedia of complexity and systems science. Springer, pp 3142–3176
25. Schneider V, Könnecke R (2001) Simulating evacuation processes with aseri. In: Schreckenberg M, Sharma, SD (eds) Pedestrian and Evacuation Dynamics, pp 301–313
26. Schultze-Naumburg P (1909) Kulturarbeiten. band iv: Städtebau. 2. Auflage München
27. Seyfried A, Passon O, Steffen B, Boltes M, Rupprecht T, Klingsch W (2009) New insights into pedestrian flow through bottlenecks. Transp Sci 43(3):395–406
28. Shi J, Ren A, Chen C (2009) Agent-based evacuation model of large public buildings under fire conditions. Automat Constr 18(3):338–347
29. Weidmann U, Weidmann U, Weidmann U, Weidmann U (1993) Transporttechnik der Fussgänger: Transporttechnische Eigenschaften des Fussgängerverkehrs (Literaturauswertung). ETH, IVT
30. Willard S (2004) General topology. Courier Corporation
31. Yu W, Chen R, Dong L, Dai S (2005) Centrifugal force model for pedestrian dynamics. Phys Rev E 72(2):026112
32. Zawidzki M (2014) Interactive demonstrations of Crowd-Z. http://zawidzki.com/Crowd-Z/, an interactive demonstration

Chapter 4
The Influence of Various Factors on Crowd Behavior

Abstract This chapter presents a number of experiments performed in Crowd-Z (CZ). The objective is to investigate the influence of various factors on the collective behavior of the crowd. These factors include: the type of metric & neighborhood, agents' *perkiness* and type of tessellation. The robustness of models based on different tessellations are examined by grid rotation. Three platonic tessellations, that is: hexagonal, triangular and square are compared. Two experiments based on the literature are presented:

1. The egress of 200 agents from a square room (SRE) in grid rotated by: 0°, 15°, 30°, and 45°.
2. The one-directional flow (ODF) in grid rotated by: 15°, 18.434 . . .°, 26.565 . . .°, 30°, and 45°.

The basic CZ setup is used for the experiments: the crowds are homogeneous, and the agents' walking speed is unitary and constant. The qualitative analysis is based in heat maps. The quantitative analysis is based on density-flow rate and evacuation time diagrams.

4.1 Introduction

The main objective of CSs in architectural floor-plans is the examination of the evacuation safety. Therefore the most natural crowd behavior can be described as an unordered flow toward the nearest exit. Thus in a properly modeled CS:

- the agents proceed to the nearest exits;
- the movement of agents is scattered, in other words queuing behavior is undesired;
- the walking paths of the agents should be consistent despite rotation of the grid;
- the density-flow rate should not be disturbed by rotation of the grid.

In the discrete spatiotemporal agent-based modeling there are a number of factors determining the collective behavior of crowds. The fundamental factors include:

- Tessellation type, e.g.: square, triangular, hexagonal.
- Metric type, e.g.: Euclidean, "Taxicab", Maximum.

M. Zawidzki, *Discrete Optimization in Architecture*, SpringerBriefs
in Architectural Design and Technology, DOI 10.1007/978-981-10-1106-1_4

- Neighborhood range and type. e.g.: von Neumann's, Moore's.
- Type of agents' *perkiness*, that is willingness to move.
- The degree of randomness in the agents' movement.
- The update scheme, etc.

These are only the most fundamental factors. More subtle parameters such as individual walking speeds, proxemics, etc. are not considered in this book. For a brief study of the last factor listed above, that is the influence of randomness on the crowd behavior see Sect. 3.5.3. This chapter begins with a preliminary study of the agents' *perkiness*. The subsequent sections describe systematical study of the robustness of grid types examined by rotation of the grid.

4.2 Experiment 1: The Influence of Metric and Agent *Perkiness* on the Crowd Behavior

This experiment compares qualitative behavior of *perky*, *conservative* and *lazy* agents. The experiments are based on three potential distance fields (DFs) based on: von Neumann (*vN*), Moore with Maximum (*MM*) and Moore with Euclidean (*ME*) metrics, as shown in Fig. 4.1. In this visualization the three combinations of metrics and neighborhoods correspond to the positions framed in red in Fig. 3.9. For brief descriptions of metrics, neighborhoods and DF see Sect. 3.3.

In this experiment "random shuffled update" (RSU) has been used [9]. Further on, the term "setup" refers to the combination of: the neighborhood type, the metric and the agents' *perkiness*. The error rate (eR) has been set to 0, which means that the agents' movement is extremely (unnaturally) efficient.

A relatively free-form architectural layout has been selected for this CS. Namely, the floor-plan of Saint Paul's Church (SPC), Frankfurt am Main, Germany has been appropriated. The historical plan from 1896, which is readily available from the

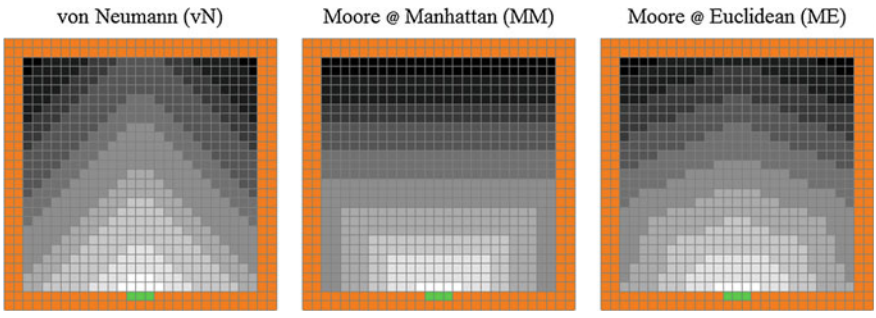

Fig. 4.1 Visualization of three DFs. The exits and walls are indicated in *green* and *orange*, respectively. The closer to the exits the lighter are the cells. Posterization has been applied to improve legibility

Internet has been imported to CZ as a raster image. Figure 4.2 shows the layout discretized to a 40 × 40 cm square grid. Initially, 1370 agents occupy all available seats.

Figure 4.3 shows the heat maps (HMs) for three types of: agents (*lazy*, *conservative* AND *perky* and DFs (vN, MM & ME). As Fig. 4.3 indicates, in this qualitative evaluation the most natural in this experiment are: Euclidean metric, Moore's neighborhood and *perky* strategy of the agents.

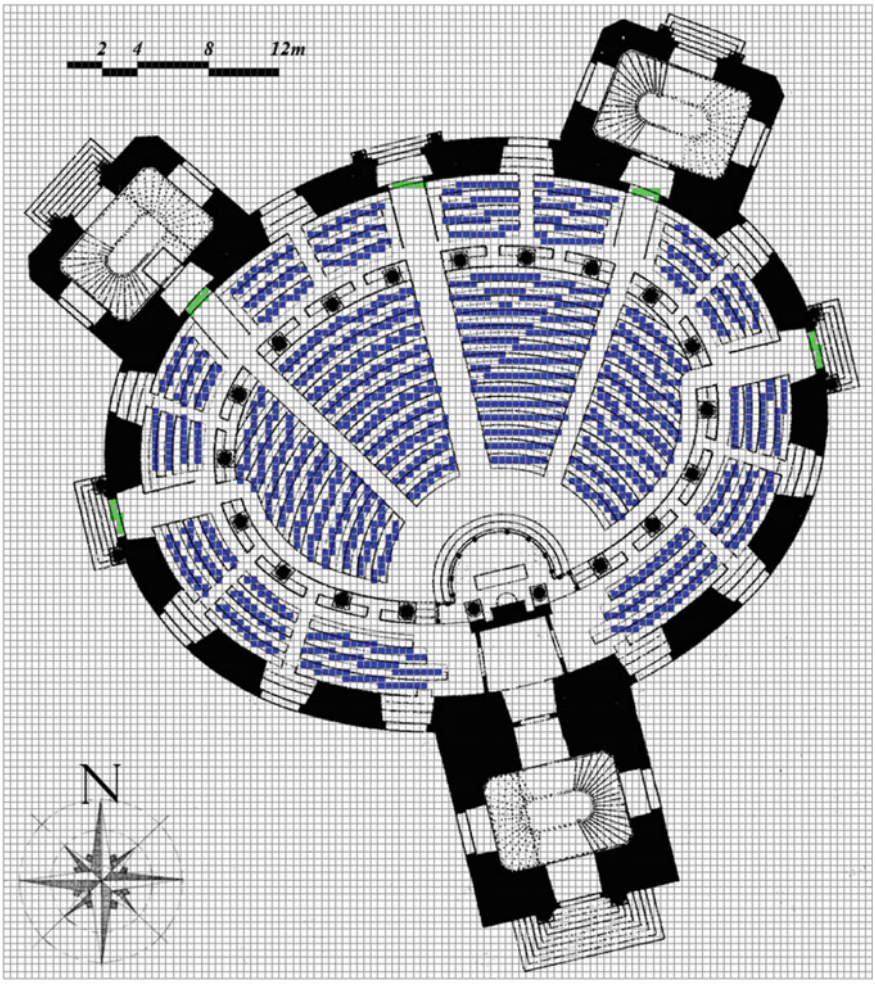

Fig. 4.2 The floor-plan of the St. Paul's Church, Frankfurt am Main (version from 1896). 1370 agents placed on 40 × 40 cm grid and exits are shown in *blue* and *green*, respectively

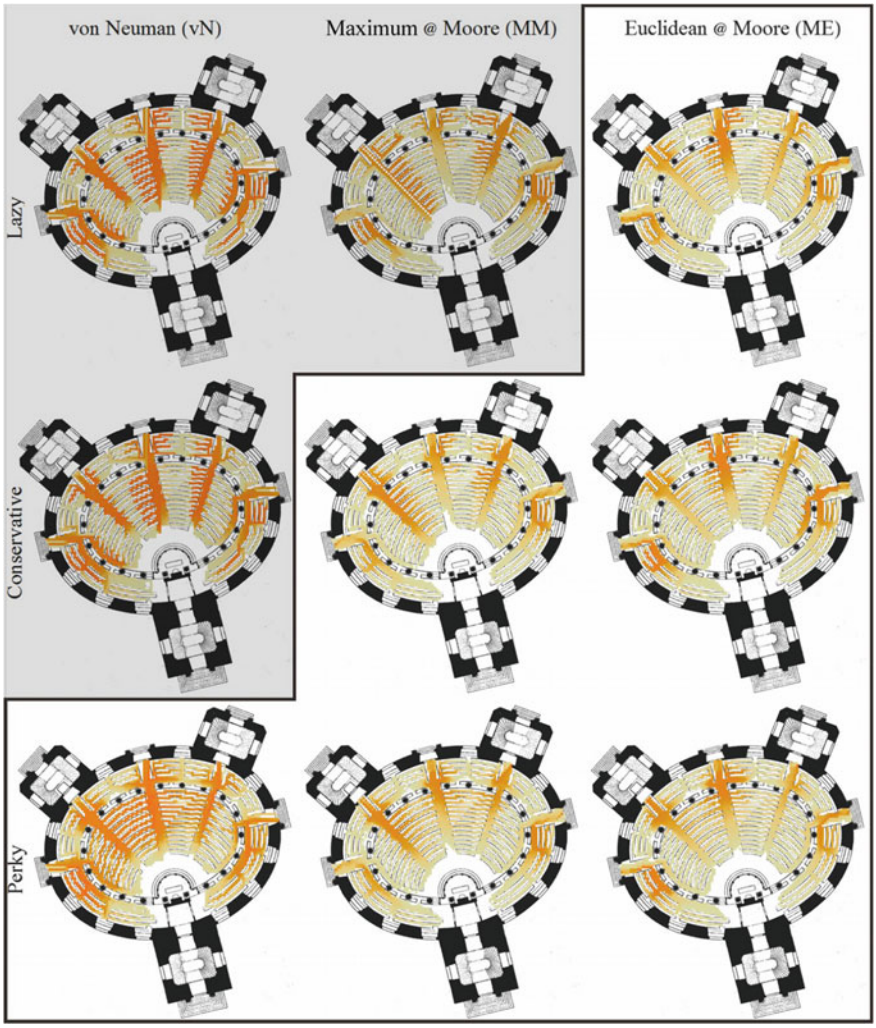

Fig. 4.3 HMs of egress of 1370 agents from SPC at various setups and agents' *perkiness*. HMs with less natural evacuation pattern are shown on the gray background

4.3 Three Regular Tessellations

Most of rule-based crowd simulations (CSs) in discretized space employ 0.4×0.4 m square grid. In many architectural designs, however, the layouts can not be properly described by multiples of the 40 cm unit [4]. An early investigation of CS in hexagonal tessellation [7] have shown that the critical density of square-grid model is lower than the hexagonal-grid model. Moreover, in urban scale CSs a larger grid of 0.75×0.75 m is recommended [3].

In the following sections Crowd-Z (CZ) is used to analyze the agents' behavior in three regular tessellations, that is square, triangular and hexagonal.

4.3.1 CS Setups

The following simulations employ the basic setup of CZ. It is an agent based model (ABM) with every pedestrian represented as an autonomous agent with its own "knowledge" and objectives [11].

For the review of ABM systems for CSs see [1, 3, 13]. Alike the previous Chapter, the agents' movement is determined by static potential distance field (DF) [2, 12]. DFs based on three regular tessellations are computed once for every floor-plan as described in the sub-section below.

4.3.2 DFs in Square, Triangular and Hexagonal Grid

DFs tabulates the shortest distances from every position in grid to the exits. Most of DF models adopt the shape of a square cell. However, such simplification has a number of consequences:

- Simulation of a movement in an oblique direction becomes difficult or unnatural.
- The diagonal walk in von Neumann neighborhood (vN) requires two steps and a right angle turn, which is also very unnatural.
- The diagonal displacement in Moore neighborhood (M) differs from the orthogonal displacement, which results in different velocities.

Although in classic crowd simulations space is discretized into square grid, recently hexagonal grid has gained increased popularity [6]. The simulations in square grids have been carried out in three types of DFs based on: von Neumann (vN), Moore with Maximum (MM) and Moore with Euclidean (ME) metrics, triangular (T) and hexagonal (H) grids, as illustrated in Fig. 4.4.

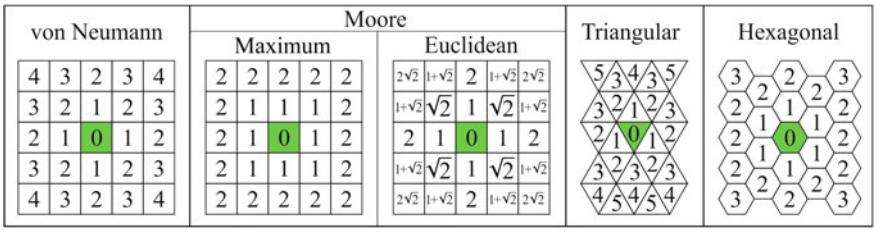

Fig. 4.4 The calculation of distance in five regular grid setups. The exit is placed in the *center* of each DF and the corresponding value is 0

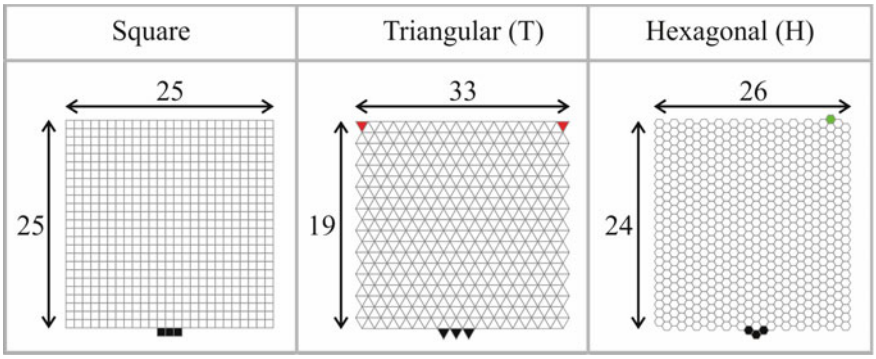

Fig. 4.5 The space of SRE discretized by *square*, *triangular* and *hexagonal* grid. The number of cells in both dimensions are given for each grid. Each grid has the same number of cells. T & H have been adjusted by removing two cells, and adding one cell, shown in *red* and *green*, respectively. The exit cells are shown in *black*. The space for SRE has been discretized by three grids with the same number of cells. Each cell has the size of an agent and 1/625 of the entire area. The geometrical properties, such as dimensions of cells are not relevant in the actual simulations which are based solely on adjacency and neighborhood

4.4 Experiment 2: Square Room Evacuation (SRE)

This experiment extends the CS described in the previous Sect. 3.5.3. It replicates the scenario investigated in [10], that is the egress of 200 agents from a 25 × 25-cell room through a three-cell exit. In addition to the previous experiments, two other regular tessellations (triangular & hexagonal) have been introduced and the grid has been rotated by three angles: 15°, 30°, and 45° relative to the layout.

The size of the original layout is 25 × 25 cells, thus it consists 625 "walkable" cells. The quantization of space in T and H have been adjusted to maintain the original proportions and sizes as closely as possible: 33 (transverse) × 19 (longitudinal) 2 = 625 walkable cells and 26 × 24 + 1 = 625 walkable cells), respectively. Figure 4.5 shows the space quantization and adjustments for T and H grids.

4.4.1 Qualitative Analysis: Heat Maps for SRE

Figures 4.6, 4.7 and 4.8 collect HMs for *lazy*, *conservative*, and *perky* agents at three square setups (vN, MM and ME), and triangular (T) & hexagonal (H) grids, respectively.

As Figs. 4.6, 4.7 and 4.8 indicate, both strategy and grid type have strong influence on agent's behavior. vN(L), MM(L), T(L), T(C) and H(L) demonstrate the same queuing phenomenon as in [10]. Two further observations can be made:

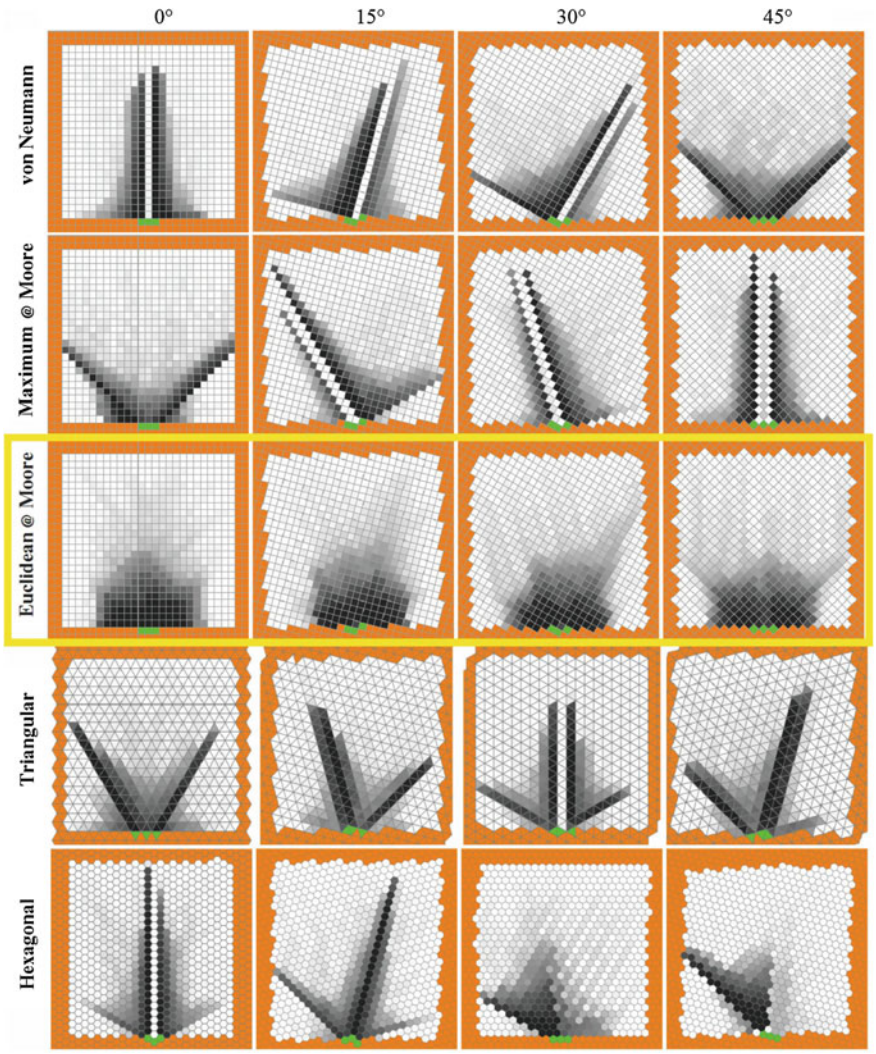

Fig. 4.6 HMs illustrating the behavior of *lazy* agents in regular grids at four different angles. The walls and three exits are shown in *orange* and *green*, respectively. The best setup is framed in *yellow*

1. The behavior of all three types of agents, that is *lazy* (L), *conservative* (C), and *perky* (P) in square grid with Moore's neighborhood and Euclidean metric (ME) is similar, and seems natural.
2. The behavior of *perky* (P) agents is similar in all grid types and also seems the most natural.

As above indicate, the behavior of agents is highly influenced by the model used. This is also reflected in quantitative data presented in the following subsection.

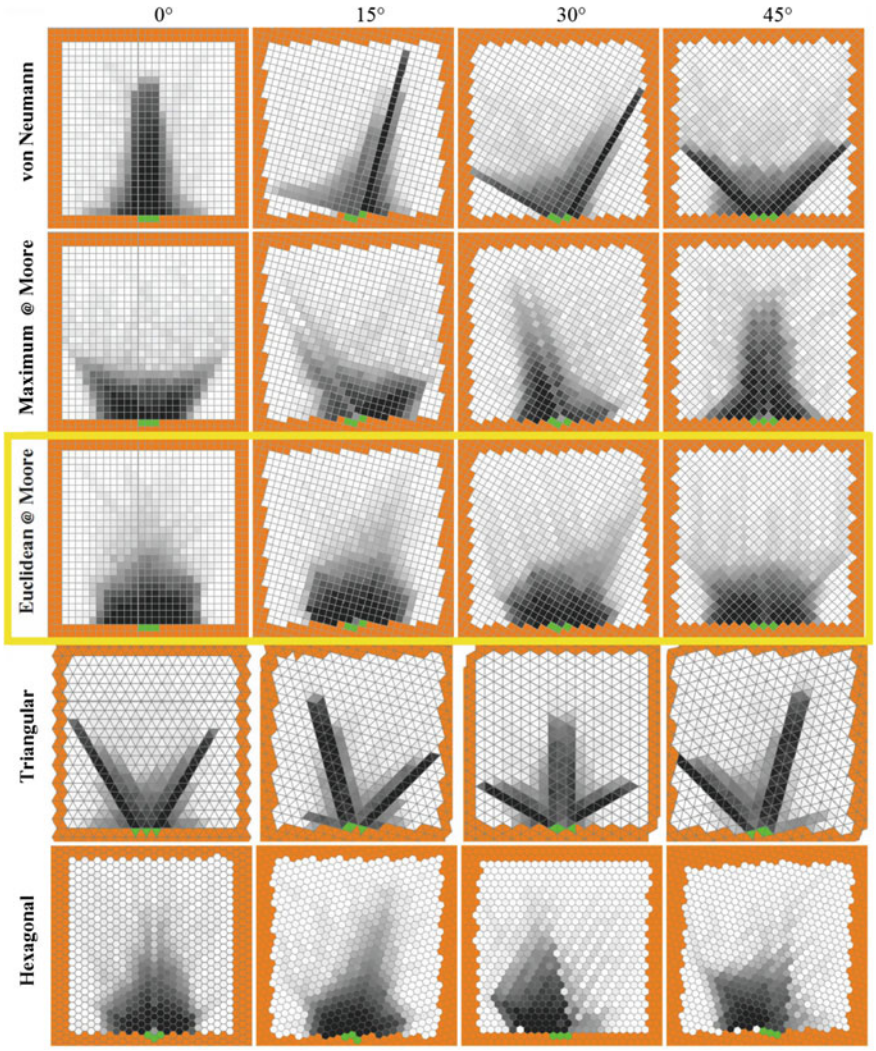

Fig. 4.7 HMs illustrating the behavior of *conservative* agents in regular grids at four different angles. The best setup is framed in *yellow*

4.4.2 Quantitative Analysis: Evacuation Time for SRE

Figure 4.9 collects HMs which illustrate agents' behavior in all setups in this experiment. The crowd density (ρ) is 3.125 agents per m^2, that is 50 % of the maximum. The values are averaged from 10 trials.

As Fig. 4.9 indicates, in the considered scenario the type of agents seem not to influence substantially the evacuation time for MM and ME. Moreover, H produces

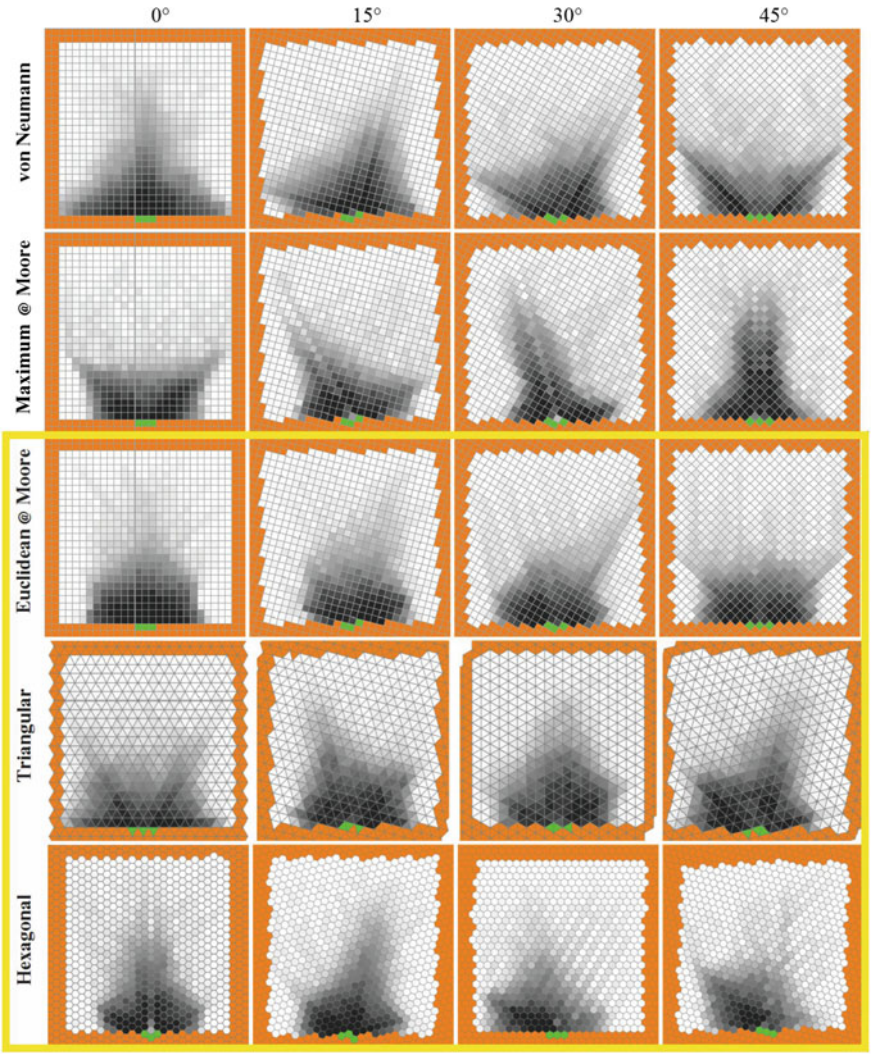

Fig. 4.8 HMs illustrating the behavior of *perky* agents in regular grids at four different angles. Relatively natural setups are framed in *yellow*

very similar results to MM and ME. In fact, H(C) clear the space the fastest; H(P) is the second fastest, but H(L) is slower than MM and ME. On the other hand, vN (von Neumann neighborhood in square grid) and T (triangular grid) produce similar results.

This suggests that H, MM and ME model architectural space in this scenario rather naturally, while both T and vN seem rather unnatural. As Fig. 4.9 also indicates, vN is the most sensitive to the grid rotations, regardless of agent *perkiness*. MM,

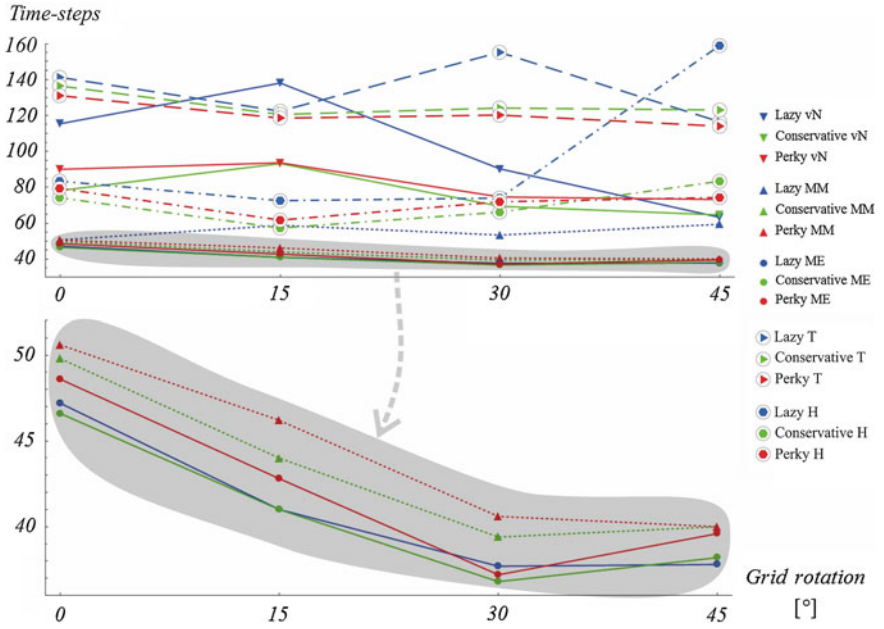

Fig. 4.9 The average evacuation time-steps for all agent types, DFs (setups), and grid rotations. For clarity, the *bottom* sub-figure is a magnification of the lower part of the main *plot* (*on the top*)

T & H with *conservative* and *perky* agents, and ME regardless of agent *perkiness* are relatively consistent.

4.5 Experiment 3: One-Directional Flow (ODF)

One-directional pedestrian flow through inclined pathways has been investigated in [5]. The inclinations were: 0°, 18.434...°, 26.565...° and 45°. The advanced CS model used there also implemented the variable velocity, "personal space", etc. The quantitative analysis, namely density-flow rate and density-velocity rate have indicated good agreement with the "Japanese Public Guideline for Evacuation". However, certain discrepancy in the flow rate among inclinations has occurred. It has been attributed to "*the difference between the directional vectors of pedestrians and the direction of pathways model*". The inclinations investigated there, however, are not the most representative for real floor-plans, as they are directly result from a square grid as illustrated in Fig. 4.10.

These azimuthal angles α_N can be expressed as follows:

$$\alpha_N = ArcTan\left(\frac{1}{N}\right) \tag{4.1}$$

where $N \in \mathbb{N}^*$.

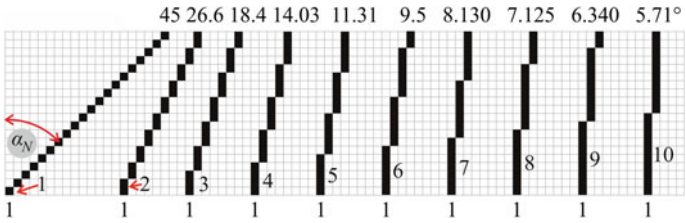

Fig. 4.10 Azimuthal inclinations constructed directly from the square grid. The values on the *top* are given in degrees

For $N = \{1, 2, 3, 4, ...\}$ such procedure generates the following angles: {45°, 26.565...°, 18.434...°, 14.036...°, ...}. Except for 1, every value of the corresponding angle α_N is an irrational number.

On the other hand, architectural layouts traditionally employ rational angles, such as: 15°, 18° (for pentagon), $\frac{45}{2}°$, 30°, $\frac{270}{7}°$ (for heptagon), 45°, etc. Thus the applicability of those irrational angles for realistic layouts is very limited. Therefore the CSs described here has also been performed for two rational angles: 15° and 30°.

4.5.1 Qualitative Analysis: Heat Maps for ODF

The crowd density (ρ) has been set to 3.125 $\frac{agents}{m^2}$, which is half of the theoretical maximal value mentioned in Sect. 3.4.1. Figure 4.11 shows the HMs illustrating the crowd behavior in three setups in square grid: vN, MM & ME.

As Fig. 4.11 indicates, in none of the square setups crowd moves naturally at all inclinations. However, ME is the least affected by the grid rotation, particularly with *perky* agents. In this model of CS, the crowd moves in natural way only at: 0° and 45° inclinations. Other angles distort the movement causing visible asymmetries. ME is relatively the most natural, however, even in this setup the crowd tends to "adhere" to one wall. This asymmetry in the movement is more evident at smaller angles in ME and vN setups. On the other hand, it is more evident for wider angles in MM setup. Interestingly, HMs also show certain anti-symmetry between vN and MM setups. Figure 4.12 shows the HMs for triangular (T) and hexagonal (H) grids.

As Fig. 4.12 indicates, the behavior of agents in T and H grids is also highly influenced by the inclination angle. However, in both cases *conservative* and *perky* agents behave similarly. They are also less affected by the grid rotation, as they use most of the exits regardless of the inclination. In this simple crowd model, the agents in T and H grids move naturally only at 0° and 30° inclinations. Also in T grid at 26.565...° which is very close to 30°. At other inclinations, also a certain asymmetry is evident.

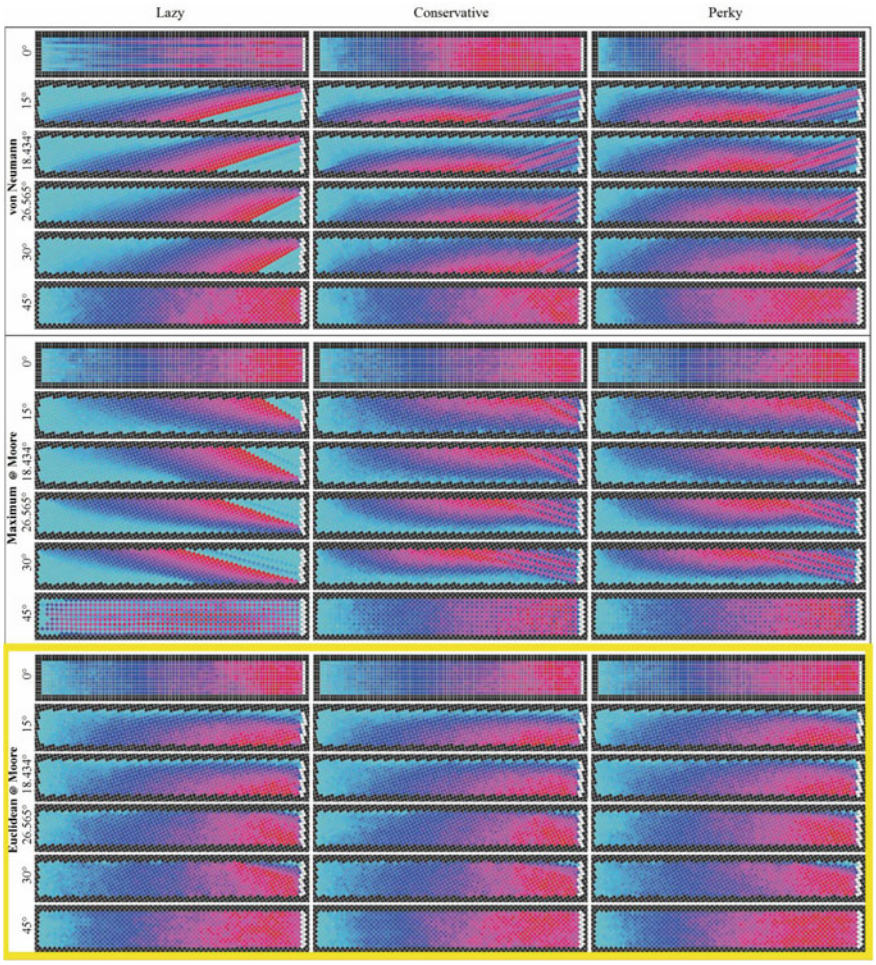

Fig. 4.11 HMs for all inclinations and *square* setups. *White* indicates the exit cells. The most natural setups are framed in *yellow*

4.5.2 Quantitative Analysis: Density-Flow Rate Diagrams for ODF

Density-flow rate diagrams for all types *perkiness*, setups and inclinations are shown in Fig. 4.13.

In so the "free-walking state" the interferences among agents are insignificant, thus the increase of the number of agents directly increases the density-flow rate. In a higher crowd density, that is in the "crowd-flow state", the interferences among

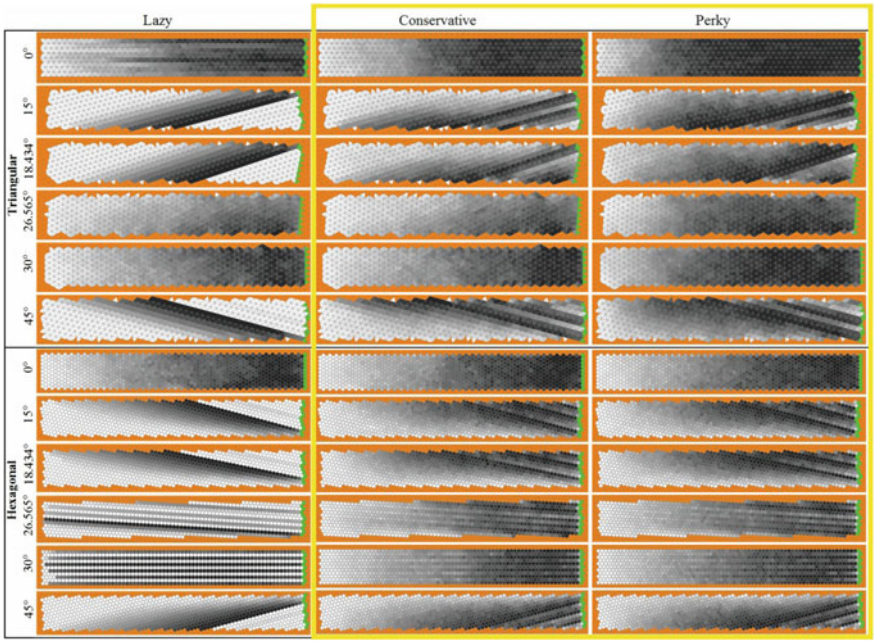

Fig. 4.12 HMs for triangular (T) and hexagonal (H) grids with all types of *perkiness* at all inclinations. The exits and walls are shown in *green* and *orange*, respectively. *Gray-scale* is the color function. Relatively natural setups are framed in *yellow*

agents impede the flow, so increasing the number of agents does not directly increase the density-flow rate.

As Fig. 4.13 indicates, in all setups the density-flow rate reflects, to various degrees, the natural transition from: "free-walking state", when the flow-rate grows proportionally to ρ, to the "crowd-flow state", when the flow rate is approximately constant. As Fig. 4.11 indicates, the *perky* and *conservative* crowds in MM do not behave particularly naturally. However, these are the most consistent setups regarding density-flow rate. Moreover, H setup with *perky* and *conservative* crowd shows similar consistency. Interestingly, in ME setup, the agents' *perkiness* seems not to affect the density-flow rate. Figure 4.13 also indicates that this CS model produces exaggerated values of density-flow rates. However, as demonstrated in Sect. 3.5.2, the flow-rate can be reduced by adding stochasticity to the agents movement by setting the value $eR > 0$. For square and hexagonal grids, in most setups, density-flow rates for the inclinations derived from a square grid: 18.434 . . .° and 26.565 . . .° are very similar to the rational angles: 15° and 30°. It is not the case for the triangular grid. Most importantly, HMs do not indicate substantial differences between these inclinations. This suggests that the findings presented in [5] would hold for any arbitrary angle.

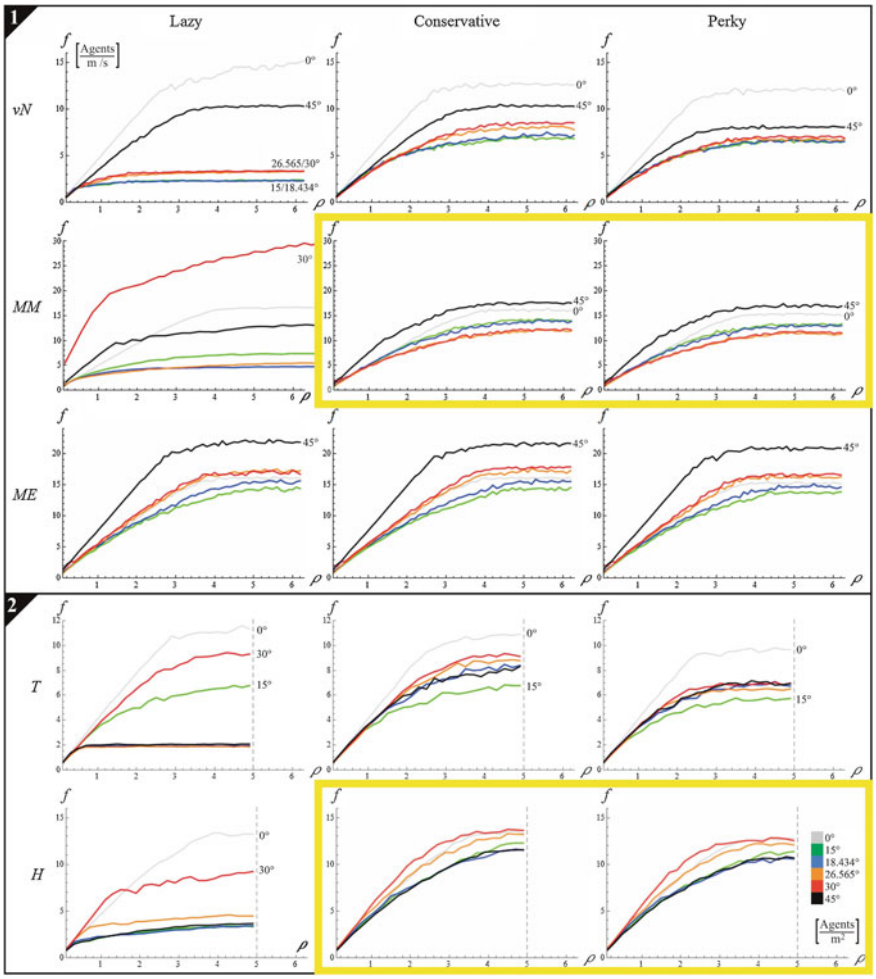

Fig. 4.13 Density-flow rate diagrams for all types of *perkiness*, setups and inclinations. **1** The setups in square grid: vN, MM & ME at crowd density ρ up to 6.25 $\frac{agents}{m^2}$ (theoretical maximum). **2** Triangular (T) & Hexagonal (H) grids at ρ up to 5 $\frac{agents}{m^2}$. The most robust setups in respect to grid rotation are framed in *yellow*

4.6 Conclusions

- The concerns regarding the realism of modeled crowd behavior have been expressed in the past. Deficiencies of crowd simulations (CSs) based on square tessellation have been addressed in the literature. Especially the von Neumann neighborhood has been criticized and a number of improvements have been suggested. For example Ref. [8] has proposed the "wall potential" and "contraction effect" at

a wide exit to simulate the behavior of agents near bottlenecks and corners more realistically.
An alternative approach called "real-coded cellular automata" (RCAs) based on the "real-coded lattice gas" (RLG) has been presented in [14]. RLG has been originally developed for fluid modeling, and RCAs have demonstrated relatively realistic pedestrian simulation in square grid in oblique directions.
Nevertheless, the robustness of most of these findings has not been examined by grid rotations.

- Although the usefulness of hexagonal lattice for traffic simulation has been already discussed over a decade ago in [7], it is finally gaining popularity in CSs [6]. Triangular lattice, however, which sometimes is present in the architectural design, seems to be never considered for such modeling. This chapter puts all three regular tessellations together for comparative analysis with the intention to help selecting the most appropriate setup for a given layout and scenario.
- Grid rotations and agent *perkiness* have been used to study the consistency of qualitative and quantitative performance in simulated scenarios. As the simulations indicate, some setups are more robust in this regard than others.
- The consistency in density-flow is important, however, it should not be assumed as the ultimate criterion of the reliability of a CS model. The qualitative crowd behavior captured by the Heat Maps should also be considered.
- The *perky* agents proceed to the exits the most naturally, and their evacuation paths are the least affected by the grid rotation.
- In this very simple model, all three regular tessellations suit well CSs. However, the computations on the square grid are straightforward, while T and H require additional computations for mapping them on rectangular matrices. Not surprisingly, Euclidean metric, as the most realistic, produces the most natural CSs. In these basic setups hexagonal grid did not perform better than the square one.

References

1. Bandini S, Vizzari G (2007) Regulation function of the environment in agent-based simulation. In: Environments for multi-agent systems III, pp 157–169. Springer
2. Burstedde C, Klauck K, Schadschneider A, Zittartz J (2001) Simulation of pedestrian dynamics using a two-dimensional cellular automaton. Phys A Stat Mech Appl 295(3):507–525
3. Kerridge J, Hine J, Wigan M (2001) Agent-based modelling of pedestrian movements: the questions that need to be asked and answered. Environ Plan B 28(3):327–342
4. Kirchner A, Klüpfel H, Nishinari K, Schadschneider A, Schreckenberg M (2004) Discretization effects and the influence of walking speed in cellular automata models for pedestrian dynamics. J Stat Mech Theor Exp 2004(10):P10011
5. Koyama S, Shinozaki N, Morishita S (2013) Pedestrian flow modeling using cellular automata based on the Japanese public guideline and application to evacuation simulation. J Cell Autom 8(5–6):361–382
6. Leng B, Wang J, Zhao W, Xiong Z (2014) An extended floor field model based on regular hexagonal cells for pedestrian simulation. Phys A Stat Mech Appl 402:119–133
7. Maniccam S (2003) Traffic jamming on hexagonal lattice. Phys A Stat Mech Appl 321(3):653–664

8. Nishinari K, Kirchner A, Namazi A, Schadschneider A (2004) Extended floor field CA model for evacuation dynamics. IEICE Trans Inf Syst 87(3):726–732
9. Rajewsky N, Santen L, Schadschneider A, Schreckenberg M (1998) The asymmetric exclusion process: Comparison of update procedures. J Stat Phys 92(1–2):151–194
10. Rogsch C, Schadschneider A, Seyfried A (2009) Simulation of human movement by cellular automata using different update schemes. In: 4th international symposium on human behaviour in fire (conference proceedings), pp 543–548
11. Ronald N, Sterling L, Kirley M (2007) An agent-based approach to modelling pedestrian behaviour. Int J Simul 8(1):25–38
12. Rosenfeld A, Pfaltz JL (1966) Sequential operations in digital picture processing. J ACM (JACM) 13(4):471–494
13. Shi J, Ren A, Chen C (2009) Agent-based evacuation model of large public buildings under fire conditions. Autom Constr 18(3):338–347
14. Yamamoto K, Kokubo S, Nishinari K (2007) Simulation for pedestrian dynamics by real-coded cellular automata (RCA). Phys A Stat Mech Appl 2(379):654–660

Chapter 5
Application of Crowd Simulation for a Layout Improvement

Abstract This chapter presents a creative experiment, where a series of simulations in Crowd-Z have been used for improvement of an architectural floor-plan. As an example one floor of a retail store has been used for this experiment. A simple scenario of crowd passing from the entrance towards the exit has been implemented. The objective is to possibly uniform the usage of space, in other words to reduce the number of unvisited areas and congestion in other places. Two types of layout elements are introduced: fixed and modifiable. The fixed elements represent structural members such as construction walls, pillars, other unmodifiable elements such as elevators, escalators, etc. The modifiable elements represent partition walls, sale stands, etc. The heat maps (HMs) are used to determine the use of spaces. A semi-totalistic two-dimensional cellular automaton has been defined to "melt" modifiable elements of the layout if the value of HM exceeds a given threshold and to "create" wall-cells in the unvisited areas. The process is iterative. A brief interpretation of the results has been presented.

5.1 Introduction

Crowd simulation (CS) is an established field of research. However, it practically always is used for safety issues, such as emergency evacuation, congestion mitigation etc. This chapter presents a project where CS has been employed for creative purpose. In this example CS has been applied iteratively to "shape" the given space:

1. CS is performed in an architectural layout;
2. The layout is modified according to the HM;
3. Go to 1 until the stop criterion is met.

The objective of this procedure is to uniform the space usage. In other words to remove modifiable elements of the floor-plan that cause congestion and to fill the unused space.

M. Zawidzki, *Discrete Optimization in Architecture*, SpringerBriefs
in Architectural Design and Technology, DOI 10.1007/978-981-10-1106-1_5

5.2 Pre-processing of the CS Environment

Figure 5.1(1) shows the floor-plan for the experiment. Similarly to the examples described in Sects. 3.4.3 and 3.5.2, it has been pre-processed in a CAD program and exported as a digital image. The "checklist" for such preparation is given in Sect. 3.4.3. Figure 5.1(2) shows the layout properly prepared for importing to CZ. Next, the image has been rescaled to fit in the 0.4×0.4 m grid defined by the dimensions of an agent (a simulated pedestrian), as shown in Fig. 5.1(3). For more detail of analogous example see Figs. 3.18 and 3.19.

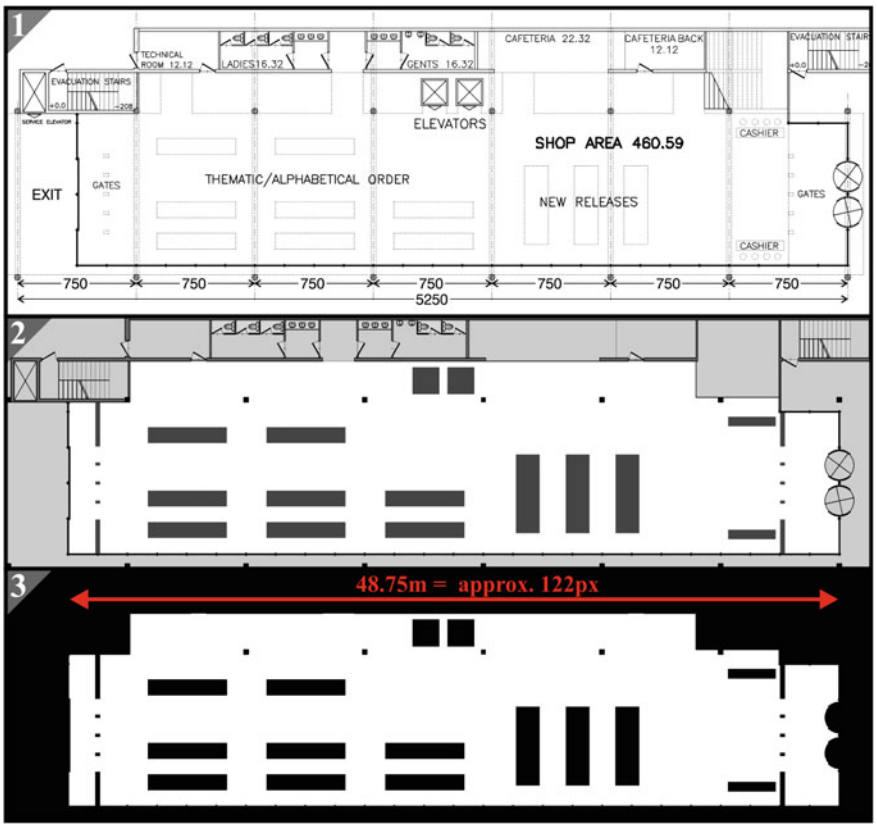

Fig. 5.1 **1** The CAD-drafted layout of one floor of a retail shop. The surface areas and linear dimensions are given in: square meters and centimeters, respectively. **2** The pre-processed layout ready for export from CAD software. The "walkable" area is shown in *white*. **3** The bitmap of the CS environment ready for rescaling

5.3 Setting up the Experiment

After uploading the digital image of the floor-plan to CZ, the fixed and modifiable grid cells have been selected. Next, the exit cells and holding area for the initial positions of the agents have been assigned, as shown in Fig. 5.2(1).

Similarly to other experiments in CZ described in previous chapters, the movement of the agents in this CS is also based on the shortest path strategy. The list of experimental parameters set in CZ for this experiment:

- Moore's neighborhood;
- Maximum metric;
- Number of agents: 200;
- *Perky* agents;
- Ordered sequential update (OSU) scheme;

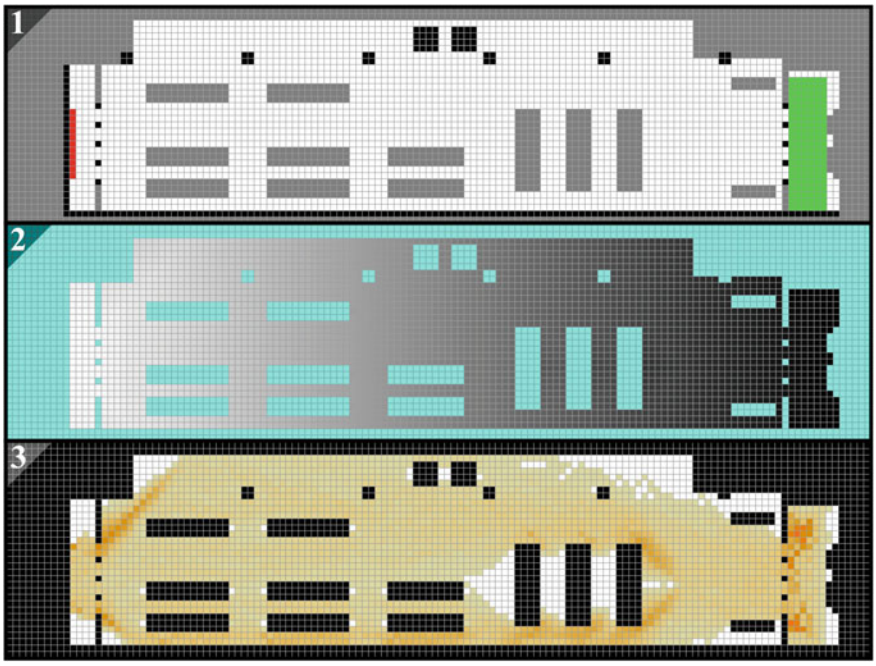

Fig. 5.2 **1** The rescaled floor-plan ready for CS. A single pixel is equivalent to the 0.4×0.4 m area corresponding to the size of an agent. The fixed and modifiable elements are shown in *black* and *gray*, respectively. *Red* indicates the exit cells. *Green* indicates the holding area where the initial positions of agents are randomly distributed. **2** A visualization of the combined DF for all exit cells. The closer is an exit, the brighter are the cells. Background is shown in *cyan*. **3** A visualization of the HM recorded at the preliminary simulation. *Higher color saturation* of cells correspond to more frequent visits of the agents. *White* indicates the unvisited areas

- Error rate $eR = 0$;
- Static potential distance field (DF).

For detailed description of the parameters listed above see Sect. 3.3. The combined potential distance field (DF) for all exit cells is shown in Fig. 5.2(2). Heat map (HM) is a typical method for visualization of the qualitative behavior of crowds. In a nutshell, HM records the frequency of agents' visits in every "walkable" cell of the grid. Figure 5.2(3) visualizes the HM recorded at the preliminary CS of 200 agents moving from the holding area to the exit cells.

As Fig. 5.2(3) indicates, this layout is not well balanced in regarding the use of space in this particular pedestrian scenario. Some areas of the floor-plan are overcrowded, other parts are not visited at all. The following section describes an algorithm for automated "growing" of wall-cells in the most unvisited areas and "melting" the modifiable elements in the congested areas.

5.4 Designing a Cellular Automaton

Cellular automata (CAs) are dynamical systems characterized by *local interactions*. CAs are defined in discrete: time and space, which are quantized to: time-steps and topologically regular lattices, respectively. According to Ref. [1], CAs "*are discrete complex systems which can be regarded as a fully distributed computational system with local processing only of very simple components*". The history of CA commenced from the concepts of automata introduced by Alan Turing [2] and Emil Post [3] in 1930s. Since then CA modeling is used in microstructural mechanics, physics, theoretical biology, mathematics, computation theory and crowd dynamics.

In this experiment, CA "decides" based on local conditions when to "grow" rigid cells (of value 1) in unused parts of the floor-plan and when to "melt" modifiable rigid cells in order to alleviate local congestions. This action is performed in a single-step evolution of two-dimensional radius-one semi-totalistic CA (2DR1STCA).

This CA operates in the superimposed HM and properly prepared floor-plan. Moore's neighborhood defines the eight adjacent cells around (clockwise) the central cell (X). The local neighborhood of X at time-step t is defined as follows:

$$\begin{matrix} a_1^t & a_2^t & a_3^t \\ a_8^t & X^t & a_4^t \\ a_7^t & a_6^t & a_5^t \end{matrix} \tag{5.1}$$

where X^t is the state of the central cell at the time-step t;
$a_1^t \ldots a_8^t$ are the states of eight neighboring cells at the time-step t.

Local transition rules (TRs) define the state of X at the next time-step $t + 1$ as follows:

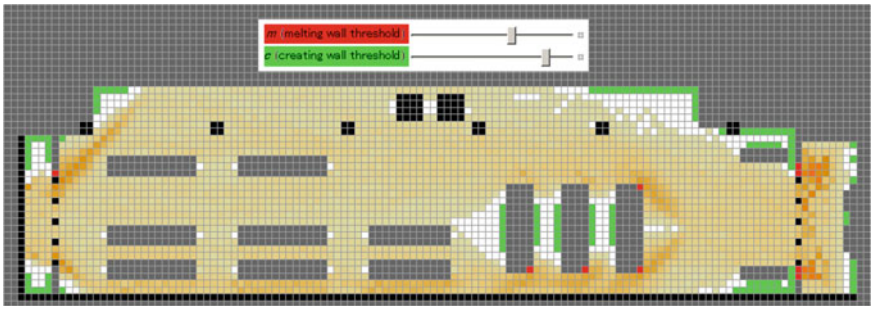

Fig. 5.3 The threshold parameters c and m have been fine-tuned empirically to strike an approximate balance between the number of "grown" and "melted" wall cells. Modifiable walls, fixed walls, newly created cells and removed cells are shown in: *gray*, *black*, *green* and *red*, respectively. Only modifiable wall cells may be removed. On the other hand, the new wall cells can be grown adjacently both to modifiable and fixed wall cells

$$
\begin{aligned}
X^t = w \Rightarrow X^{t+1} &= \begin{cases} 0 & \sum_{i=1}^{8} a_i^t \geq m \\ w & \text{else} \end{cases} \\
X^t = 0 \Rightarrow X^{t+1} &= \begin{cases} w & \sum_{i=1}^{8} a_i^t \leq c \\ 0 & \text{else} \end{cases} \\
X^t = f \Rightarrow X^{t+1} &= f
\end{aligned}
\tag{5.2}
$$

where: a_i^t is an adjacent cell to X at the relative position i at the time-step t;
0, f and w are: "walkable", fixed-wall and modifiable wall cells, respectively;
c is the parameter which triggers the wall "creation", that is changing $0 \rightarrow w$;
m is the parameter which triggers the wall "melting", that is changing $w \rightarrow 0$.

The values of the threshold parameters c and m have been fine-tuned by trial-and-error to reach the relative equilibrium between the intensity of "growing" and "melting" of the cells. Figure 5.3 shows the screenshot of the corresponding CZ interactive panel.

In the following experiments the values for parameters c and m have been set to 3 and 1.5, respectively.

5.5 Iterative Experiment

The procedure of gradual improvement of the floor-plan in respect to the given crowd movement is based on iterations of the following cycle:

1. Begin with the initial floor-plan;
2. Traverse the entire crowd through the current floor-plan;
3. Compute the HM;
4. Combine the floor-plan with the HM;

5. Run the CA in the combined layout and "grow"/"melt" modifiable cells accordingly;
6. The modified floor-plan becomes the "current floor-plan";
7. If the stopping condition is met: return the results, else: go to 2.

There are a number of possible stopping conditions, e.g.:

- The uniformity of HM is satisfactory;
- Relatively minor difference between the subsequent floor-plans;
- The maximal values in HM are reduced to a satisfactory level;
- The number of unvisited cells is reduced to a satisfactory value etc.

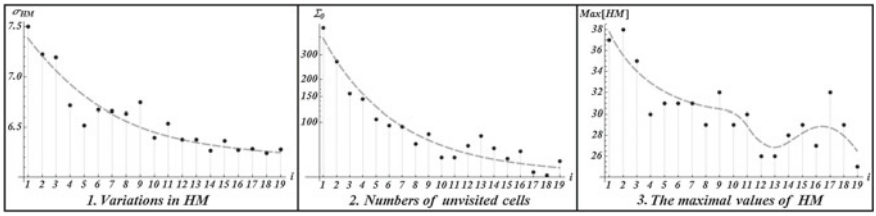

Fig. 5.4 The plots of the statistical parameters characterizing the HMs during 19 iterations. **1** The diversity of values in HM measured by standard deviation σ. **2** The total number of unvisited cells in a floor-plan. **3** The maximal values in HM

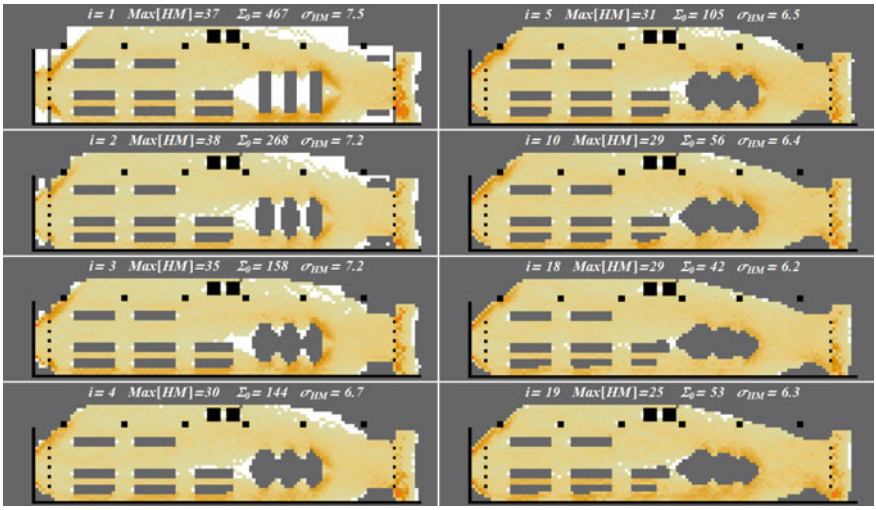

Fig. 5.5 The intermediate and final results of the experiment. The iteration number (i), maximal value in HM ($Max[HM]$), the total number of unvisited cells (Σ_0), and the standard deviation of values in HM (σ_{HM}) are shown above each sub-figure

Here the loop stops and returns the results for $Max[HM] \leq 25$. The CZ procedure runs relatively fast. Completion of 19 cycles takes a few minutes for a modest PC. Figure 5.4 collects the plots of converging main parameters.

As the plots in Fig. 5.4 indicate, the use of space by the agents in this crowd scenario systematically has become more uniform. Almost every iteration produces an improvement. Figure 5.5 shows selected intermediate and final results.

5.6 The Suggestion of the Floor-Plan Alteration

The intention of this chapter is to present the methodology for a rational floor-plan improvement. This method is not intended for a fully automated generation of the optimal layout. Thus the final results produced by the iterative CS need to be interpreted by a designer. In the basic crowd flow scenario presented here, the simulation of pedestrian behavior indicates certain problematic areas in the floor-plan. Namely, there are some overcrowded and unvisited areas, as shown in Fig. 5.6(1).

The improved floor-plan adjusted according to the interpreted results of the experiment is shown in Fig. 5.6(2).

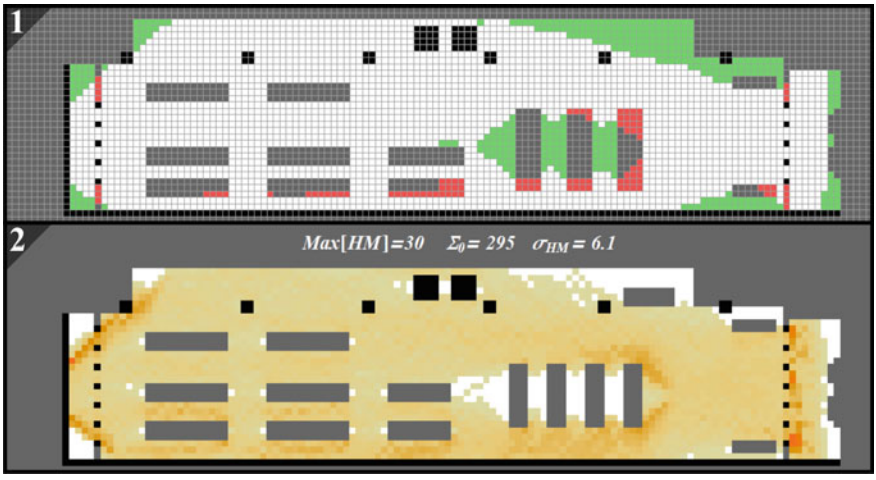

Fig. 5.6 **1** In order to alleviate congestion problem which occurs in the initial floor-plan for the given scenario, the cells marked in red should become "walkable". On the other hand, the unvisited areas marked in *green* can be filled with rigid elements such as walls, furnitures, stands, etc. **2** Relatively subtle adjustments to the floor-plan substantially improve the comfort, safety and efficiency of the space use for the given scenario. The crowd flow has become more uniform despite slightly reduced number of "walkable" cells. Thus the increased surface area of rigid cells can be used for the merchandise display

References

1. Wolz D, De Oliveira PP (2008) Very effective evolutionary techniques for searching cellular automata rule spaces. J Cell Autom 3(4):289–312
2. Turing AM (1936) On computable numbers, with an application to the entscheidungsproblem. J Math 58(345–363):5
3. Post EL (1936) Finite combinatory processes-formulation. J Symb Logic 1(03):103–105

Glossary

Artificial Neural Networks are a family of models inspired by biological neural networks (the central nervous systems of animals, in particular the brain) and are used to estimate or approximate functions that can depend on a large number of inputs and are generally unknown. Artificial neural networks are generally presented as systems of interconnected "neurons" which exchange messages between each other. The connections have numeric *weights* that can be tuned based on experience, making neural nets adaptive to inputs and capable of learning.

Backtracking is a general algorithm for finding all (or some) solutions to some computational problems, notably constraint satisfaction problems, that incrementally builds candidates to the solutions, and abandons each partial candidate ("*backtracks*") as soon as it determines that such a candidate cannot possibly be completed to a valid solution.

Crowd flow state—a "crowded" state, where the interferences among agents impede the flow. As a result increasing the number of agents does not clearly increase the density-flow rate.

Free walking state—an "uncrowded" state when the interactions among agents do not impede the traffic flow. In such a state, the density-flow rate grows proportionally to the number of agents.

Functional layout is a graphic representation of relationships among spaces of a building. Although the level of accuracy greatly varies, the sizes and dimensions of spaces (for example rooms) need to be represented with proper proportions, so the elements of a functional layout roughly correspond to the final architectural plan.

Heat Map—a common way of visualizing the frequency of each cell being occupied by agents throughout the entire crowd simulation.

(Traffic) bottleneck is a localized disruption of vehicular or pedestrian traffic. A bottleneck is a result of a specific physical condition.

M. Zawidzki, *Discrete Optimization in Architecture*, SpringerBriefs
in Architectural Design and Technology, DOI 10.1007/978-981-10-1106-1