PhD Dissertation

International Doctorate School in Information and Communication Technologies

DISI - University of Trento

Optimization-based Strategy for Polyomino Subarraying in Wideband Phased Array Design

Roman Chirikov

Advisor:

Prof. Paolo Rocca

Università degli Studi di Trento

Advisor:

Prof. Valeriy Bagmanov

Ufa State Aviation Technical University

June 2014

Acknowledgements

First of all, I would like to thank Prof. Paolo Rocca and Prof. Andrea Massa for inviting me to the Eledia group, teaching me and changing all my life in the very best sense.

I thank Erasmus Mundus programme for financial support, in particular Kerstin Kruse and Claudia Tarzariol.

I thank Silvano Gambadoro and Andrea Steccanella for the time we worked together.

Special thanks go to Prof. Nafisa Yussupova for believing in me and supporting me.

Many thanks to my teachers, colleagues and dear friends from USATU: Prof. Albert Sultanov, Prof. Valeriy Bagmanov, Prof. Vladimir Vasiliev, Prof. Vadim Kartak, Dr. Ivan Meshkov, Dr. Guzel Abdrakhmanova, Marat Musakaev, Pavel Kucherbaev, Elizaveta Grakhova and others.

Thanks to my English teacher, Lidia Miniakhmetovna, for forcing me to learn the language long before I ever needed it.

Finally, I thank my dearest Mama, Papa, Vova and Granddad for everything I am.

Abstract

Phased antenna arrays provide ultimate performance in areas where high directivity and electronic scanning are needed. That performance is achieved by involving large number of radiators as well as corresponding control units. As a result, such systems become bulky and heavy. In order to reduce the number of control units, elements are grouped in subarrays with one of the control units, such as time delay, put at the subarray level. The drawback of this approach is that if elements are grouped into subarrays producing repetitive patterns in the array structure, radiation pattern of such array will be affected by undesired grating lobes. To eliminate that effect, subarrays of irregular shapes, such as polyominoes, are used. Still, those structures are an object for optimization. This work aims at applying optimization techniques like genetic algorithm to the problem of finding optimal structures of phased antenna arrays composed of polyomino-shaped subarrays. For this purpose a new mathematical model, new algorithm and optimization methods are developed. Application of those techniques showed significant advances in radiation characteristics, in particular sidelobe level. Also new features were enabled, for example, multi-beam radiation pattern forming.

Keywords

phased antenna array, genetic algorithm, optimization, polyomino

Contents

List of Tables

List of Figures

Structure of the Thesis

The thesis consists of chapters as follows.

The first chapter contains analysis of the field of phased antenna arrays together with the state-of-the-art. The actual problem is formulated. Also the topic of optimization is discussed on examples of several common tasks and various techniques are listed.

The second chapter describes development of the mathematical model of planar phased antenna arrays built of polyomino-shaped subarrays. Two optimization methods based on that model are also developed and analysed.

Chapter 3 describes the problem of application of the genetic algorithm and shows in details the algorithm for polyomino placement by Gwee and Lim. Then the "Snowball" algorithm is developed and compared to the Gwee—Lim algorithm with examples.

Chapter 4 is dedicated to the analysis of antenna array structures obtained by two algorithms. It shows advantages of the developed algorithm in terms of sidelobe level suppression.

Chapter 5 includes several examples and analysis of multi-beam radiation patterns obtained by structures with two shapes of subarrays. They show further enhancements of the array technology.

Chapter 1

Introduction and State-of-the-Art

1.1 Actuality of the problem of polyomino placement

The task of optimization of technological processes and solutions has always been a part of a technical thinking. Optimization is a process of finding an optimal solution. Optimal solution is the one that is more preferable than other according to some criteria [86].

Optimization of technological processes can be divided into four general groups.

- *Time optimization* means that the process should run as fast as possible. For example, the slowest part of a conveyor may need to be optimised to increase productivity of the whole assembly process.

- *Raw materials optimization* pursues the most effective usage of the needed raw materials. As an example one may consider optimization of a shape of a plastic cover of some unit to decrease the consumption of plastic [42].

- *Cost optimization* in opposite of raw materials optimization supposes minimization of all the costs including salary and equipment.

- *Quality optimization* has a goal of reaching the best quality in a pro-

cess. The example is manual potato cleaning instead of automated. The process becomes expensive and long but the quality increases.

In a similar way optimization of technological solutions is made in accordance with some set of criteria. The set may consist of one or more criteria. Under a technological solution may be considered a solution of some problem by applying an existing technology as well as some system in a broad sense. Optimization of a system may be focused on choosing the optimal components of the system, optimization of connections (including spatial) between them or all together.

Optimal placement of a set of objects in a limited space represents a wide class of optimization problems. Usually such space is a rectangular area on a plane while objects are geometrical shapes, real objects or even abstract objects represented in geometrical shapes.Further examples of such problems are listed. All of them belong to NP-hard class of problems.

Optimal packing problem. This problem is classical in such areas as system analysis, combinatorics and linear algebra. This problem can be formulated like: there is a container of a given shape in which the maximum number of objects again of given shapes should be placed. The objects may be of different shapes and sizes. Such a problem has been investigated in a one-dimensional [35], two-dimensional [80, 12], three-dimensional [87] and multidimensional variations. In figure 1.1 an example of a solution of such problem is shown.

Optimal cutting problem. The difference of this problem is that there are several shapes into which a sheet of some material should be cut. The point is to minimize the garbage from cutting, i.e. maximization of the density of the shape placement. This problem is studied in several well-known works [30, 29, 46].

The rucksack problem. This problem is different from the previous ones because it is multi-criterial by definition. The essence is in the fol-

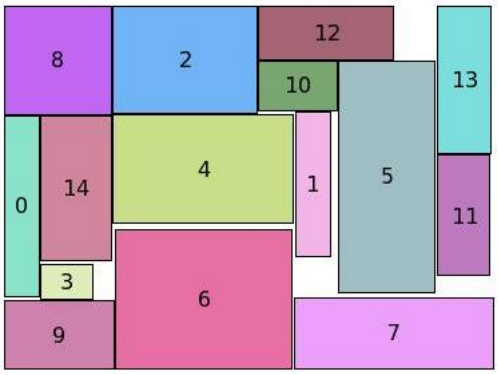


Figure 1.1: Example of a solution of an optimal packing problem

lowing: there is a rucksack of a limited volume. There is a set of objects of given sizes and prices. It is needed to put in the rucksack such a set of objects so that their total cost would be maximal. There is a variation of this problem where instead of the limit of the volume there is a limitation of its load capacity and the objects have weight instead of price.

But along with the listed above problems there is an important task of optimization of the structures of phased antenna arrays built with irregular subarrays in order to increase their energy efficiency, communication range and electromagnetic interoperability. The topic of this work is in the area of optimization of a system that consists of a fuzzy set of components with particular properties and two types of connections — geometrical and electrodynamic. Both connections will be studied during the process of the mathematical model development.

First of all it is necessary to explain what is a phased antenna array and what is it used for. An antenna is a device for radiating or receiving electromagnetic waves. Antennas are the key elements in such well-known areas as wireless communication, radio location and radio astronomy. In recent years the research in wireless energy transfer has got an impulse.

Antennas can be of various types, shapes and sizes. But there are several basic parameters applicable for all of them:

- radiation pattern (RP) — graphical representation of dependence of gain or directivity of the antenna on the direction in a given plane;
- gain — ratio of input power of the reference antenna to the input power of the given antenna given that both antennas produce the same electric field intensity in one direction at the same distance or same power flux density;
- directivity — ratio of squared electric field intensity produced by the antenna in a given direction to the average square of electric field intensity in all directions;
- sidelobe level (SLL) — relative (normalized to the maximum of RP) level of antenna radiation in direction of side lobes;
- front-to-back ratio (F/B) — ratio of the front radiation level to the back radiation level;
- beamwidth — angle within which the radiation level is higher than half of the level of the main beam (or in other words, less than the maximum for 3 dB or less).

In figure 1.2 a typical radiation pattern is shown in polar coordinates. It has a main beam and side lobes including a backside lobe.

In some cases there are special requirements for the antenna parameters, such as high gain or ability to change radiation pattern. In such cases phased antenna arrays (PAA) are used (figure 1.3). Antenna arrays are complex systems consisting of two or more equal antennas joined by one feeding network and functioning in coherence. Radiators belong to passive components. Phased arrays differ by having active components such as amplifiers and phase shifters. PAAs have the ability to apply amplitude and phase corrections at every radiator (array element) forming therefore the

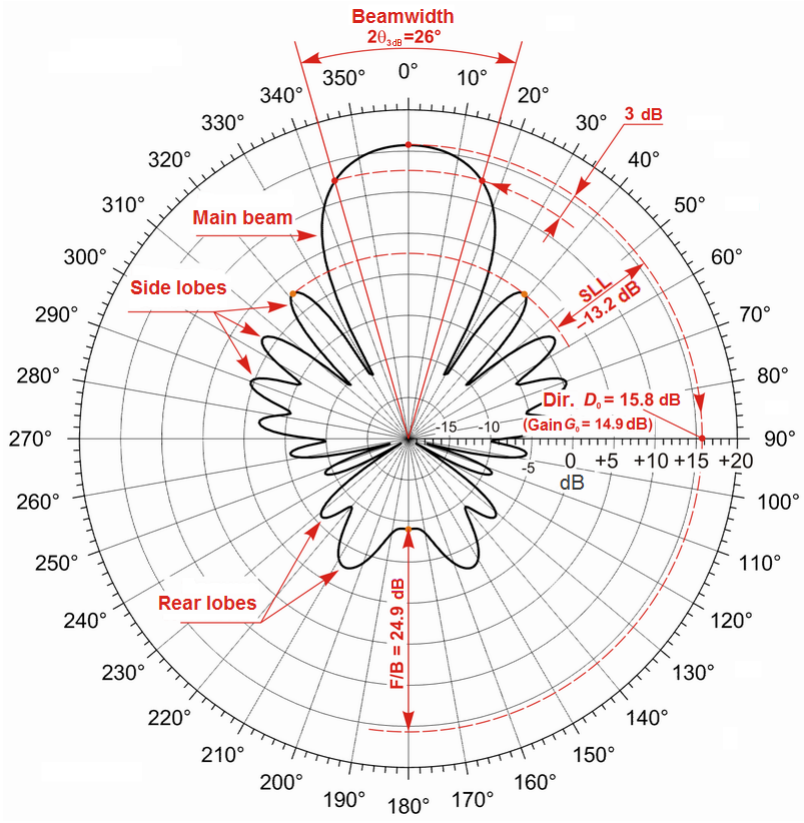


Figure 1.2: Example of a radiation pattern in polar coordinates

needed radiation pattern. Sharp RP increases the quality of transmitting and receiving signals in a given direction decreasing the noise level coming from other directions. PAA may be linear, when all elements are aligned in one line, and planar, when elements lie in a plane.

In order to increase the throughput of wireless communication systems in recent years a shift towards wideband systems has been initiated [81, 43]. The feature of those systems is wide signal spectrum, i.e. a big set of frequencies of electromagnetic waves is used for signal transmission [77]. As it was said, PAA can change its radiation pattern, i.e. steer its main beam [71]. In case of wideband systems and/or large antenna array the phase shifters cannot handle the task of beam forming [49]. In this case time delay (TD) is required. Such arrays find their application in radio

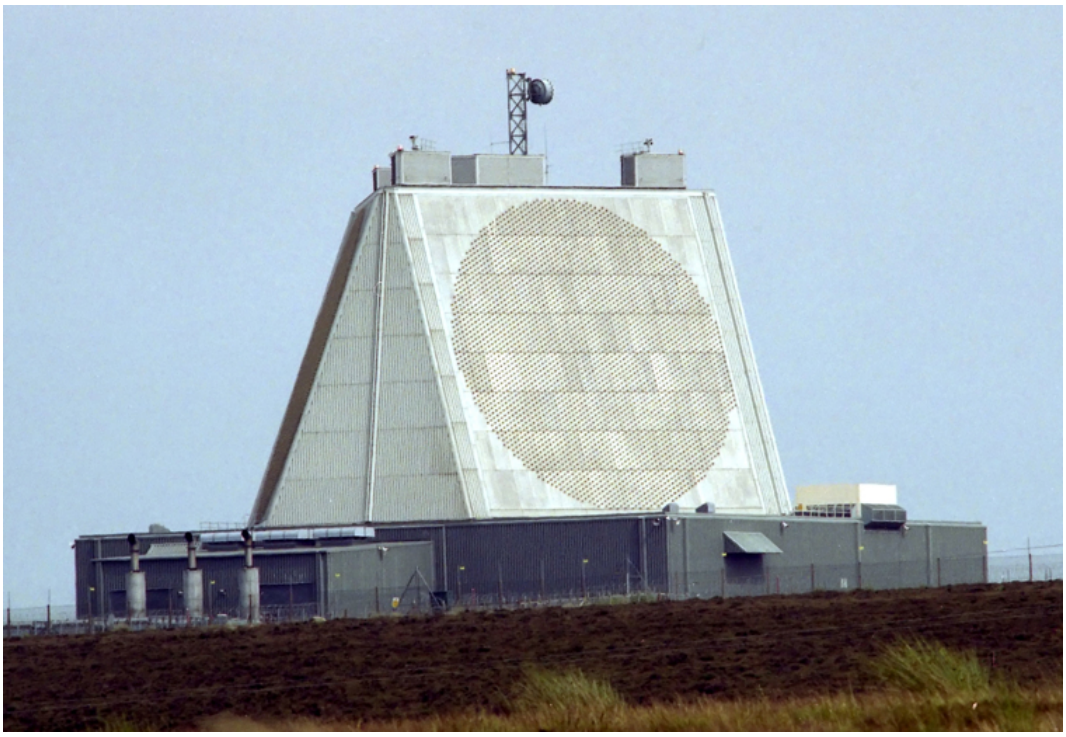


Figure 1.3: Large phased antenna array for radio location

location, radio astronomy and communication systems [32].

A PAA of 8×8 elements is shown in figure 1.4*a*. Every element has the same architecture behind it shown in figure 1.4*b*. A typical radiation pattern is presented in figure 1.5. The RP is shown in three-dimensional sine space (discussed in more details in chapter 3). The radiation level is expressed in decibels relatively to the main beam. The simulation was run with the beam steering to angle $(45°; 45°)$ that corresponds to sine space coordinates $(0,5; 0,5)$. As it is seen from RP, there is one strongly pronounced beam and two rows of weak side lobes in accordance with Taylor current distribution [82].

The problem of time delay components is that they are significantly more expensive, bigger and heavier than other components. If we don't consider the cost, their application is still limited in satellite communications due to its weight. To solve this problem by sacrificing the performance

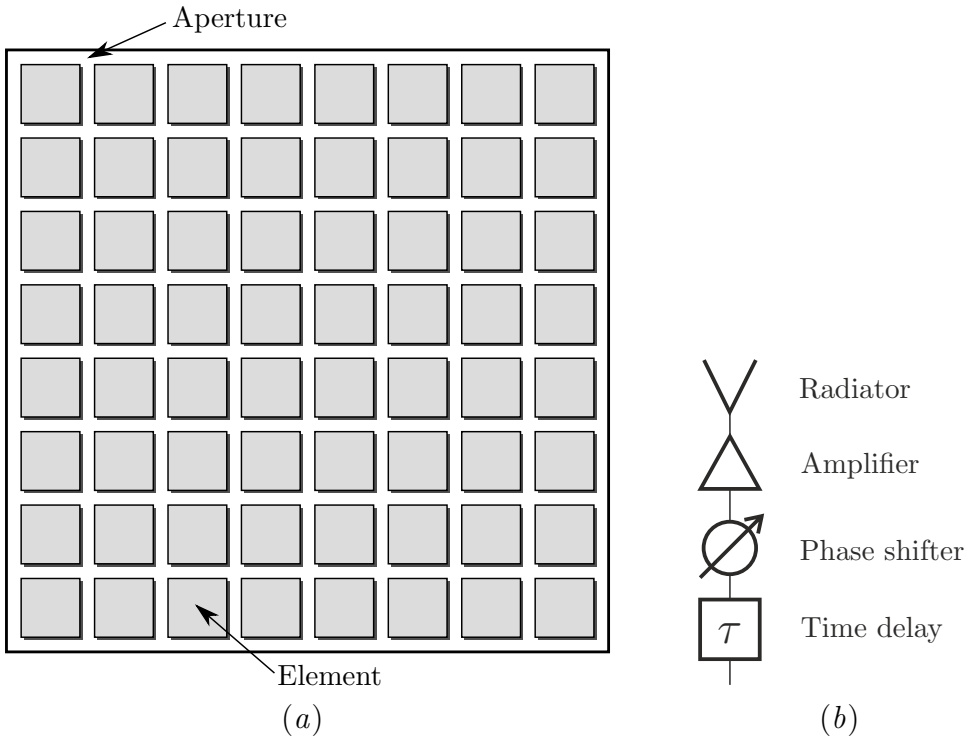


Figure 1.4: Phased antenna array (a) and architecture of an element (b)

it was proposed to split PAA into subarrays and put one time delay component before subarray input, i.e. move time delays to the subarray level [79, 78]. Therefore, the number of needed time delays decreases proportionally to the number of elements grouped in one subarray. In figure 1.6 a 8×8 PAA is shown tiled with subarrays of 4×2 elements.

There is another problem with subarrays. The delay of a signal formed by the time delay is true only for that point of an array that it was calculated for. Let us consider that TD is placed under the central element of a subarray and the value of the delay is calculated for that element. In this case an error will be generated on all other elements of the subarray represented by a time shift of the signal. To be exact, it should be reminded that this effect occurs only for wideband systems. Every element receives its own error value and own time shift. That shift is the same for corresponding elements of neighbour subarrays. Consequently, the sig-

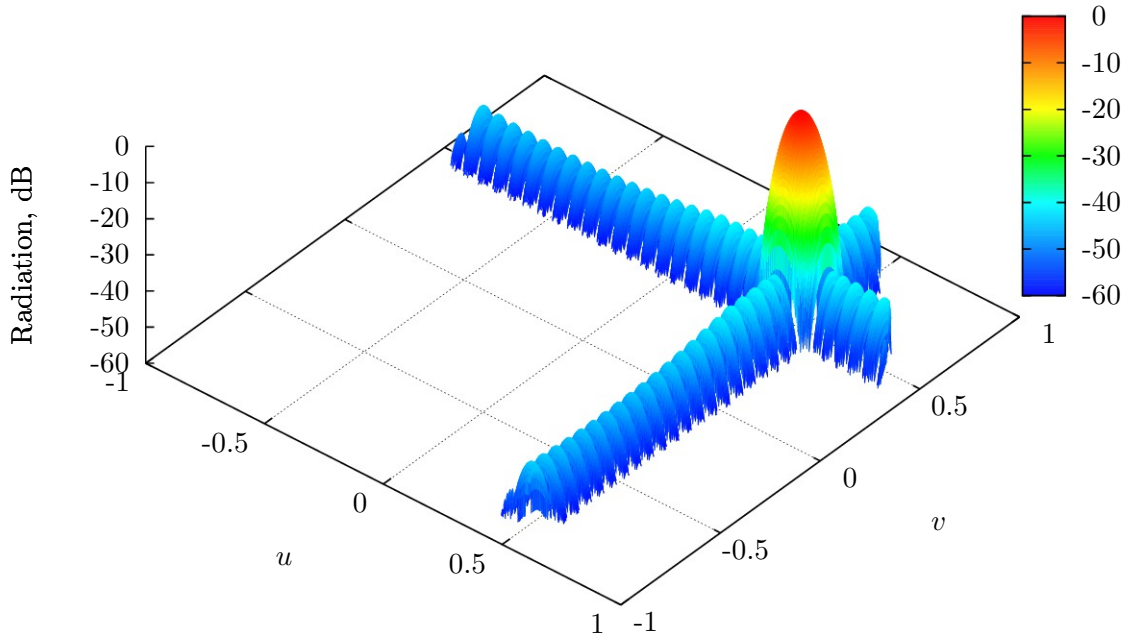


Figure 1.5: Radiation pattern of PAA with no subarraying in the sine space

nal from those elements is coherently added and amplified. It turns out to be an error accumulation. This error is represented in grating lobes in the RP that happen to be side lobes [68]. Their existence is highly undesired because they lead to decrease in power of the main beam and affect the electromagnetic interoperability of a system. Note that the error accumulation from the corresponding elements happens due to the regular placement of subarrays in the array. The illustrative radiation pattern is shown in figure 1.8. Besides the main beam it has five strong side lobes with the highest level at $-9,5$ dB.

One of the ways to overcome the occurrence of strong side lobes is to use the subarrays of irregular shape, namely polyomino-shaped (figure 1.9) [56]. Character C defines the centre. Such shapes rotated by a multiple of 90 degrees allow to eliminate the regularity in the subarray placement and, therefore, prevent the error accumulation. Figure 1.10 shows the same 8×8 structure tiled this time with L-shaped octominoes. On the radiation

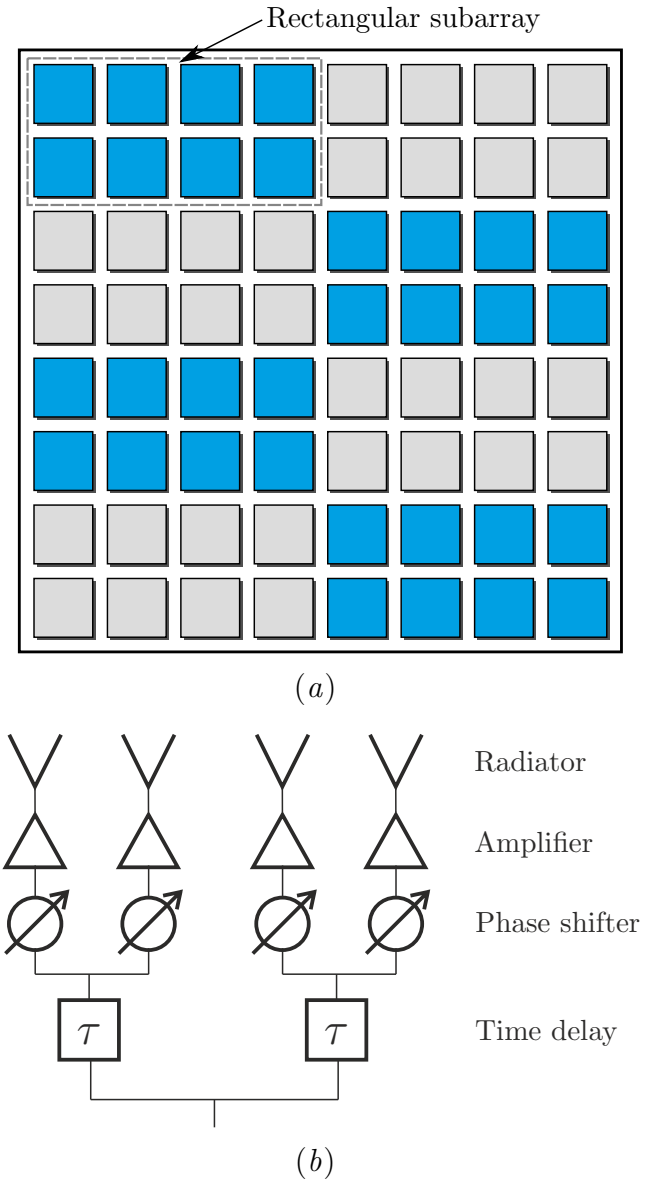
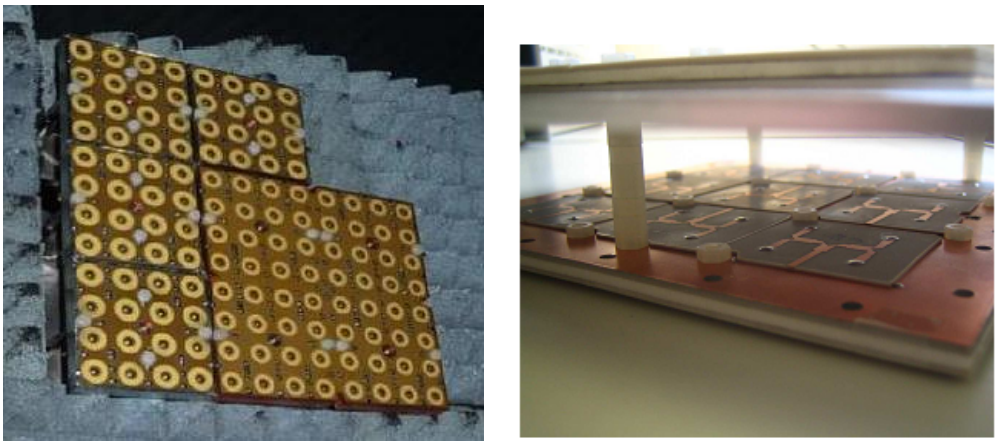


Figure 1.6: PAA tiled with rectangular subarrays (a) and architecture of subarrays (b)

pattern (figure 1.11) it can be seen that the side lobes are spread over the whole space, decreasing therefore its maximum down to $-20,3$ dB.

PAA of the same size in one and the same frequency band but with different structures, obviously, produce different level of side lobes. It is easy to estimate the minimal number of possible structures of size $M \times N$ elements tiled with polyomino with q cells. Let us define the area of a



(a) (b)

Figure 1.7: Picture of a PAA tiled with rectangular subarrays: front view (a) and feeding network behind (b)

minimal rectangle completely tiled with such a polyomino as $S_{\min}$. Inside this rectangle $S_{\min}/q$ polyominoes are placed unambiguously. This rectangle can be rotated by 90 degrees or flipped. In total there are four possible minimal rectangles. Inside the structure there can be $M \times N/S_{\min}$ such rectangles. So, we can say that the minimal number of various structures with full coverage is

$$N_K = \left(2\frac{S_{\min}}{q}\right)^{\frac{M\times N}{S_{\min}}}. \tag{1.1}$$

In table 1.1 this number has been calculated for some polyomino types. As it is seen, even the minimal estimation of the structures number prevents from using brute force search for the optimal structure larger than 8×8 according to some criteria. Therefore, an approximate method is needed.

Table 1.1: Minimal estimation of number of different structures for different polyominoes

	L-tromino	L-tetromino	L-octomino
8×8	2642246	6536	256
16×16	$4.87 \cdot 10^{25}$	$1.84 \cdot 10^{19}$	$4.29 \cdot 10^{9}$
32×32	$5.64 \cdot 10^{102}$	$1.16 \cdot 10^{77}$	$3.4 \cdot 10^{38}$

In papers by Mailloux [59, 61] such structures were composed manually

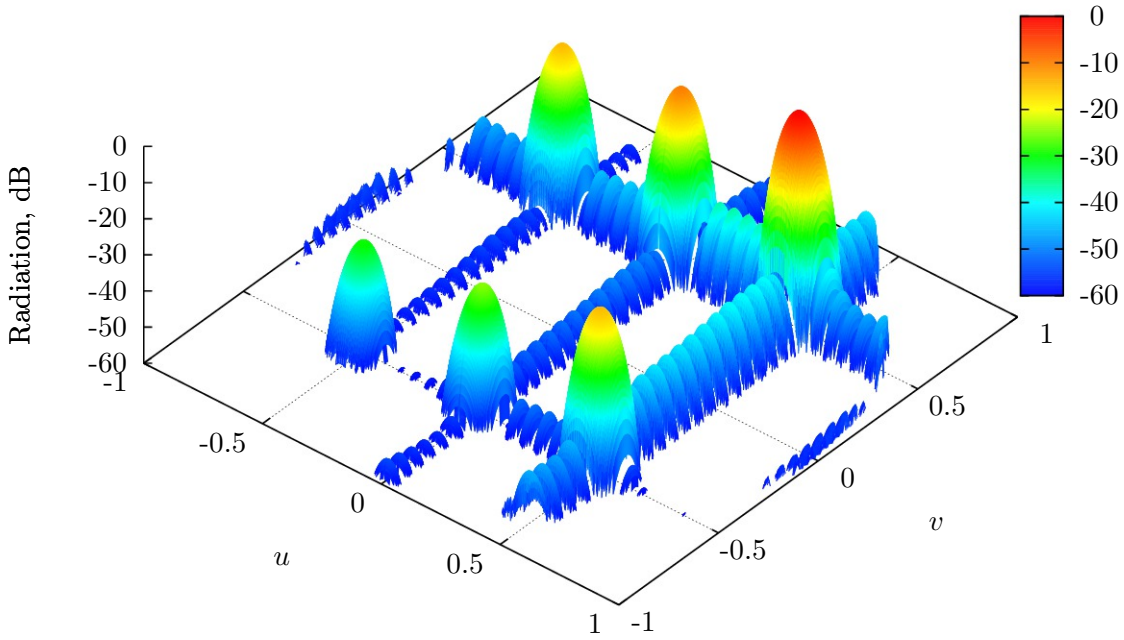


Figure 1.8: Radiation pattern of a PAA built with rectangular subarrays

without any optimization. The question of choosing particular structure was not stated. So, there is an actual problem of optimization of phased antenna arrays structures tiled with polyomino-shaped subarrays.

Therefore, the actuality of the problem was shown. Practical application field was described in details, in which phased antenna arrays act as complex spatially distributed objects. Problems were shown that occur in design of large phased antenna arrays with polyomino-shaped subarraying.

1.2 Analysis of existing methods and approaches to the optimization of planar structures

In the previous section the urgency of the research topic was described. Now it is necessary to analyse the state of the art. Analysis of the achievements in the field will be performed from two sides. From one side, methods and algorithms for optimization of two-dimensional structures, tiled with

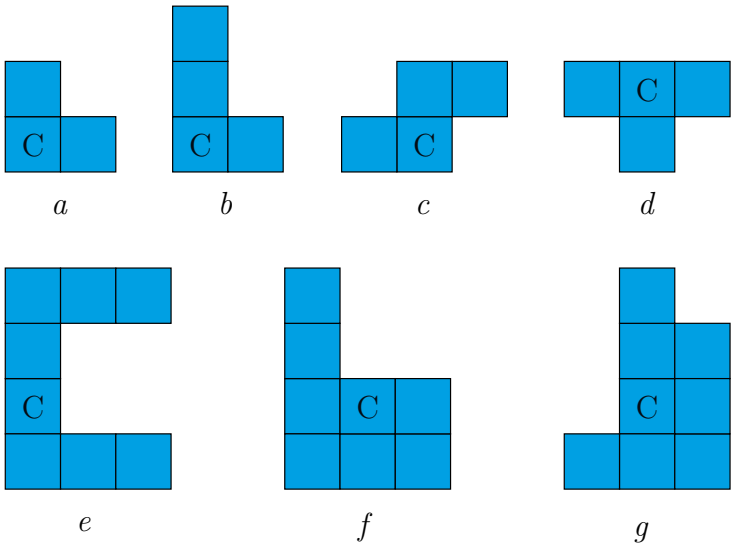


Figure 1.9: Polyomino shapes: L-shaped tromino (a), L-shaped tetromino (b), S-shaped tetromino (c), T-shaped tetromino (d), C-shaped octomino (e), L-shaped octomino (f), PU-shaped octomino (g)

objects of irregular shape, would be considered. That is geometric or system component of the work will be analysed. From the other side, the analysis of sidelobe level suppression methods in phased antenna arrays will be performed.

There are analytical and empirical methods of packing objects in structures. For example, Chinn and Grimaldi in their work analytically pack polyominoes into rectangular areas of the smallest size, which then cover the structure [14].

Opting empirical methods was done for the following reasons:

1. Such evaluation criterion of a PAR structure as the sidelobe level is in the complex implicit dependence on the structure itself and can only be calculated by modelling the entire system. Thus, the optimization problem is finding the global maximum in the large discrete space of solutions [85].

2. number of structures that can be obtained by filling a large rectangular

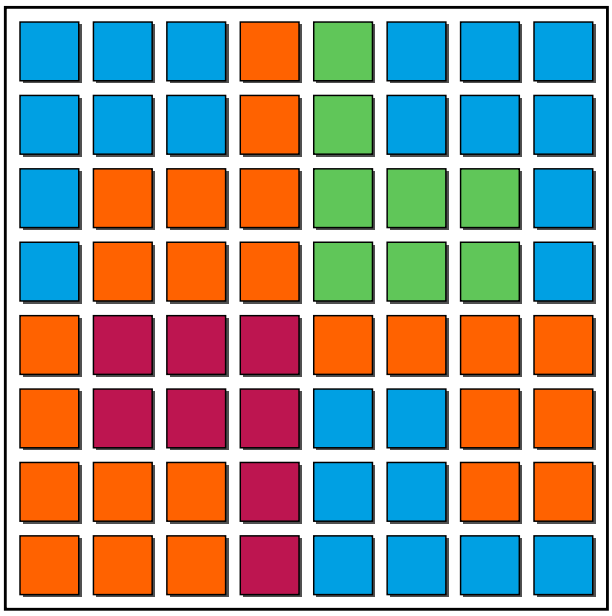


Figure 1.10: Phased antenna array tiled with L-shaped octominoes

area with polyominoes is so big that it is not possible to sort out all the options and model them.

Empirical methods include optimization, that implements search for solutions in some complex multidimensional space. There are a number of studies that have compared different optimization algorithms such as particle swarm optimization method with genetic algorithm [40, 31], evolutionary algorithms [4, 63] and ant colony optimization [73]. Some researchers are interested in combining particle swarm optimization with such genetic mechanisms as breeding and selection [55, 5, 65]. Many variations of the original method of particle swarm optimization were suggested. For example, parallel optimization of several smaller swarms [83, 75, 84], adding negative entropy for mixing the particles [91], dissemination of findings within a limited number of nearest neighbours [48, 50, 62, 47], variation of searching objects in time [3], application of the particle swarm optimization method for controlling mutation in the evolutionary methods [89], dispersal of clustered particles to increase diversity [54], application of fuzzy logic

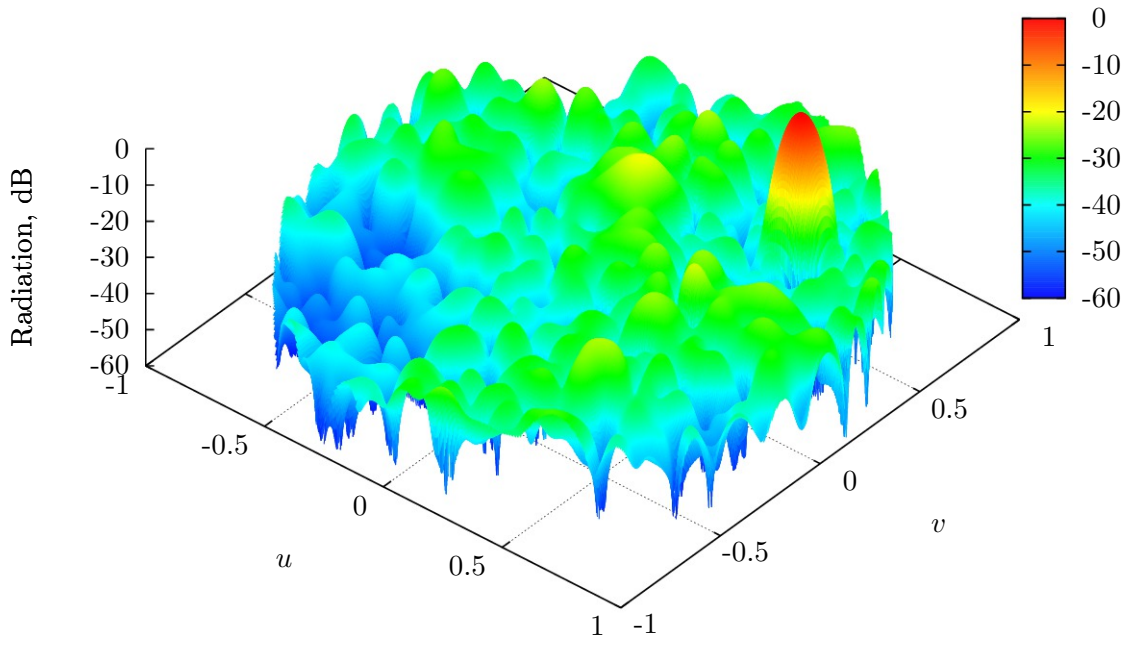


Figure 1.11: Radiation pattern of PAA tiled with L-shaped octomino subarrays

for adjusting the parameters of the algorithm [74]. Daniel Boeringer and Douglas Werner compared particle swarm optimization with the genetic algorithm [8] and showed that the latter has a better ability to beam forming. There are examples of the use of genetic algorithm in electromagnetics and antenna design [90, 2].

Among the empirical methods we have chosen the genetic algorithm (GA) for the following reasons:

1. Independence from the task type. In this work the task is a battery of many parameters for which the optimal value has to be found.

2. Discreteness of the nature of the task. Since PAAs have regular grid and polyominoes are placed in the nodes of that grid with strictly defined possible orientations, the search space is discrete.

3. Continuous actual range of the cost function. In opposite to the search space, the criteria that describe potential solutions are continuous

values. This concept fits well into the nature of genetic algorithms.

Now let us face the question of sidelobe level suppression. As it was mentioned, grating lobes appear due to the accumulation of the phase shift error among regularly placed subarrays. The subarrays and elements inside them could be placed within a periodic and an aperiodic grid. In the first case the array is called equidistant and in the second — non-equidistant. The shape of the subarrays can be rectangular and irregular. Accordingly, there are four domains of planar array subarraying.

The first domain represents the simplest case, when rectangular subarrays are put into a periodic grid. In other words, the array is being split into equal rectangle areas of several radiating elements. Although in this case the area of the array is simply and effectively filled (figure 1.6), the corresponding radiation pattern is characterized by poor radiation performance with grating lobes due to the periodicity of the structure (figure 1.8) [60].

Other three domains aim at breaking this periodicity in different ways. In the second domain the rectangular subarrays are arranged in an aperiodic grid, which results in small arbitrary relative displacements and/or rotations (figure 1.12) [51]. This is the simplest solution for breaking periodicity. This solution is still simple from the point of view of production process but the achieved sidelobe suppression is not high (around 11 dB) [52].

The third domain is represented by irregular subarrays that aperiodically tile the aperture of the array [76, 88]. Here subarrays of more than one irregular shape are used simultaneously and placed in an aperiodic order, thus in nodes of an irregular grid.

One of the examples is the “Danzer” structure by Thomas Spence and Douglas Werner [76]. Their structure consists of many various triangles each being a subarray (figure 1.13). Such an array produces sidelobes of

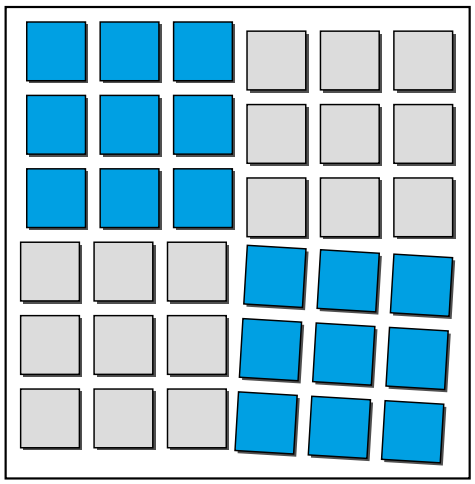


Figure 1.12: Rectangular subarray placement on an irregular grid

−17.3 dB.

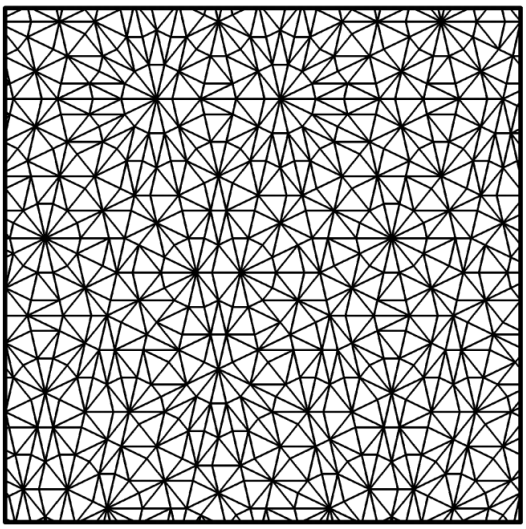


Figure 1.13: The “Danzer” structure

Pierro with colleagues published a structure called “Penrose” [66] (figure 1.14). It also contains a lot of subarrays and suppresses SLL down to −17 dB.

A very beautiful variant of an antenna array structure was shown by Viganó [88]. It is called “Sunflower” (figure 1.15) and also keeps SLL at level of −17 dB.

Much better results are demonstrated by the way of subarrays placement called "Pinwheel" [69]. In their article Morabito and others declare SLL suppression to −21.5 dB.

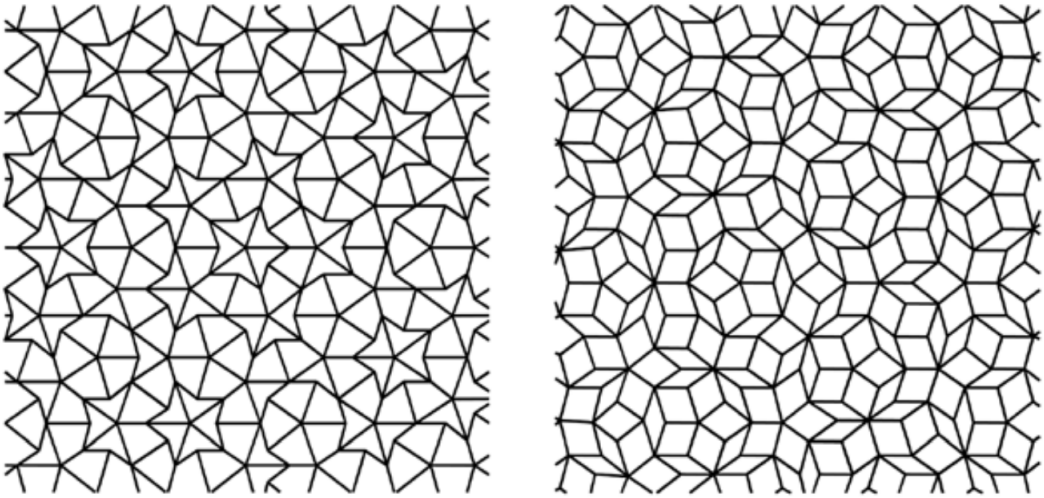


Figure 1.14: The "Penrose" structure

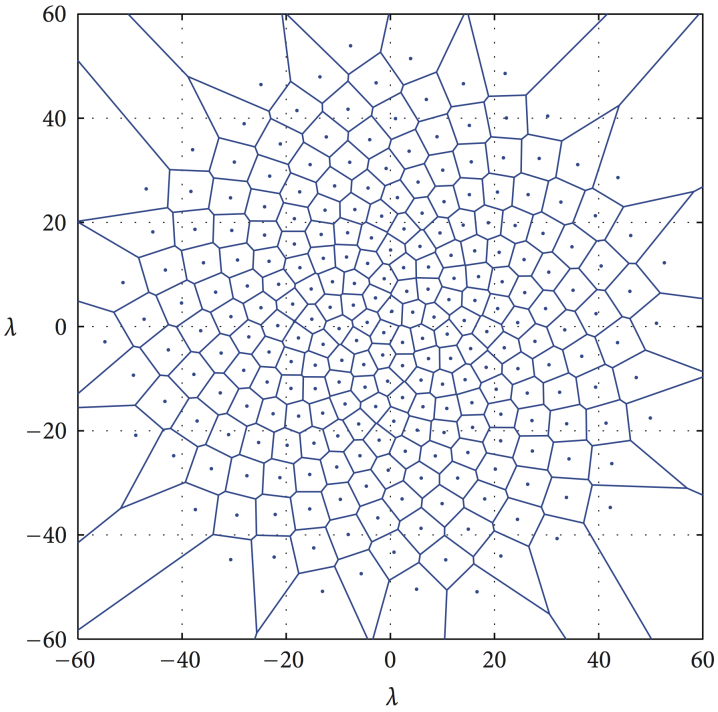


Figure 1.15: The "Sunflower" structure

The results achieved in this domain are significant due to high irregularity of the array structures (also considering arbitrary placement of elements within a subarray) [64, 66]. Although, the structures become very complex and include a big number of elements, which leads to high costs of the arrays and their size and weight.

Accordingly, the fourth domain — the use of subarrays of irregular shapes placed onto a periodic grid — has been adopted to avoid the presence of grating lobes of the array factor [57]. This approach provides good sidelobe level suppression still keeping the structure feasible in terms of mass production (figure 1.11). Having one irregular shape of a subarray it is possible to produce them first and then use them to build the whole array, just rotating the shape. Circular polarization used in communication systems will not be ruined. The question that arises is how to obtain such a structure to meet particular requirements. In other words, the array structure has to be optimized. The problem of large array tiling with subarrays of irregular shape comes to finding subarray positions with minimal number of holes (i.e. uncovered cells), that decrease the gain, and avoiding periodicity in subarray placement in order to minimize the number and level of the side lobes [6].

Therefore, existing methods and approaches to the problem of optimization of planar structures were analysed. Separately the optimization methods for phased antenna array structures were considered. Disadvantages of those methods applied to antenna array design were indicated and the way of research was chosen.

1.3 Goals and tasks of research

After substantiating actuality and analysing present methods and approaches, the goal of the work was set to be an increase of operating efficiency

of phased antenna arrays, including sidelobe level suppression, by means of optimization of their structures composed of polyomino-shaped subarrays. The choice of the genetic algorithm is grounded on the features of the application field. Phased antenna array optimization is one of the ways to fulfil the growing requirements for wireless communication speed and electromagnetic interoperability. The following research tasks have been formulated:

1. To develop a mathematical model of a phased antenna array structure composed of polyomino-shaped subarrays. This model should join geometric properties of the system, common for all planar structures, and electrodynamic properties that are specific for antenna arrays. The development of the model lies in the base of the whole research and is a fundamental step for further activities.

2. To develop a optimization method for polyomino placement based on a criterion of estimation of irregularity of structures. The sidelobe level of an antenna array that is being optimized is connected with subarray placement in the structure, more exactly with their irregularity. Therefore, by the irregularity estimation of a structure it is possible to estimate the sidelobe level. This task plays an important role in universalization of the algorithm to be developed.

3. To develop an algorithm of a structural-parametric synthesis of structures of polyominoes. This is the main theoretical task of the work. The algorithm is meant to synthesize structures, optimized by given criteria applying developed methods.

4. To develop a software based on the proposed algorithm for solving the phased antenna array optimization problem. The task has both theoretical — parameters calibration — and practical sides — synthesis of

antenna array structures. The software is needed for testing the algorithm and running numerical simulations on the obtained structures.

5. To assess the efficiency of the proposed algorithm and obtained structures by the mathematical simulation. Using the software it is needed to get output results of the algorithm and to analyse them, proving that the goal is achieved.

1.4 Chapter 1 conclusions

1. The actuality of the problem was shown. Practical application field was described in details, in which phased antenna arrays act as complex spatially distributed objects. Problems were shown that occur in design of large phased antenna arrays with polyomino-shaped subarraying.

2. Existing methods and approaches to the problem of optimization of planar structures were analysed. Separately the optimization methods for phased antenna array structures were considered. Disadvantages of those methods applied to antenna array design were indicated and the way of research was chosen.

3. The goal of the work was formulated according to analysis of the state of the art in the field. The scientific tasks were stated that will lead to the goal achievement by a consistent progress.

Chapter 2

Development of mathematical model and optimization methods for rectangular structures of polyominoes

2.1 Development of a mathematical model of antenna array structures built of polyomino-shaped sub-arrays

First of all in order to solve the optimization problem it is necessary to formulate and describe it mathematically. Moreover, except for the static characteristics of the whole system it is required to describe the relations between the system components, as well as the characteristics of the components. This is called the development of a model of the system. The model reflects all the necessary properties of the system and its components and its response to external stimuli. Only having a correct model of the system one can develop methods and algorithms for optimization and be sure they are adequate. Main results of the chapter are published by the author in journals and conference proceedings [16, 18, 21, 70, 23].

Thus, the model of the same system may be different for different applications. For example, for the calculation of the deformation forces of the bridge, a model is needed that takes into account the type, size and material of the structure. If we need a model for visualizing the same bridge, it will contain information about the shape, size and colour of the bridge.

The theme of this work is to optimize rectangular structures on the example of antenna arrays. Accordingly, it is necessary at first to consider the characteristics of the antenna arrays and to decide which criteria will be used in the optimization of the characteristics and which must be present in the model. Thereafter, these features must be combined into a single mathematical apparatus capable in terms of the laws of physics to reliably describe the antenna array.

In the first chapter it was mentioned that the antenna array is a system in which there are two types of inter-element relations: geometric and electromagnetic. Also there was a list with definitions of the main characteristics of antennas: radiation pattern, gain, directivity, sidelobe level, front-to-back ratio, beamwidth.

The usage of subarrays of different polyomino shapes (figure 1.9) in the design of the antenna array was originally aimed at the suppression of side lobes in the radiation pattern. However, the maximum suppression of SLL does not mean the maximum coverage of the array by subarrays. If a portion of the array is not included into any subarray, this means that in this area (which may consist of one or more elements) no radiating elements are set. Such areas are called holes. Large number of holes in the array reduces the antenna gain, which affects both the receiving and the transmission of signal. Thus, the optimization criteria selected are sidelobe level and geometric fullness of the array.

In this work planar rectangular equidistant antenna arrays are considered. They are rectangular planar structures consisting of equal-sized cells.

The easiest way to represent this structure is a matrix. Let the structure have M rows and N columns and lies in the $x - y$ plane, where rows are parallel to the axis x, and the columns to axis y (Figure 2.1).

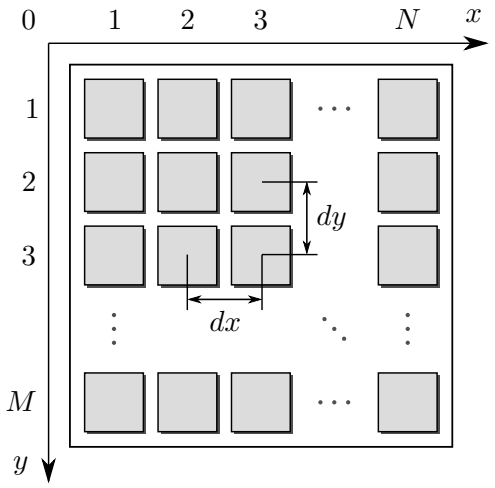


Figure 2.1: Element location in the structure

As was already mentioned, the antenna arrays consist of identical emitters. The distance between the centers of the elements in an equidistant array are the same too within the axes. Note that the physical dimensions of the radiators do not play an important role in the work and are not counted. It is supposed that the dimensions of the emitters are small enough to fit in a predetermined inter-element distance, which is set in the wavelengths at the central frequency of the bandwidth. Accordingly, we denote the inter-element distance along the axes x and y as dx and dy.

An empty structure is represented by a zero-filled matrix [45]:

$$A = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} \tag{2.1}$$

As filling the structure with polyomino forms, matrix elements belonging to those polyominoes are assigned sequence numbers, starting with one.

Each shape of polyomino may be rotated inside the structure by an angle that is a multiple of 90 degrees. In addition, the polyomino can be flipped that adds four more orientations. In total there are eight orientations. Each of the orientations may be provided by a separate matrix with two columns and as many rows as the number of elements in polyomino without one. Matrix elements are the coordinates defining the location of each element relative to a pre-determined center of the polyomino. Each row of this matrix determines the shift of each element polyomino (except center) relative to the center. The first column specifies the offset for axis y, and the second for axis x. For example, the L-shaped octomino comprises eight elements. To describe its orientations we need to make eight matrices of 7×2 elements. Following are the eight orientations of L-octomino in accordance with figures 1.9 and 3.15:

$$T^0 = \begin{pmatrix} -2 & -1 \\ -1 & -1 \\ 0 & -1 \\ 0 & 1 \\ 1 & -1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}, \quad T^1 = \begin{pmatrix} -1 & -1 \\ -1 & 0 \\ -1 & 1 \\ -1 & 2 \\ 0 & -1 \\ 1 & -1 \\ 1 & 0 \end{pmatrix}, \quad T^2 = \begin{pmatrix} -1 & -1 \\ -1 & 0 \\ -1 & 1 \\ 0 & -1 \\ 0 & 1 \\ 1 & 1 \\ 2 & 1 \end{pmatrix},$$

$$T^3 = \begin{pmatrix} -1 & 0 \\ -1 & 1 \\ 0 & 1 \\ 1 & -2 \\ 1 & -1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}, \quad T^4 = \begin{pmatrix} -2 & 1 \\ -1 & 1 \\ 0 & -1 \\ 0 & 1 \\ 1 & -1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}, \quad T^5 = \begin{pmatrix} -1 & -1 \\ -1 & 0 \\ 0 & -1 \\ 1 & -1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{pmatrix}, \qquad (2.2)$$

$$T^6 = \begin{pmatrix} -1 & -1 \\ -1 & 0 \\ -1 & 1 \\ 0 & -1 \\ 0 & 1 \\ 1 & -1 \\ 2 & -1 \end{pmatrix}, \quad T^7 = \begin{pmatrix} -1 & -2 \\ -1 & -1 \\ -1 & 0 \\ -1 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}.$$

These matrices are used by the program and stored in separate files. Possessing a structure matrix and polyomino orientations matrix, we can formulate the condition of possibility of polyominoes placement (in our example, L-octomino) at position (x, y). It is understood that the center of polyomino is located at coordinates (x, y) and the other elements — according to their relative shifts identified by vectors in an orientation matrix. Subarrays can not be superimposed on each other, respectively a polyomino can not be put in the structure, if at least one element of it is in already occupied place. It is easy to figure it out. It is enough to compute the sum of elements of the structure matrix, found in the coordinates from the orientation matrix. Here also a ban is included on crossing border of the structure — all polyominoes must be located entirely within the structure:

$$Y_\mu(x, y) = \begin{cases} A_{x,y} + \sum\limits_{i=1}^{q-1} A_{x_1,y_1}, & 0 \leqslant x_1 < N \wedge 0 \leqslant y_1 < M; \\ 1, & x_1 < 0 \vee x_1 \geqslant N \vee y_1 < 0 \vee y_1 \geqslant M; \end{cases} \tag{2.3}$$

$$x_1 = x + T^\mu_{i,2},$$

$$y_1 = y + T^\mu_{i,1},$$

where μ — orientation of polyomino, q — number of elements in polyomino.

If $Y = 0$ then the placement of the polyomino at a given location is considered possible (and the location is considered suitable). Let us give

an example of placement of the first L-octomino with orientation number zero (matrix T^0) into an empty structure of size 8×8 at the center with coordinates $(4, 4)$:

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{2.4}$$

Therefore, we have a model of a structure (array matrix), model of polyominoes (orientation matrices) and conditions for polyomino placement. This set lets us to describe and consider geometric relations between the elements in a system.

As well as geometric, we need to consider electrodynamic relations to compute the radiation pattern. It is well known that the far field of an antenna array $E(\theta, \phi)$ is derived from the field of a single element:

$$E(\theta, \phi) = E_1(\theta, \phi) \times AF(\theta, \phi), \tag{2.5}$$

where θ and ϕ — spherical coordinates, E_1 — single element field, AF — array factor. For ease, instead of spherical coordinates they use sine space coordinates [67]:

$$\begin{aligned} u &= \sin\theta\cos\phi, \\ v &= \sin\theta\sin\phi. \end{aligned} \tag{2.6}$$

The array factor characterizes the interference of radiation from single elements taking into account given amplitude distribution and phase shifts.

The standard notation is the following:

$$AF(\theta, \phi) = \sum_{m=1}^{M} \sum_{n=1}^{N} a_{mn} e^{-jk[md_x(u-u_0)+nd_y(v-v_0)]}, \tag{2.7}$$

where a_{mn} — amplitude coefficient that sets amplitude distribution, $k = {}^{2\pi}/_{\lambda}$ — wave number, λ — wave length, u_0 and v_0 — steering angle of the main beam. The exponent defines the phase shift from phase shifter. Figure 2.2 shows a simplified scheme of phase shift forming in a linear antenna array composed of three subarrays with four elements in each. The y axis measures the signal phase of each element. Linear phase distribution through the array provides the forming of the main beam in the desired direction.

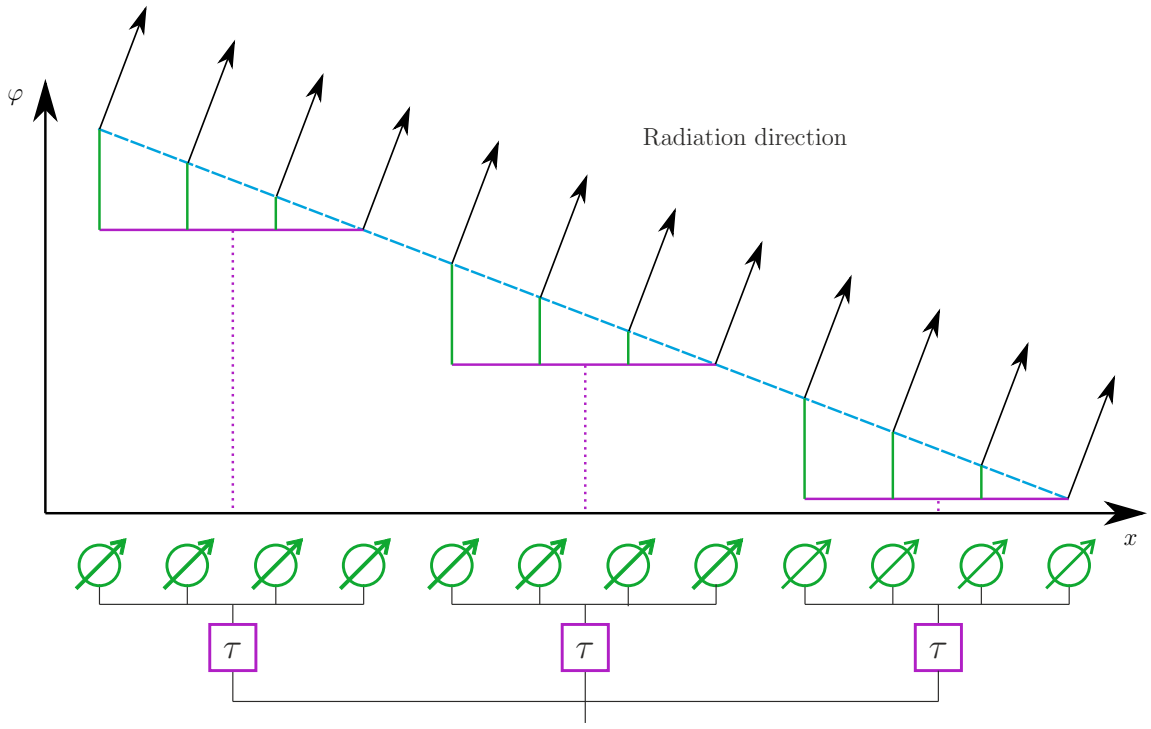


Figure 2.2: Phase shift forming

Equation (2.7) supposes that beneath each element in the array there is an amplifier and a phase shifter (figure 1.4*a*). In case of subarraying that equation should be rewritten. We shall start with rectangular subarrays.

Let the array of size $M \times N$ be completely filled with rectangular subarrays of size $M_0 \times N_0$ (figure 1.6a) with architecture according to figure 1.6b. Subarray factor will be written as:

$$SAF(\theta, \phi) = \sum_{m=1}^{M_0} \sum_{n=1}^{N_0} a_{mn} e^{-jk[md_x(u-u_0)+nd_y(v-v_0)]}. \tag{2.8}$$

Array factor will be expressed through subarrays as:

$$AF(\theta, \phi) = \sum_{m_0=1}^{M/M_0} \sum_{n_0=1}^{N/N_0} SAF(\theta, \phi) e^{-j2\pi f \tau_{m_0 n_0}}, \tag{2.9}$$

where f — frequency, τ — time delay value from the time delay element, that can be calculated for rectangular subarrays placed regularly:

$$\tau_{m_0 n_0} = \frac{1}{c} \left[c_m + (m_0 - 1) M_0 d_x\right] (u - u_0) + \frac{1}{c} \left[c_n + (n_0 - 1) N_0 d_y\right] (v - v_0), \tag{2.10}$$

where c — speed of light and c_m and c_n — subarray center coordinates relatively to top left corner.

In case of polyomino-shaped subarrays the situation becomes more complicated, because polyominoes and their centres are located not regularly. Actually, for this they are used. However, they can also be described mathematically, meaning that we know all the orientations matrices and centre coordinates for each of them. Subarray factor will be expressed as:

$$SAF(\theta, \phi) = \sum_{i=1}^{q-1} a_i e^{-jk\left[T^{\mu}_{i,2} d_x (u-u_0) + T^{\mu}_{i,1} d_y (v-v_0)\right]}. \tag{2.11}$$

In the calculation of the factor of the entire array composed of polyomino-shaped subarrays, it is also needed to know positions and orientations of all the polyomino in the structure:

$$AF(\theta, \phi) = \sum_{i=1}^{N} SAF_i(\theta, \phi) e^{-j2\pi f \tau_i}, \tag{2.12}$$

where N — number of subarrays in the array. Delay τ_i can also be calculated from the subarray centre coordinates in the array [13]:

$$\tau_i = \frac{1}{c}\left[x_i d_x\right](u - u_0) + \frac{1}{c}\left[y_i d_y\right](v - v_0), \tag{2.13}$$

where x_i and y_i — coordinates of the centre of i-th subarray in the array.

Having all these equations one can compute the field of a phased antenna array and measure the sidelobe level. At the same time there is no need to put into equation (2.5) a field of a particular radiator: radiation pattern of a radiating element itself is not optimized in this work, and so instead of a real radiator we can use expression for the ideal isotropic radiator:

$$E_1(r, \theta, \phi) = \frac{e^{-jkr}}{4\pi r}\vec{r}(\theta, \phi), \tag{2.14}$$

where r — distance to the measuring point, $\vec{r}$ — unit vector. In such called far field $r \gg \lambda/2\pi$, therefore (2.14) can be simplified significantly by normalizing the amplitude to some value in the far field:

$$E_1 = 1. \tag{2.15}$$

So, a mathematical model of a structure of polyominoes, representing a phased antenna array, was developed that describes and joins geometrical and electrodynamic relations between the elements. The model takes into account technical features of radiating structures. Radiation properties of the structures are described by array factor.

2.2 Development of the optimization method based on the structure irregularity estimation

In this work the optimization of phased antenna arrays acts as the application area for the optimization methods and algorithms being developed. Among different parameters of antenna arrays in this work optimization is

focused on gain and sidelobe level. The gain is optimized by implication by the increase of geometric fullness of the structure. But the sidelobe level is not connected with evident dependence only to the geometric part of the structure and requires experiments or numerical simulation for its obtaining.

For obvious reasons it is impossible to run hardware experiments during the optimization of an antenna array. There are software libraries for numerical simulations of the sidelobe level. Depending on the sizes of the array and accuracy the simulation may take from half a second up to several minutes. This time multiplied by the number of iterations of the genetic algorithm and population size grows to hours spent on one experiment. In order to solve this problem a task was set to find another optimization criterion that could replace the sidelobe level and be quicker to calculate.

In works of Mailloux [59, 61] it is stated that the sidelobe suppression is proportional to the irregularity of the array structure tiled with subarrays. Therefore, we should search for a criterion that could estimate the irregularity of a structure.

Two attempts were made to find such a criterion. The first one uses the colour filtering method. This way did not show good results and so another attemp was done based on calculation of the autocorrelation function of the structure scanning. The second approach has shown good results and was used in the examples provided in the fourth chapter. Further the two methods are described in more details.

Irregularity of a structure tiled with polyominoes may mean that it does not have patterns repeated with some spatial periodicity. At the same time a pattern can be represented by a single polyomino as well as a group of two, three or more. So, it is important to consider uniform distribution of not only all eight orientations of the polyomino, but also groups of such polyominoes. For this purpose it was proposed to use the principle of colour

filtering. Its essense is in the following.

According to the RGB model, all the colours can be obtained by mixing three basic colours: red, green and blue (figure 2.3). The basic colours are orthogonal to each other: they cannot be obtained by mixing two other colours.

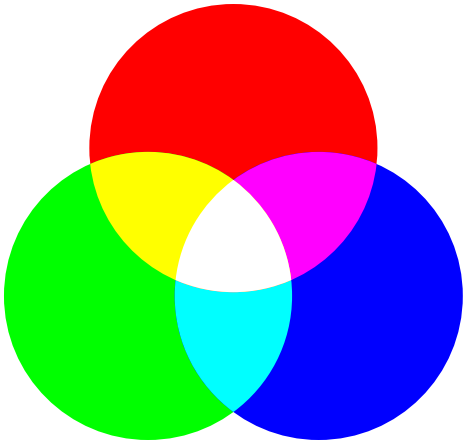


Figure 2.3: RGB model: three basic and three secondary colours

If we mix each pair of basic colours in equal proportion we will get three secondary colours:

$$\text{red} + \text{green} = \text{yellow}$$
$$\text{red} + \text{blue} = \text{magenta}$$
$$\text{green} + \text{blue} = \text{cyan}$$

In total we can use these six colours. We paint polyominoes in the structure with these colours. Each colour is associated with one orientation. In figure 2.4 a structure is shown where all the polyominoes are painted in their colours.

Now let us describe colour channels. Colour channels correspond to the basic colours of the model. In RGB it is red, green and blue. They say a colour is visible in a channel if the corresponding basic colour is used to obtain it. Therefore, in the red channel among our six colours we will see red, yellow and magenta. In the green channel it is green, yellow and cyan

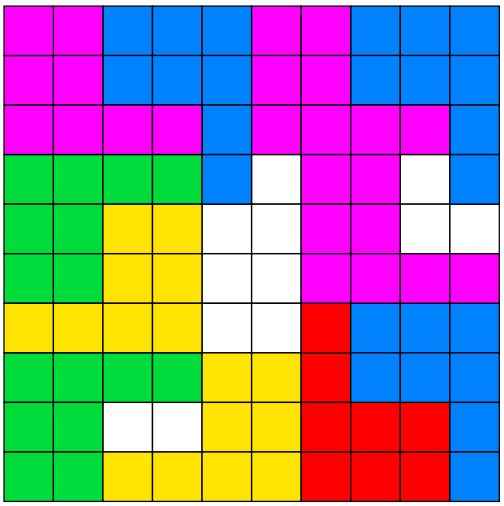


Figure 2.4: Example of a structure in which polyominoes are painted in colours according to orientations

while in blue channel — blue, cyan and magenta.

Now, if we "turn on" only one channel we will see only those polyominoes in the structure that are painted in the corresponding visible colours. Figure 2.5 shows the initial structure in each of three channels.

But we ought to remember that every polyomino in the structure has eight orientations, while there are only six basic and secondary colours. Black colour is used to designate invisible polyominoes and holes that are invisible in any channel. White colour is useless because it is visible in all channels.

It is impossible to find four orthogonal colours. But we can abstract our mind from colours and transfer the same principle (mixing and elicitation) to other objects. In this work the prime numbers have been chosen as such objects. Four imaginary colours act as basic: C_2, C_3, C_5 and C_7. They are orthogonal and they don't divide by one another. Their multiplication will represent mixing. Since the numbers are prime, every product will be divisible by only the numbers that represent basic colours. In total there

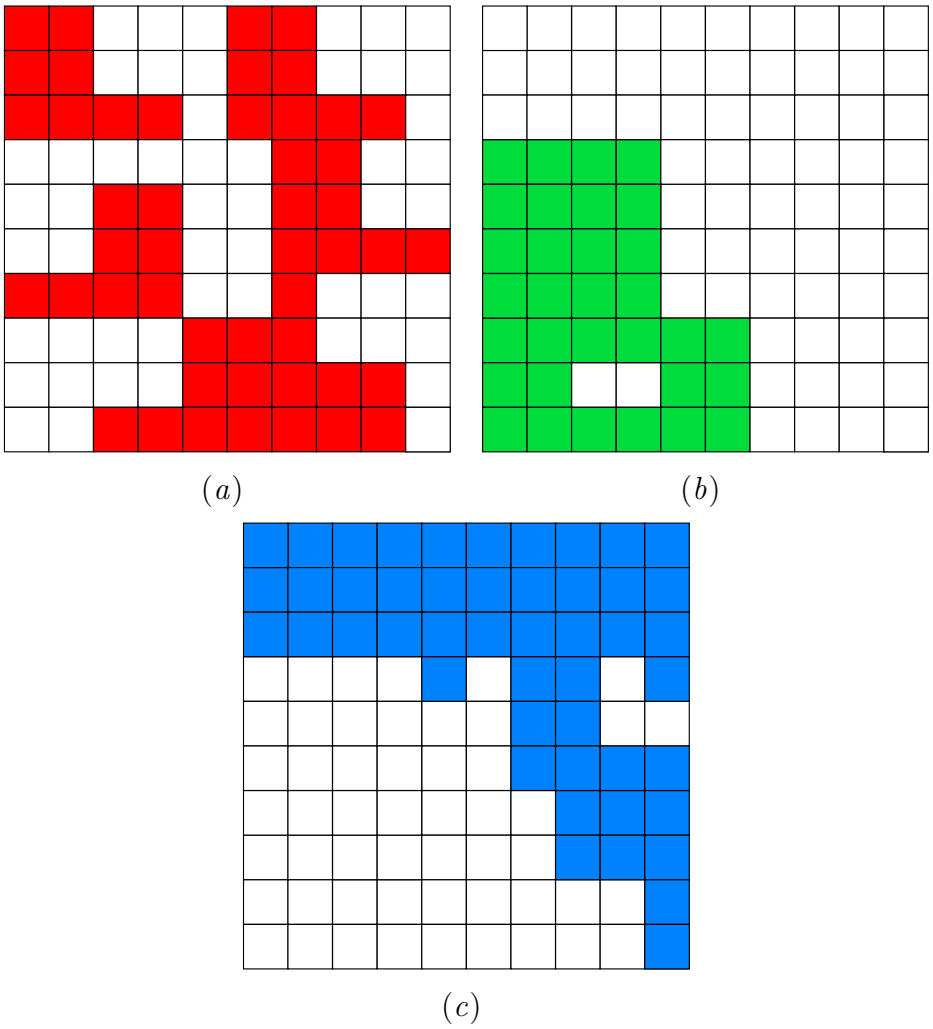


Figure 2.5: Views of the structure in red (a), green (b) and blue (c) channels

are six secondary colours:

$$C_2 + C_3 = C_6,$$
$$C_2 + C_5 = C_{10},$$
$$C_2 + C_7 = C_{14},$$
$$C_3 + C_5 = C_{15},$$
$$C_3 + C_7 = C_{21},$$
$$C_5 + C_7 = C_{35}.$$

We will use only two basic colours (C_2 and C_3) and six secondary to paint eight orientations of polyominoes.

Then we need somehow to estimate the uniformity of visible elements in the structure in each channel. For that we calculate the number of visible elements in all the rows and columns and their standard deviation (separately among rows and columns). The average value is set exactly to the number of elements in a row/column per one channel. In channels C_2 and C_3 four colours are visible, while in channels C_5 and C_7 only three. Therefore we divide the number of elements in a row/column by 3.5 to obtain the average:

$$\langle U^{(C)} \rangle = \frac{N}{3.5}, \tag{2.16}$$

$$\langle V^{(C)} \rangle = \frac{M}{3.5}, \tag{2.17}$$

$$\sigma_C^{\shortmid} = \sqrt{\frac{1}{M} \sum_{i=1}^{M} \left(V_i^{(C)} - \langle V^{(C)} \rangle \right)^2}, \tag{2.18}$$

$$\sigma_C^{-} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left(U_i^{(C)} - \langle U^{(C)} \rangle \right)^2}, \tag{2.19}$$

where $\sigma_C^{\shortmid}$, σ_C^{-} — standard deviation of visible elements in a row and column in channel C, $U_i^{(C)}$ and $V_i^{(C)}$ — number of visible elements in the i-th row or column in channel C, M and N — number of rows and columns in the structure.

By this the information about the uniformity of the elements distribution for each colour channel is extracted. Then all the standard deviations are summed up forming a numerical value of the irregularity of the structure R:

$$R = \sum_C \left(\sigma_C^{\shortmid} \sigma_C^{-} \right). \tag{2.20}$$

The optimization criterion in this case will be positive minimization down to zero, meaning uniform distribution of visible elements among rows

and columns in all the channels and, therefore, absence of repeated patterns inside the structure.

Next we present several experiments to identify the dependence between sidelobe level and calculated value of irregularity of structures.

2.2.1 Example values of irregularity by colour filtration method

Here two examples of structures of 32×32 elements are presented for which irregularity values are calculated by the colour filtering method as well as sidelobe levels. In the first example the structure is tiled with L-shaped trominoes, in the second — L-shaped octominoes. These examples are listed for understanding the range of values of irregularity. According to the formulae, those values can be non-integer. In the first example it is 346.36 and in the second — 370.54. At the same time sidelobe level differ slightly: 8.5 dB for bandwidth $r = 1.3$ and 11 dB for bandwidth $r = 1.818$.

Example 1: structure 32 × 32, L-tromino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-tromino.

Figure 2.6 shows the structure, numerical results are given in table 2.1.

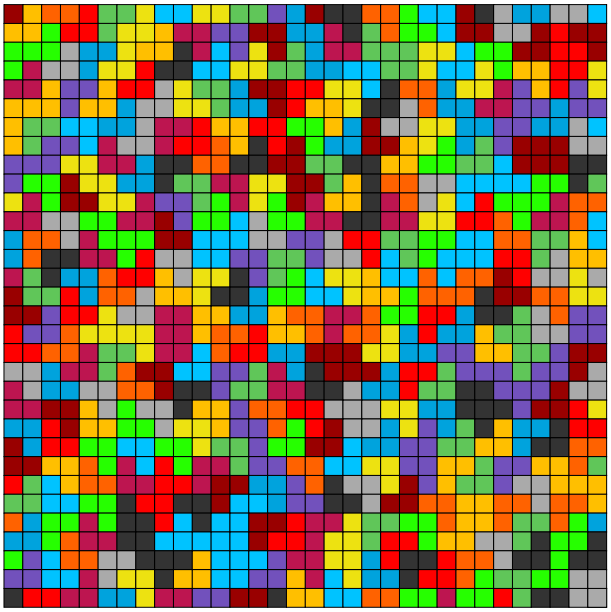


Figure 2.6: Array structure in the first example

Table 2.1: Output data of the first example

Parameter	Value
Number of polyominoes	363
Fullness of the structure A, %	100
Irregularity R	346.36
Sidelobe level γ for $r = 1.300$, dB	−28.43
Sidelobe level γ for $r = 1.818$, dB	−21.76

Example 2: structure 32 × 32, L-octomino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-octomino.

Figure 2.7 shows the structure, numerical results are given in table 2.2.

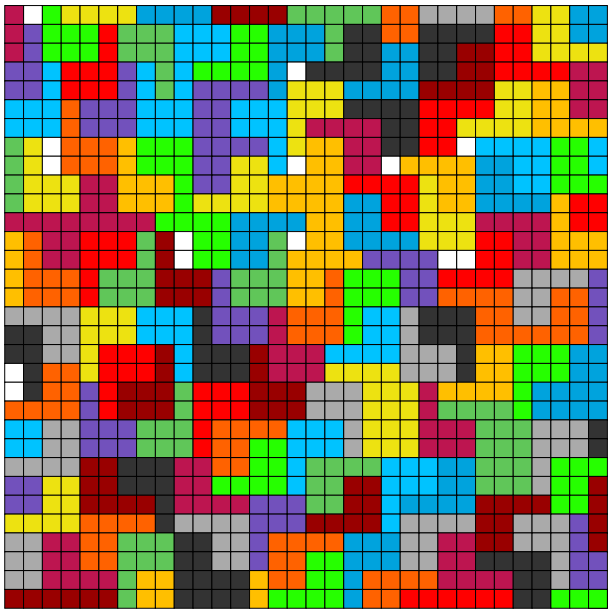


Figure 2.7: Array structure in the second example

Table 2.2: Output data of the second example

Parameter	Value
Number of polyominoes	144
Fullness of the structure A, %	98.63
Irregularity R	370.54
Sidelobe level γ for $r = 1.300$, dB	−19.97
Sidelobe level γ for $r = 1.818$, dB	−10.76

Below the analysis of the method is provided from the point of view of stability of values and correspondence to the sidelobe level.

2.2.2 Colour filtering method analysis

Figure 2.8 shows graphs of irregularity calculated by the colour filtering method and sidelobe level obtained by simulation for bandwidth $r_{sim} = 1.3$. The experiments were run for structures of different sizes tiled with L-tromino. Numerical data is presented in table 2.3.

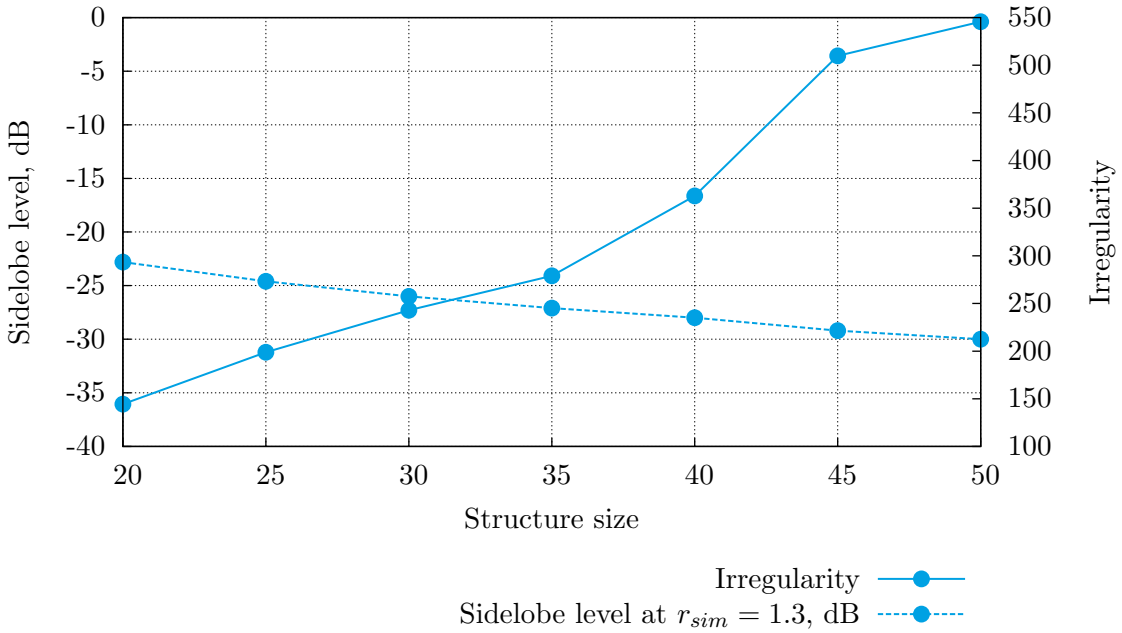


Figure 2.8: Irregularity and sidelobe level of structures of different sizes, tiled with L-trominoes

Table 2.3: Irregularity and SLL of structures of different sizes of L-tromino

Structure size $M = N$	Irregularity R	SLL, dB
20	144.32	−11.75
25	198.80	−13.30
30	242.85	−16.25
35	279.09	−20.20
40	362.88	−19.39
45	509.85	−21.76
50	545.69	−21.24

Similar analysis for structures tiled with L-octominoes is shown on figure 2.9 and table 2.4.

From this graphs it can be seen that the irregularity value behaves in opposite to the sidelobe level. Despite that the value of irregularity can be inverted and scaled, beforehand we need to make sure that values are stable, i.e. see how do they differ for structures with similar SLL.

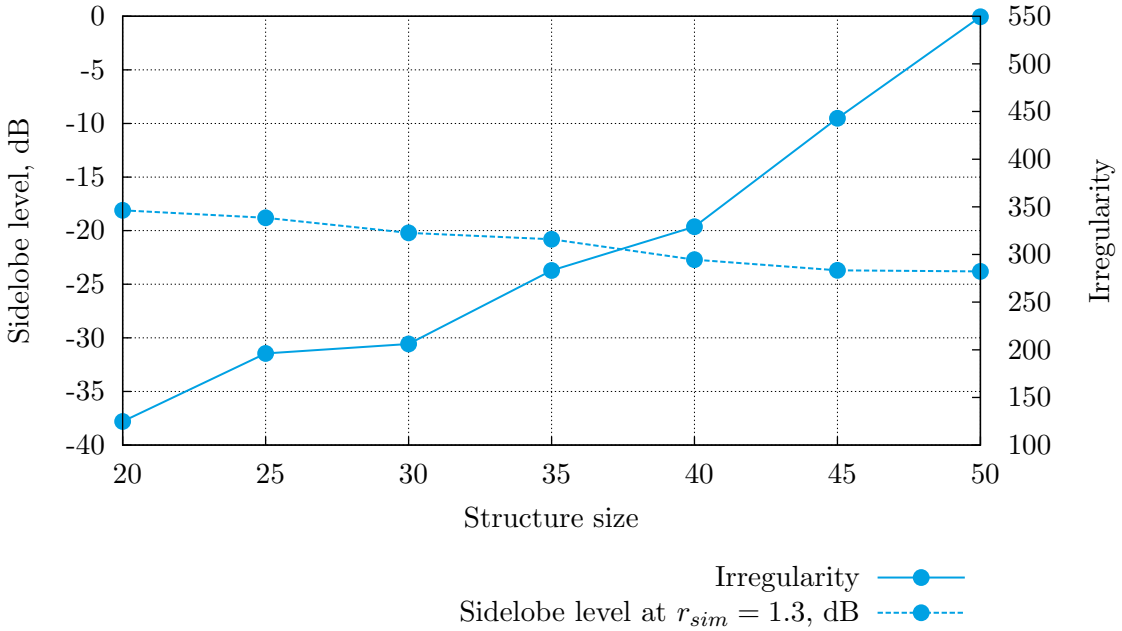


Figure 2.9: Irregularity and sidelobe level of structures of different sizes, tiled with L-octominoes

Table 2.4: Irregularity and SLL of structures of different sizes of L-octomino

Structure size $M = N$	Irregularity R	SLL, dB
20	124.71	−11.38
25	196.14	−11.94
30	206.15	−10.63
35	283.20	−12.51
40	329.16	−14.90
45	442.91	−15.33
50	549.38	−16.21

Further there are graphs, comparing irregularity values with sidelobe level of structures obtained with different random generator seeds. SLL was simulated for $r_{sim} = 1.3$ and $r_{sim} = 1.818$. The structures are of equal sizes 32×32. Figure 2.10 shows the graph for structures tiled with L-trominoes while figure 2.11 presents the graph for structures of L-octominoes.

From the graphs it can be seen that the irregularity values are not only unstable and allow a deviation of ± 25 with SLL being more or less the same, but also demonstrates ungrounded tendency (figure 2.11). Considering this we can say that the developed method is not suitable as a replacement for sidelobe level in cost function.

Nevertheless, the method has proved its capacity. It can be studied further and applied to other fields of research dealing with planar discrete structures. For example, the method can be used for calculation of image hashes,

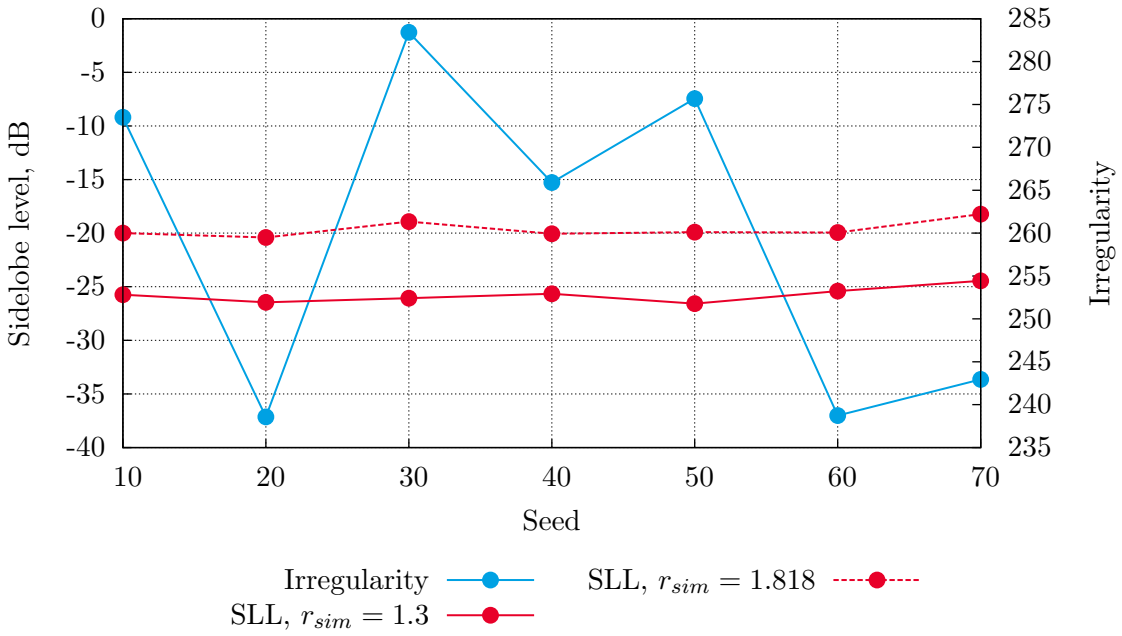


Figure 2.10: Irregularity and sidelobe level of structures of same size tiled with L-tromino

Therefore, a colour filtering method has been developed that provides

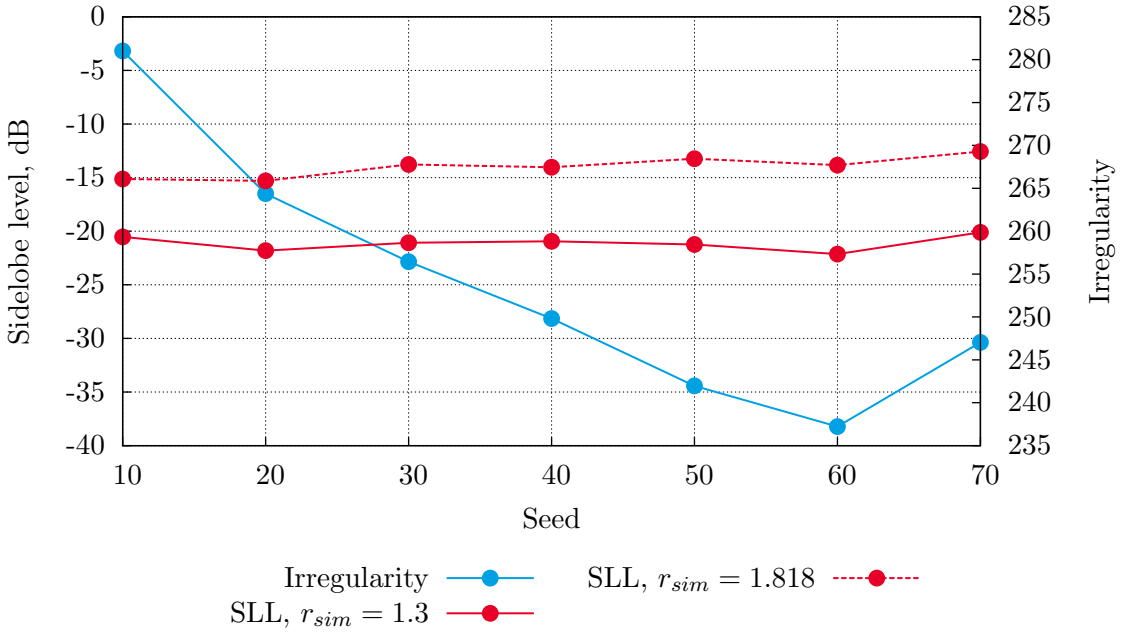


Figure 2.11: Irregularity and sidelobe level of structures of same size tiled with L-octomino

numerical estimation of irregularity of a structure. The method was implemented and analysed. As a result of the analysis it was found out that the method cannot be used for PAA optimization, because the derivable irregularity estimation does not correlate with the sidelobe level.

2.3 Development of an optimization method based on estimation of structure's self-similarity

Experiments with colour filtering method showed that the calculated value of irregularity has almost nothing to do with the simulated value of the sidelobe level. We had to find a new approach for the optimization criterion that could replace SLL.

Since we talk about irregularity of some structure, it makes sense to apply the concept of autocorrelation. Autocorrelation function expresses the degree of self-similarity of a function at a certain shift of the argument.

There is a two-dimensional expression of the autocorrelation function, but its computing time grows exponentially with the growth of the area of two-dimensional domain of the function (in this case, the area of the structure). It was therefore decided to use the one-dimensional expression.

In order to calculate one-dimensional autocorrelation function from a two-dimensional function one has to reduce the dimensionality of it. In other words, obtain its scanning. The structure contains a finite number of discrete elements, therefore the scanning will also be discrete and finite. There are several different scannings. The simplest one — serial scanning which joins rows or columns of a structure in series (figure 2.12). But conformably to our task the recursive Hilbert scanning is of special interest [39, 26].

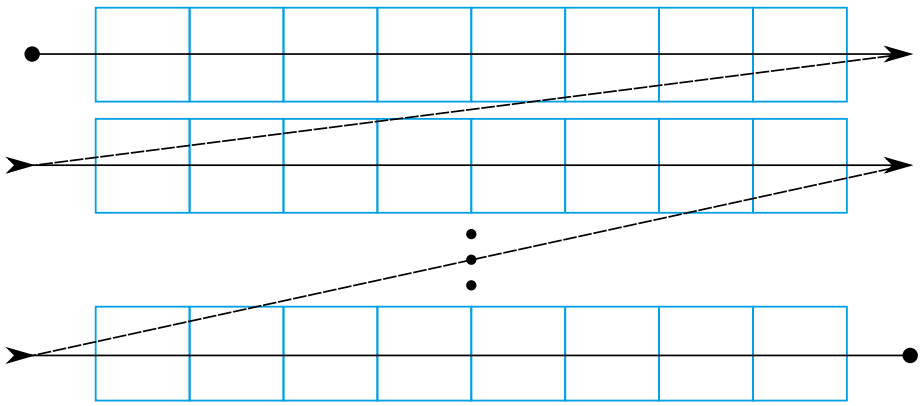


Figure 2.12: Serial scanning

The Hilbert scanning represents a continuous line, that passes all elements in the structure [72]. Depending on the size of the structure, various number of iterations is needed (figure 2.13). At the same time the size of a structure covered by the scanning is a power of two.

Let the structure have $M \times N$ elements. The length of the Hilbert curve for it will be:

$$l_h = M \times N. \tag{2.21}$$

The Hilbert scanning of such a structure will return a vector $\vec{s}$, consisting of

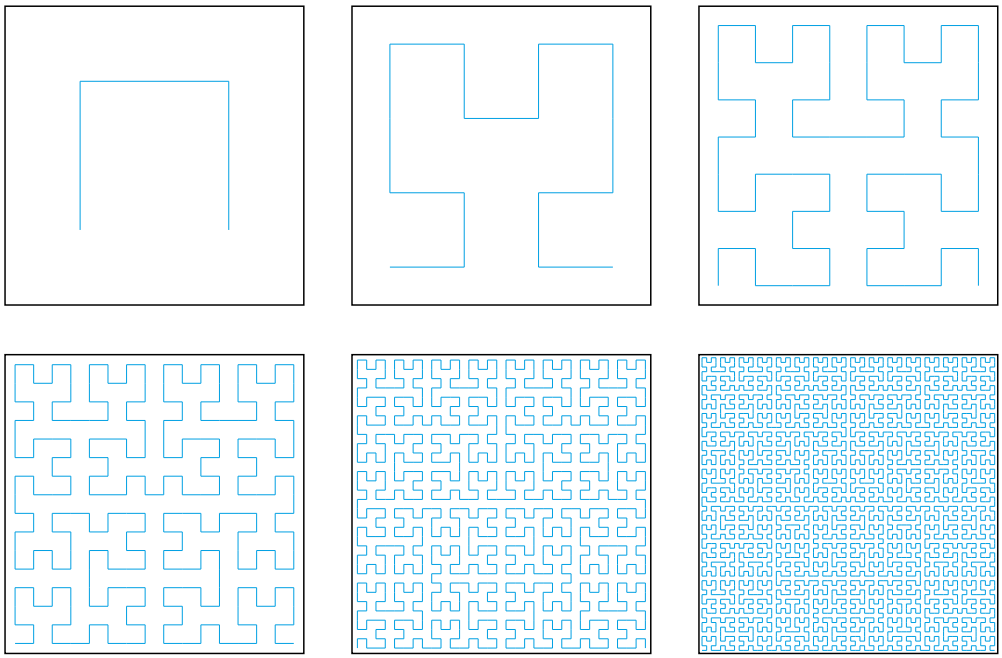


Figure 2.13: First six steps of the Hilbert curve

orientations of polyominoes to which the elements belong. Autocorrelation function from function $f(t)$ in general is expressed in the following way:

$$\Psi(\tau) = \int f(t)f(t-\tau)dt. \tag{2.22}$$

For a discrete vector it can be rewritten as a sum:

$$\Psi(\tau) = \sum_{i=\tau}^{l_h} \left[\vec{s}(i) \times \vec{s}(i-\tau)\right]. \tag{2.23}$$

In order to calculate self-similarity of the vector we have to compute an integral from the autocorrelation function for all τ:

$$R = \int \Psi(\tau)d\tau. \tag{2.24}$$

Return back to the sum:

$$R = \sum_{\tau=1}^{l_h}\sum_{i=\tau}^{l_h} \left[\vec{s}(i) \times \vec{s}(i-\tau)\right]. \tag{2.25}$$

Here we need to describe the multiplication of vectors more exactly. First of all, the vector contains orientations of not all the elements, but only centres of polyominoes. But it does not become shorter: all other elements are set equal to -1. Secondly, we want to find not the product of numbers of orientations, but the number of their coincidences. That is why multiplication is replaced with conditional intersection. Sidelobe level suppression is inversely proportional to self-similarity of the structure. In order to make SLL and self-similarity directly proportional the minus sign is added:

$$R = -\sum_{\tau=1}^{l_h}\sum_{i=\tau}^{l_h}\left[\vec{s}(i)\otimes\vec{s}(i-\tau)\right], \tag{2.26}$$

$$\vec{s}(i)\otimes\vec{s}(j) = \begin{cases} 1, & \begin{cases}\vec{s}(i)=\vec{s}(j)\\ \vec{s}(i)\neq -1\end{cases} \\ 0 & \text{otherwise}\end{cases}. \tag{2.27}$$

2.3.1 Examples of self-similarity values by the autocorrelation method

Below two examples of 32×32 structures are provided, for which the values of self-similarity and sidelobe levels are calculated.

Example 1: structure 32 × 32, L-tromino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-tromino.

Figure 2.14 shows the structure, numerical results are provided in table 2.5.

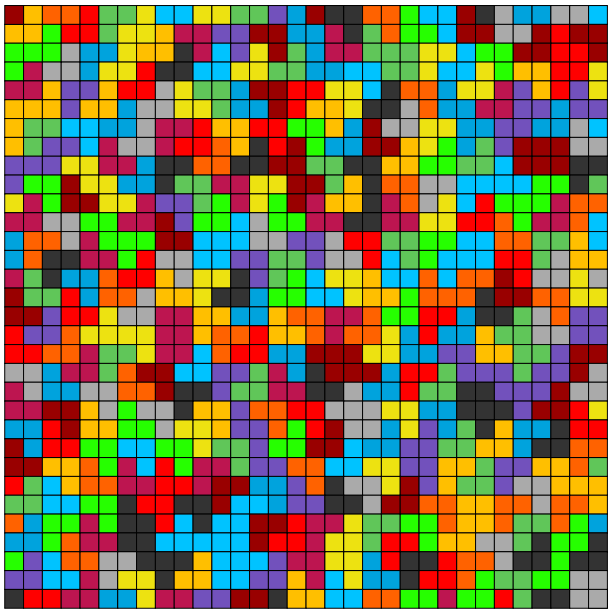


Figure 2.14: Structure of an array in the first example

Table 2.5: Output data of the first example

Parameter	Value
Number of polyominoes α	363
Fullness of the structure A, %	100
Self-similarity R	−7527
Sidelobe level γ at band $r = 1.300$, dB	−28.43
Sidelobe level γ at band $r = 1.818$, dB	−21.76

Example 2: structure 32 × 32, L-octomino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-octomino.

Figure 2.15 shows the structure, numerical results are provided in table 2.6.

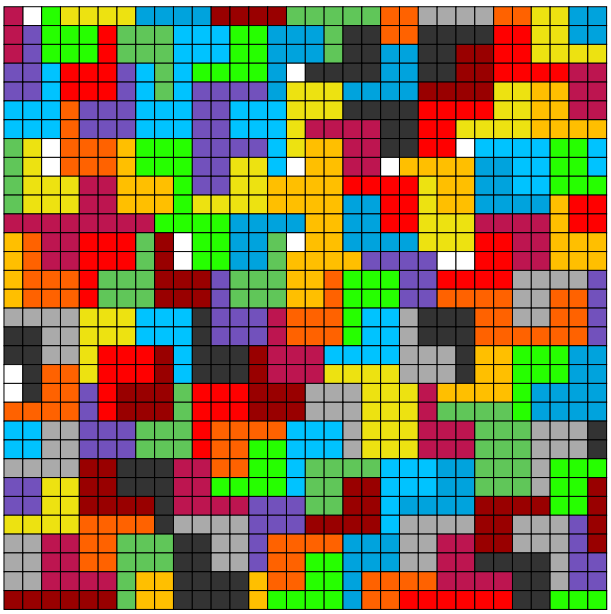


Figure 2.15: Structure of an array in the second example

Table 2.6: Output data of the second example

Parameter	Value
Number of polyominoes α	144
Fullness of the structure A, %	98.63
Self-similarity R	−902
Sidelobe level γ at band $r = 1.300$, dB	−19.97
Sidelobe level γ at band $r = 1.818$, dB	−10.76

2.3.2 Autocorrelation method analysis

Autocorrelation method analysis will be done over two criteria. Firstly, the dependence will be analysed of the sidelobe level on the structure self-similarity. Secondly, the stability of the self-similarity value will be judged, i.e. how much does it vary for similar structures.

Figure 2.16 shows graphs of the self-similarity value calculated by the autocorrelation method and the sidelobe level obtained by numerical simulation for band $r_{sim} = 1.3$. The experiments were run for structures of 32×32 elements, tiled with polyominoes of different types. Numerical data is presented in table 2.7.

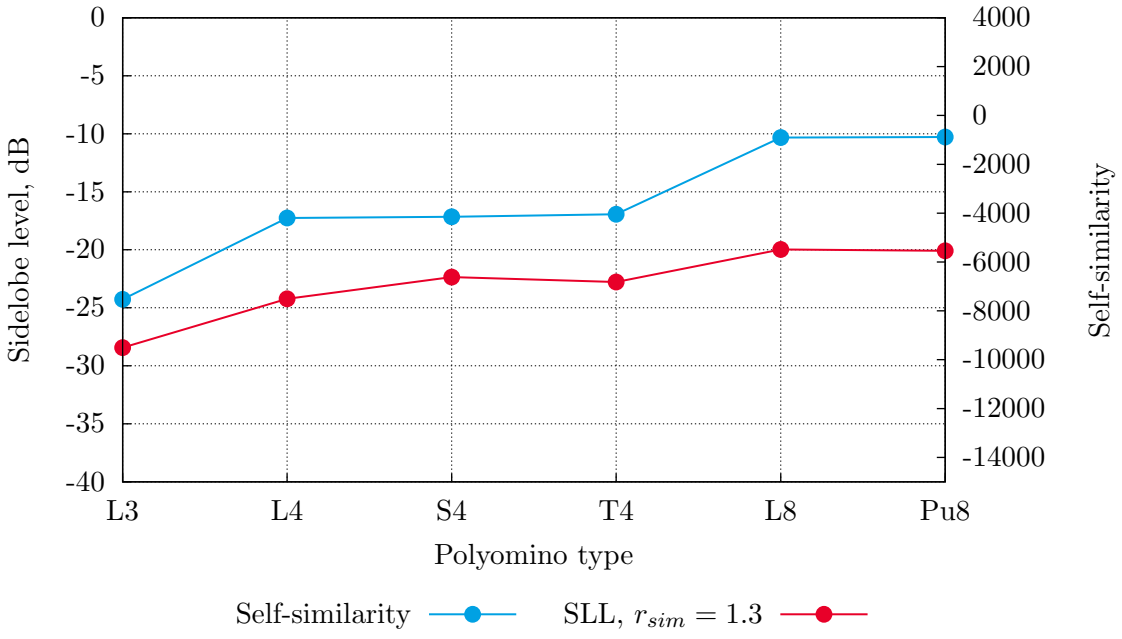


Figure 2.16: Self-similarity and sidelobe level of 32×32 structures of different polyominoes

Similar analysis for structures 64×64 is shown on graph 2.17 and in table 2.8.

It can be seen from the graphs that the self-similarity value behaves very similarly to the sidelobe level. Now we need to investigate, how much do the values vary for the similar structures, i.e. for the structures with

Table 2.7: Self-similarity and SLL of 32×32 structures of different polyominoes

Polyomino type	Self-similarity R	SLL at $r_{sim} = 1.3$, dB
L-tromino	-7527	-28.43
L-tetromino	-4196	-24.22
S-tetromino	-4146	-22.35
T-tetromino	-4042	-22.78
L-octomino	-902	-19.97
Pu-octomino	-885	-20.09

Table 2.8: Self-similarity and SLL of 64×64 structures of different polyominoes

Polyomino type	Self-similarity R	SLL at $r_{sim} = 1.3$, dB
L-tromino	-118749	-32.76
L-tetromino	-66907	-28.67
S-tetromino	-66491	-28.83
T-tetromino	-62545	-29.21
L-octomino	-15444	-25.01
Pu-octomino	-15208	-26.33

more or less the same SLL. For this purpose experiments were run with different seeds of the pseudo random number generator.

Below the graphs are listed that compare self-similarity with sidelobe level of structures, built with different seeds. SLL was simulated for bands $r_{sim} = 1.3$ and $r_{sim} = 1.818$. Structures have equal sizes of 32×32. Figure 2.18 shows the graph for structures tiled with L-tromino while figure 2.19 — for structures tiled with L-octomino.

It can be seen from the graphs that the self-similarity values are stable, i.e. little deviations of SLL correspond to little deviations of self-similarity. This means that the value of self-similarity can be used as a replacement for SLL.

Still we see that different sizes of structures and different polyomino types produce different ranges of self-similarity values. To make use of this value in the cost function it should be normalized. Table 2.9 provides

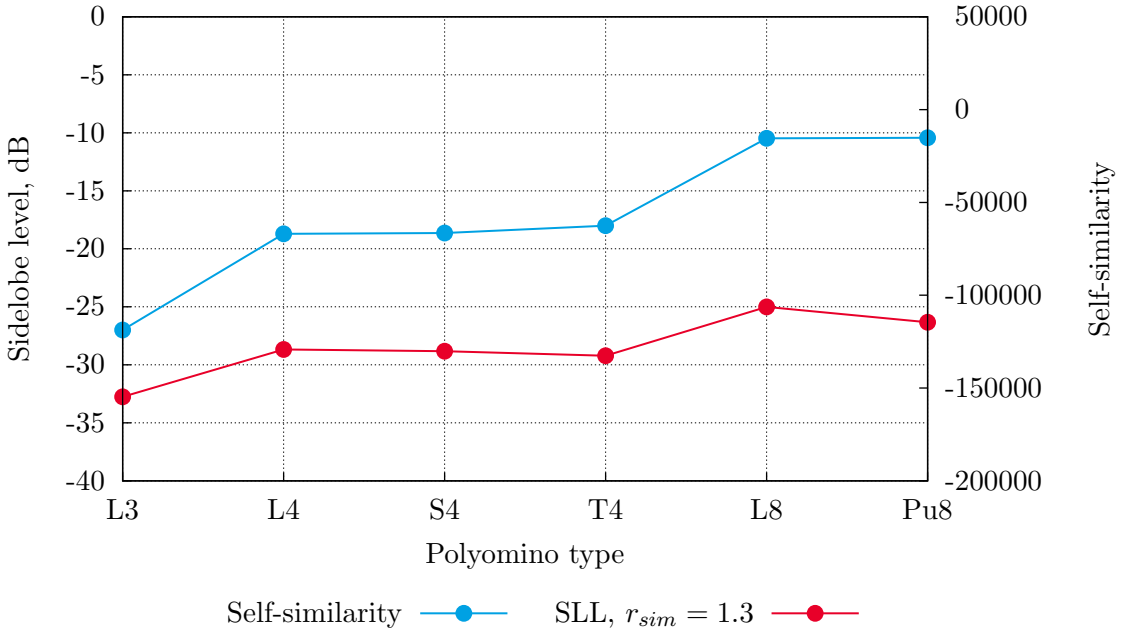


Figure 2.17: Self-similarity and sidelobe level of 64×64 structures of different polyominoes

normalizing denominators for different structure sizes and types of polyominoes. In the cost function this component will be added in a form of a weighted normalized value:

$$\psi_R \frac{R(\vec{s})}{\rho} \tag{2.28}$$

Table 2.9: Normalizing denominators ρ

	16×16	32×32	64×64
L-tromino	-500	-7500	-100000
L-tetromino	-250	-4000	-60000
S-tetromino	-250	-4000	-60000
T-tetromino	-250	-4000	-60000
L-octomino	-60	-1000	-15000
Pu-octomino	-60	-1000	-15000

Therefore, an autocorrelation method has been developed, that allows estimation of self-similarity of structures. Provided analysis of the method showed that self-similarity values correlate with computed values of side-

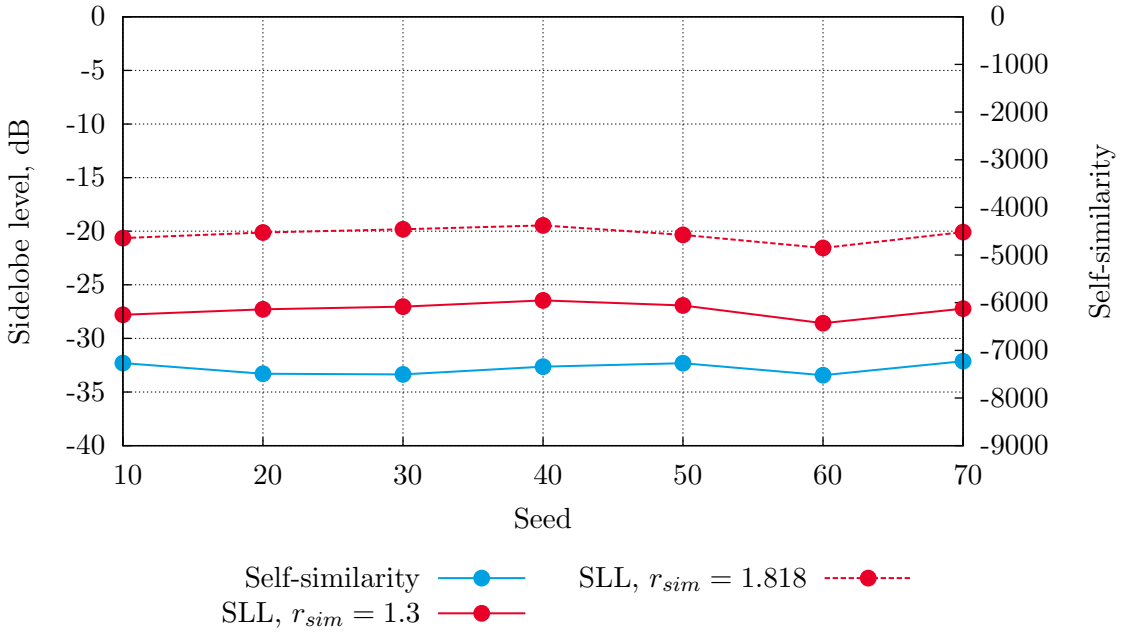


Figure 2.18: Self-similarity and sidelobe level of structures of same size tiled with L-trominoes

lobe levels. This lets us use this method during the process of optimization of phased antenna arrays.

2.4 Chapter 2 conclusions

1. A mathematical model of a structure of polyominoes, representing a phased antenna array, was developed that describes and joins geometrical and electrodynamic relations between the elements. The model takes into account technical features of radiating structures. Radiation properties of the structures are described by array factor.

2. A colour filtering method has been developed that provides numerical estimation of irregularity of a structure. The method was implemented and analysed. As a result of the analysis it was found out that the method cannot be used for PAA optimization, because the derivable

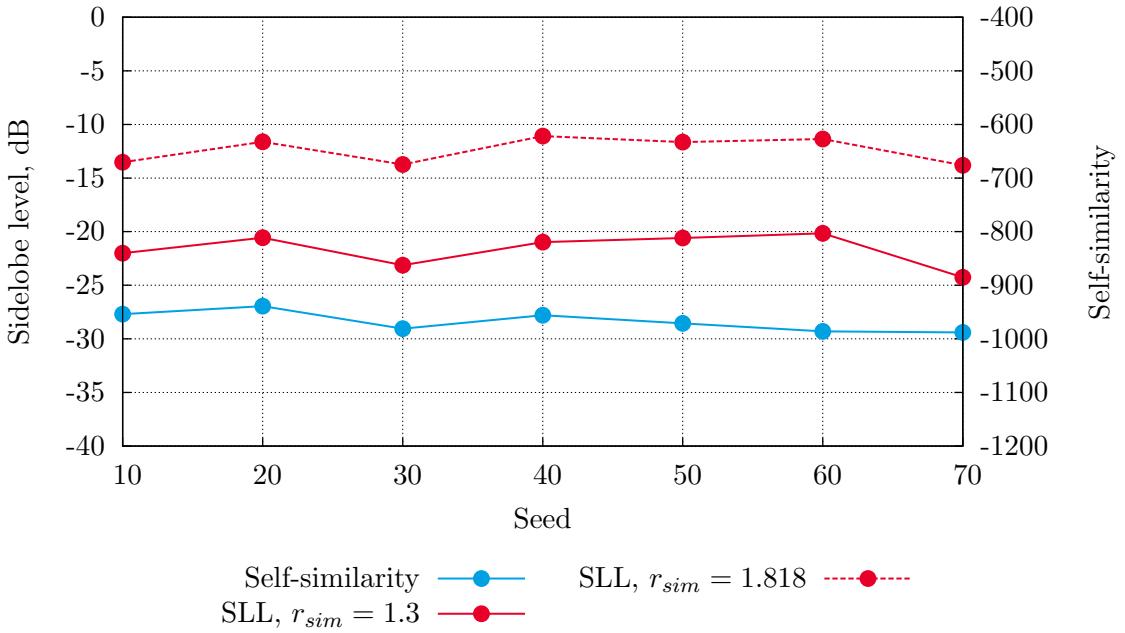


Figure 2.19: Self-similarity and sidelobe level of structures of same size tiled with L-octominoes

irregularity estimation does not correlate with the sidelobe level.

3. An autocorrelation method has been developed, that allows estimation of self-similarity of structures. Provided analysis of the method showed that self-similarity values correlate with computed values of sidelobe levels. This lets us use this method during the process of optimization of phased antenna arrays.

Chapter 3

Development of an optimization algorithm based on the genetic algorithm

3.1 Problem of the genetic algorithm application

In recent years, genetic algorithm proved to be a powerful way of solving problems. Strategy of the GA is based on the principle of natural selection and genetic recombination. GA starts with an initial population of individual chromosomes. Each chromosome with its cost function value is a separate solution in the search space. For each generation chromosomes are being selected from the previous population according to a probability that is proportional to their cost function value. These chromosomes undergo genetic operations to get the new population. During the formation of a new generation, the best chromosomes are recombined in order to produce even better offspring.

The first empirical study of GA in optimization was done by Holland [41] for two-armed bandit problem. Since then, GA was applied to many complex optimization problems in various fields, especially in those where the search space is poorly understood. Some examples of GA for practical

engineering tasks include monitoring gas pipes [36], robotics control [27], scheduling [28], medical image analysis [37] and designing adaptive fuzzy controllers logic [44, 53]. There are several studies on the use of GA in the optimal packing [9, 10].

Success of the GA depends on many factors. It is important that the solution could be represented as a binary string, suitable for manipulation using various genetic operations inside GA. Selecting string structure depends largely on the characteristics of the optimization problem. Such a representation should ideally cover all the search space without violating constraints. In the case of polyomino placement it is more efficient to encode configuration of polyominoes by changing the position of the corresponding gene in the string, especially for large structures.

When determining points of global optimality in the search space, GA does not use any additional information. So no matter whether a task is uni-modal, multi-modal or combinatorial. Necessary information is encoded in the cost function. Basically, the cost function is used to calculate the suitability of solutions. Suitability is the basis for deciding which chromosomes in the population will produce offspring in the next generation. This is a quality score of possible solutions to the problem. Thus, the choice of an appropriate cost function, which expresses the suitability of a potential solution is critical.

Figure 3.1 shows a flow-chart that indicates the steps of optimal placement problem adaptation for GA. In general, the scheme reflects a implementation methodology for GA from a problem formulation to obtaining a complex object-oriented algorithm.

To state the problem to the algorithm clearly, it is necessary to own all the necessary information regarding the specifics of the area and restrictions on the search. This applies, among other things, also systems including the human factor [33]. In the polyomino placement problem

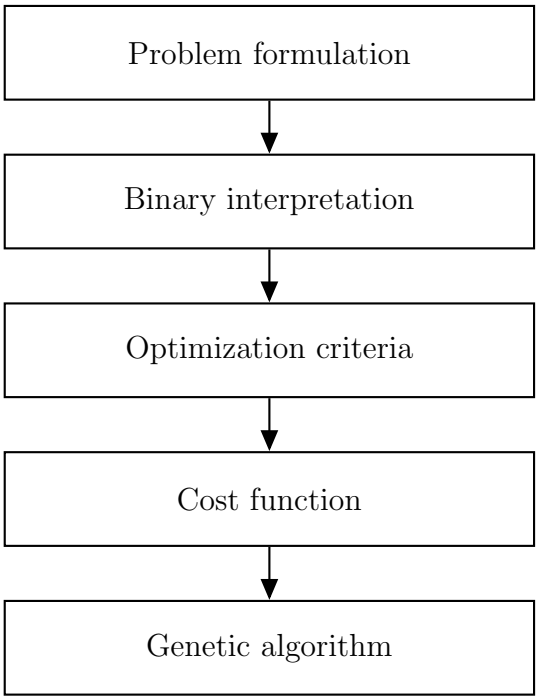


Figure 3.1: Optimal placement problem adaptation for GA

this information should include the number of polyominoes for placement, shape and orientation of each polyomino, area of the structure and all the criteria that must be met under restrictions. The next step is to choose a suitable coding scheme.

After choosing a coding scheme, optimization criteria are considered. They are used during search by the cost function. In the formulation of the objective function all the criteria are converted into maximizations of the functions of some system variables. Then their combination forms the objective function, which is used in the search. For this purpose it is proposed to merge all of the criteria to form a single scalar objective function. The idea is to get a set of weighting coefficients for all system variables. Once the objective function is formulated, the object-oriented GA is ready. Various genetic operators and parameters are carefully calibrated to meet the applicable limitations and problems.

Next will be given a description of a polyominoes placement algorithm for rectangular areas, developed by Gwee and Lim [38], which is the basis for the algorithm developed in this work. Chromosome decoding, cost function choosing and calibration of GA parameters will be addressed. At the end of this section the results of implementation of the algorithm in the program code with examples of structures and their characteristics will be provided. Main results from this chapter are published by the author in journals and conference proceedings [19, 22, 20, 25, 24, 15].

3.1.1 Circular placement principle

In the GA choice of a binary interpretation depends on the characteristics of the optimization problem. Consider the case of placing m polyomino on two-dimensional chessboard. We can encode the position of a polyomino by rearrangements in chromosome, where each gene is a unique number to identify a particular polyomino. Position of a gene in a chromosome dictates a polyomino position on the chessboard. Let $\vec{x}$ denote a chromosome representing positions of m polyominoes. Structure of the string $\vec{x}$ can be written as a concatenation of the following:

$$\vec{x} = (\mu_1 \ \mu_2 \ \ldots \ \mu_m)\,, \tag{3.1}$$

where μ_i represents i-th gene of the chromosome $\vec{x}$, $1 \leqslant \mu_i \leqslant m$.

Two requirements should be met to decode a chromosome with arranged genes:

- polyominoes should not overlap;
- all polyominoes should be placed on the board.

These restrictions mean that in the search for the optimal placement, each polyomino is located on a chessboard without overlapping other poly-

ominoes. Optimal solution is achieved when all polyominoes successfully placed without overlapping.

As a general example of a circular placement, figure 3.2 shows how to organize polyominoes designated as μ_1, μ_2, ..., μ_m. As shown in the figure, μ_1 with its orientation is placed as close as possible to the upper left corner. After that μ_2, μ_3 and μ_4 with their orientations are placed in the lower left, lower right and upper right corners respectively. Similarly, the others are pushed towards the corresponding polyomino boundaries according to the arrows. Each polyomino during placement should not overlap existing polyominoes. Figure 3.3 shows a tree diagram that corresponds to the implied sequence of placing polyominoes in quadrangular region [11]. The number of branches in the diagram corresponds to the number of corners. Circular placement works not only with quadrangular area, it may be efficient also when the object is placed on a plane defined by the polygon.

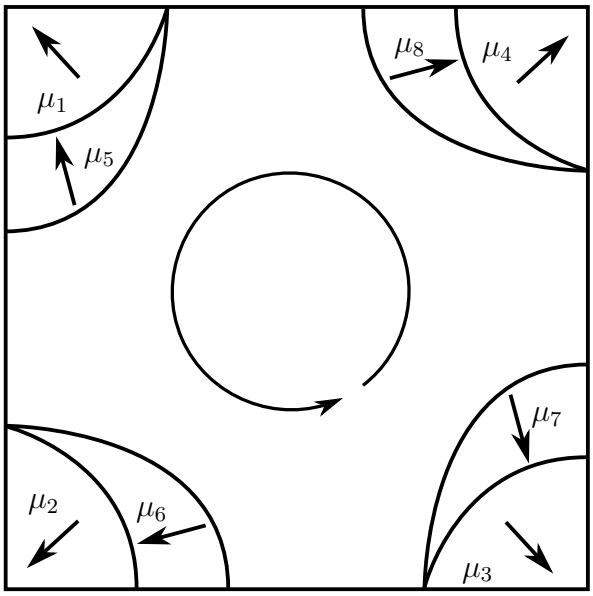


Figure 3.2: Circular placement of the Gwee—Lim algorithm

Extraction technique of polyomino positions on the chessboard from the

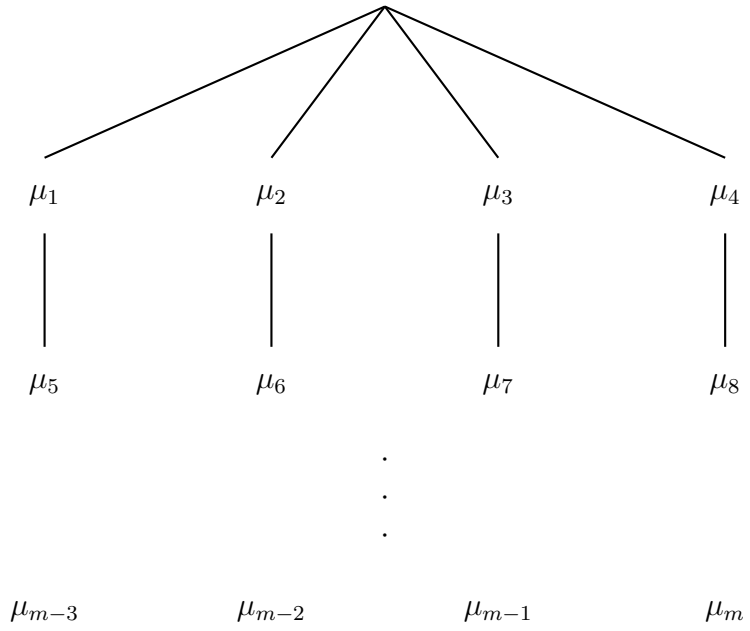


Figure 3.3: Tree diagram of the Gwee—Lim algorithm

chromosome is called circular placement. The idea is to place polyominoes with a predetermined orientations from the corners towards the centre in a counter-clockwise direction. Given the rotation and flipping, there are eight different orientations for polyominoes. It is believed that the orientation of each polyomino contributes the largest number of common sides. Number of common edges between the two polyominoes equals the number of cells of the first polyomino, which lie next to the cells of the second polyomino. Those edges of polyominoes that are in contact with the borders are considered as common edges too. Border edges can be easily understood if we consider border as an imaginary frame around the area.

Circular placement has some advantages over other methods of placing polyominoes, such as top to down or left to right. For them, the configuration is being built starting from a certain point. Changing positions of previously established polyomino entails significant changes in the whole structure. This problem is less likely for a circular placement where poly-

ominoes are positioned from four corners. Circular placement first puts polyominoes, suitable for placement in corners. During the search, groups of polyominoes are made up from corners. For the direction of arrangement from the corners to the centre, the method is called centripetal circular placement.

3.1.2 Fitness function

Measure of fitness of the chromosome is determined by the fitness function. It is a measure of optimality of a possible solution represented in a chromosome. Generally speaking, the more fit chromosome suggests a better configuration, in which polyominoes are located closely to each other. Although the study of the best region in the search space is performed using a selection mechanism, the effectiveness of search is heavily dependent on the formulation of the objective function for evaluation of the quality of chromosomes. It provides the necessary control over the direction of the search towards the best of the region during GA search.

The objective function can be used to interpret complex tasks when there is more than one optimization criterion. In order to obtain optimal solutions that satisfy all the criteria it is necessary to consider all of them in the formation of the target function. For the problem of locating m polyominoes, three system variables are used to represent the problem :

- α — number of polyominoes successfully placed in the structure without overlapping;
- β — number of border edges;
- ω — number of common edges among all polyominoes.

These variables are used in the formulation of the objective function. Value α is the number of non-overlapping polyominoes that have been

successfully placed in the structure. The maximum value of α is m, i.e. the total number of polyominoes for arrangement. Likewise, for any given configuration, β — number of edges of polyominoes that are in contact with the border of the structure. This allows us to determine the suitability of polyominoes for insertion into a corner. To get the ω, the number of common edges of polyominoes in the entire configuration is added to β. A common side is one that is between two elements of the neighbouring polyominoes. Together, the three system variables allow us to estimate the overall quality of the filling, defining suitable polyominoes for border area, the compatibility between the polyominoes and proximity to the goal of placing all polyominoes.

As an illustration of how these variables are calculated, consider figure 3.4, which shows part of the configuration with three polyominoes: C-octomino, L-tetromino and L-octomino. C-octomino has 7 border edges indicated by the numbers 1, 2, 3, 11, 12, 13 and 14 in the upper left corner and 14 common sides, numbered 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13 and 14. Note that the border edges are considered as common sides too.

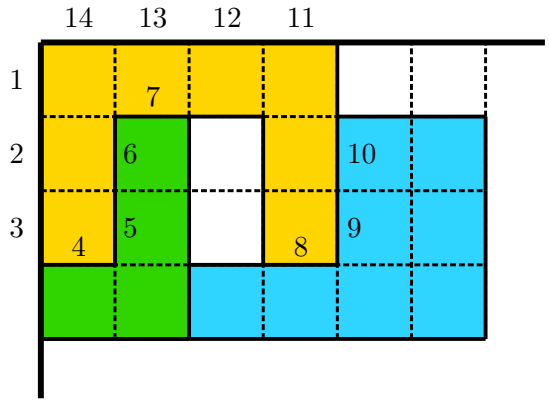


Figure 3.4: Clarification of border edges and common edges [38]

Simple way to formulate the objective function is to combine all relevant system variables into a scalar function. However, the difference in the values of the variables can seriously distort the distribution of chromosomes

in the search space, if they are combined without scaling. To avoid such inadvertent systematic errors, all the variables are scaled to range from 0 to 1. This is achieved by normalization of the variables by $\alpha_{\max}$, $\beta_{\max}$ and $\omega_{\max}$ — maximum values of α, β and ω accordingly. Let us define normalized variables as $\bar{\alpha}$, $\bar{\beta}$ and $\bar{\omega}$ in the following way:

$$\begin{aligned} \bar{\alpha}\left(\vec{x}\right) &= \frac{\alpha\left(\vec{x}\right)}{\alpha_{\max}}, \\ \bar{\beta}\left(\vec{x}\right) &= \frac{\beta\left(\vec{x}\right)}{\beta_{\max}}, \\ \bar{\omega}\left(\vec{x}\right) &= \frac{\omega\left(\vec{x}\right)}{\omega_{\max}}, \end{aligned} \tag{3.2}$$

where $\vec{x}$ corresponds to the given configuration of a set of polyominoes.

Thus, the objective function is formulated in relation to the three relevant criteria of polyomino placement. Objective function can be described as the maximization of some linear combination of these criteria:

$$\uparrow C\left(\vec{x}\right) = \left[\bar{\alpha}\left(\vec{x}\right),\ \bar{\beta}\left(\vec{x}\right),\ \bar{\omega}\left(\vec{x}\right)\right] \tag{3.3}$$

For all of these criteria to be considered equal, each criterion is assigned a weighting factor. Thus, the optimization can be performed based on a common target function. It is calculated as a linear combination of criteria. Importance of each criterion is adjusted in the expression by its weight. The general expression for the objective function for the problem of polyomino placement will be written as:

$$C\left(\vec{x}\right) = \psi_{\alpha}\bar{\alpha}\left(\vec{x}\right) + \psi_{\beta}\bar{\beta}\left(\vec{x}\right) + \psi_{\omega}\bar{\omega}\left(\vec{x}\right), \tag{3.4}$$

where ψ_{α}, ψ_{β} and ψ_{ω} are the weighting coefficients. It should be noted that relevant and not absolute values of ψ_{α}, ψ_{β} and ψ_{ω} will affect the search. Still, without any knowledge of how these criteria are interconnected, it is difficult to choose appropriate set of weighting coefficients in order to formulate well the cost function.

Gwee and Lim in their work run a series of experiments and obtain optimal values for weighting coefficients [38]:

$$\begin{aligned} \psi_\alpha &= 0.22, \\ \psi_\beta &= 0.45, \\ \psi_\omega &= 0.33. \end{aligned} \tag{3.5}$$

3.1.3 Calibration of GA parameters

GA has stochastic nature and its efficiency depends on values of several special parameters. For each particular task there is an optimal set of those parameters that provides the best convergence. That includes probability of crossover p_c, probability of mutation p_m and probability of mutation of a bit p_{bm}. In this paragraph we will describe the process of calibration of these parameters in order to achieve the maximal efficiency of the algorithm [34].

In [90] Weile and Michielssen say that, because of the incomplete nature of genetic algorithm theory, much knowledge about successful implementation of GA comes from experience and experiment. Thanks to this experience, they achieve the best ranges of values for each parameter of the GA. In our implementation we only invert the ranges between p_m and p_{bm} with respect to [90]:

$$\begin{aligned} 0.6 < p_c < 0.9, \\ 0.1 < p_m < 0.3, \\ 0.001 < p_{bm} < 0.1. \end{aligned} \tag{3.6}$$

We should try all possible combinations of these three parameters and then choose the combination which gives the best value of fitness function. Trying all possible combinations is too much expensive in terms of time. The larger the size of the array, more time is required by the tiling

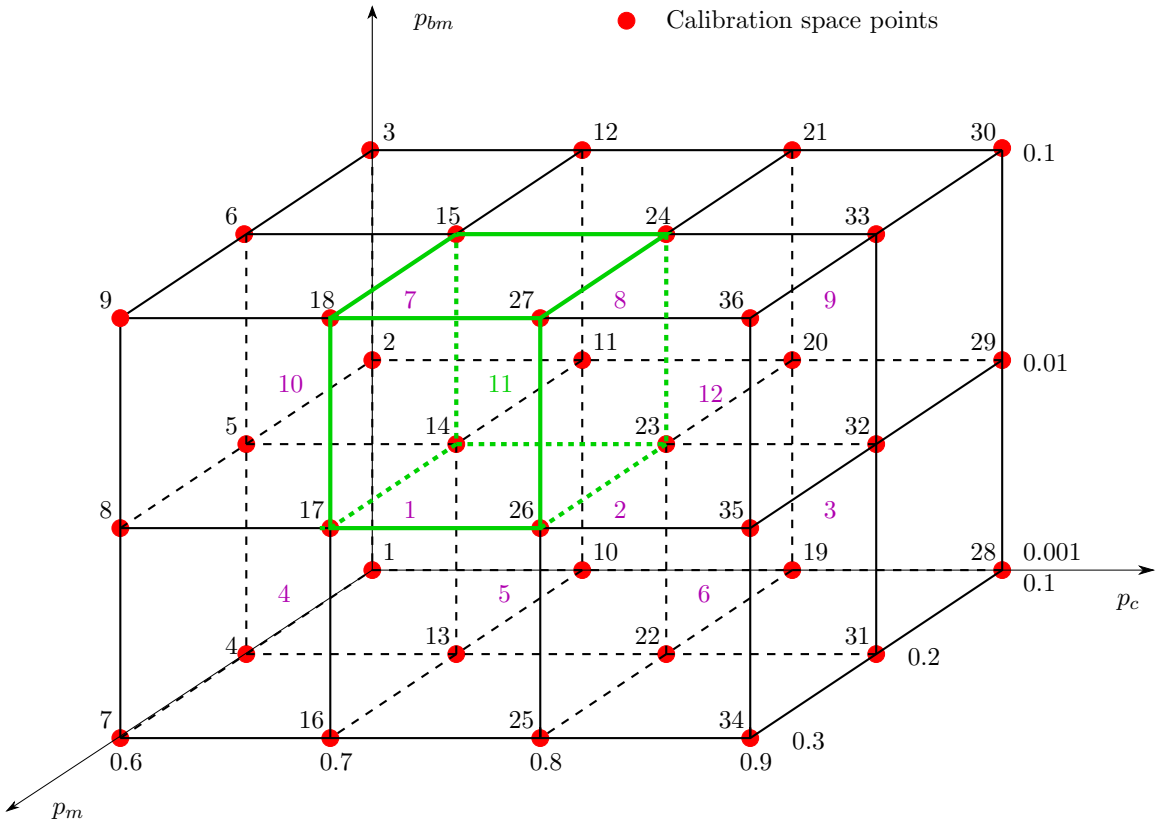


Figure 3.5: Calibration space

algorithm. Hence, we divide each interval in steps where we evaluate the fitness function:

$$\begin{aligned} p_c &\in \{0.6,\ 0.7,\ 0.8,\ 0.9\}, \\ p_m &\in \{0.1,\ 0.2,\ 0.3\}, \\ p_{bm} &\in \{0.001,\ 0.01,\ 0.1\}. \end{aligned} \tag{3.7}$$

Performing the tests for all those combinations, we achieve 36 test results (table 3.1). Figure 3.5 shows the generic initial calibration tests space, where the parameters are the axes of the coordinates system. As you can see, red points represent the tests computed: for each test, the parameters correspond to the coordinates of the point. Numbers in purple define the cubes index.

GAs are stochastic algorithms which generate and use random variables. We are dealing with a pseudo random number generator (PRNG)

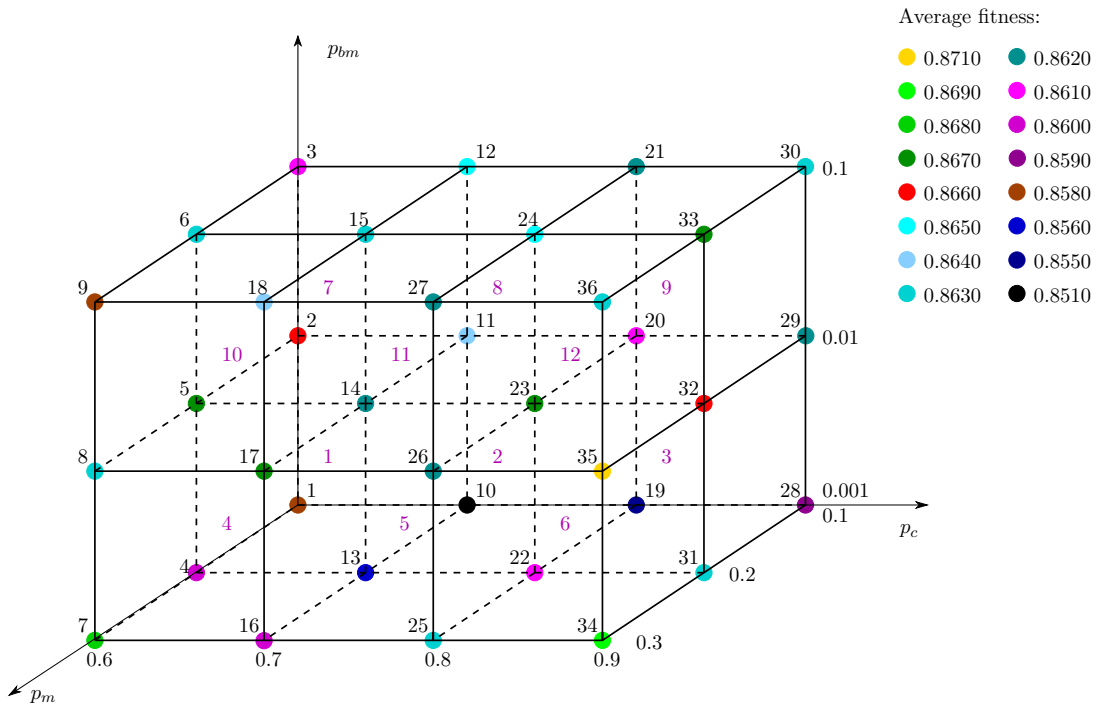


Figure 3.6: The first step of GLA calibration

which should be initialized to a starting state using a particular value called seed. If PRNG is initialized always with the same seed value, it will produce always the same sequence of random values. For this reason, we compute tests with different values of seed for each point of the initial calibration space. We use $N_s = 10$ different values of s: $s \in \{10, 20, \ldots, 100\}$. In total we get N_s solutions for each point i in the space: $\Phi_i^s \in \left\{\Phi_i^{10}, \Phi_i^{20}, \ldots, \Phi_i^{100}\right\}$. Then, for each point of the initial calibration space we compute the average fitness value (3.8) and standard deviation (3.9):

$$\Phi_i^{avg} = \frac{\sum_{k=1}^{N_s} \Phi_i^{s_k}}{S}, \tag{3.8}$$

$$\sigma_i = \sqrt{\frac{\sum_{k=1}^{N_s} \left(\Phi_i^{s_k} - \Phi_i^{avg}\right)^2}{S}}. \tag{3.9}$$

Table 3.1: Average fitness function values at the first step of GLA calibration

i	Φ_i	σ_i	p_c	p_m	p_{bm}	i	Φ_i	σ_i	p_c	p_m	p_{bm}
1	0.8580	0.0125	0.6	0.1	0.001	19	0.8550	0.0103	0.8	0.1	0.001
2	0.8660	0.0092	0.6	0.1	0.01	20	0.8610	0.0054	0.8	0.1	0.01
3	0.8610	0.0070	0.6	0.1	0.1	21	0.8620	0.0098	0.8	0.1	0.1
4	0.8600	0.0100	0.6	0.2	0.001	22	0.8610	0.0104	0.8	0.2	0.001
5	0.8670	0.0110	0.6	0.2	0.01	23	0.8670	0.0110	0.8	0.2	0.01
6	0.8630	0.0064	0.6	0.2	0.1	24	0.8650	0.0128	0.8	0.2	0.1
7	0.8680	0.0075	0.6	0.3	0.001	25	0.8630	0.0078	0.8	0.3	0.001
8	0.8630	0.0078	0.6	0.3	0.01	26	0.8620	0.0087	0.8	0.3	0.01
9	0.8580	0.0040	0.6	0.3	0.1	27	0.8620	0.0060	0.8	0.3	0.1
10	0.8510	0.0070	0.7	0.1	0.001	28	0.8590	0.0083	0.9	0.1	0.001
11	0.8640	0.0080	0.7	0.1	0.01	29	0.8620	0.0060	0.9	0.1	0.01
12	0.8650	0.0103	0.7	0.1	0.1	30	0.8630	0.0078	0.9	0.1	0.1
13	0.8560	0.0092	0.7	0.2	0.001	31	0.8630	0.0078	0.9	0.2	0.001
14	0.8620	0.0060	0.7	0.2	0.01	32	0.8660	0.0092	0.9	0.2	0.01
15	0.8630	0.0046	0.7	0.2	0.1	33	0.8670	0.0046	0.9	0.2	0.1
16	0.8600	0.0100	0.7	0.3	0.001	34	0.8690	0.0114	0.9	0.3	0.001
17	0.8670	0.0064	0.7	0.3	0.01	35	0.8710	0.0083	0.9	0.3	0.01
18	0.8640	0.0080	0.7	0.3	0.1	36	0.8630	0.0078	0.9	0.3	0.1

For more accurate calibration we use a zooming approach. Initial calibration space is divided into 12 cubes. The average value of the fitness function Φ_{cj}^{avg} is calculated among 8 vertices belonging to the same cube j, where $j = 1, \ldots, 12$. The a cube j_M is selected with the highest average value $\Phi_{cj_M}^{avg} \geqslant \Phi_{cj}^{avg}\ \forall j$ and considered as a calibration space for the second step.

Let the chosen cube be defined by intervals $p_c \in [a_c; b_c]$, $p_m \in [a_m; b_m]$, $p_{bm} \in [a_{bm}; b_{bm}]$. Parameter ranges for the second step will be expressed as follows:

$$\begin{aligned} p_c &\in \left\{a_c; \tfrac{a_c+b_c}{2}; b_c\right\}, \\ p_m &\in \left\{a_m; \tfrac{a_m+b_m}{2}; b_m\right\}, \\ p_{bm} &\in \left\{a_{bm}; \tfrac{a_{bm}+b_{bm}}{2}; b_{bm}\right\}. \end{aligned} \tag{3.10}$$

Table 3.2: Average fitness function values of cubes at the first step of GLA calibration

j	Φ_{cj}^{avg}	$a_c \div b_c$	$a_m \div b_m$	$a_{bm} \div b_{bm}$
1	0.8605	0.6 ÷ 0.7	0.1 ÷ 0.2	0.001 ÷ 0.01
2	0.8596	0.7 ÷ 0.8	0.1 ÷ 0.2	0.001 ÷ 0.01
3	0.8618	0.8 ÷ 0.9	0.1 ÷ 0.2	0.001 ÷ 0.01
4	0.8629	0.6 ÷ 0.7	0.2 ÷ 0.3	0.001 ÷ 0.01
5	0.8623	0.7 ÷ 0.8	0.2 ÷ 0.3	0.001 ÷ 0.01
6	0.8653	0.8 ÷ 0.9	0.2 ÷ 0.3	0.001 ÷ 0.01
7	0.8639	0.6 ÷ 0.7	0.1 ÷ 0.2	0.01 ÷ 0.1
8	0.8636	0.7 ÷ 0.8	0.1 ÷ 0.2	0.01 ÷ 0.1
9	0.8641	0.8 ÷ 0.9	0.1 ÷ 0.2	0.01 ÷ 0.1
10	0.8634	0.6 ÷ 0.7	0.2 ÷ 0.3	0.01 ÷ 0.1
11	0.8640	0.7 ÷ 0.8	0.2 ÷ 0.3	0.01 ÷ 0.1
12	0.8654	0.8 ÷ 0.9	0.2 ÷ 0.3	0.01 ÷ 0.1

Now there are $3 \times 3 \times 3 = 27$ points. As before, the fitness function is calculated at every point with 10 different seeds. The average values are computed and the one with the highest average is chosen. Its coordinates will be the optimal parameters of GA.

All the experiments use the following parameters:

- number of elements along axis x: $M = 64$;
- number of elements along axis y: $N = 64$;
- type of polyomino: L-shaped octomino;
- number of individuals: $P = 10$;
- number of iterations: $K = 100$.

Figure 3.6 shows the values of the fitness function at each point of the initial calibration space. Table 3.2 provides average values of the fitness function for each of the 12 cubes.

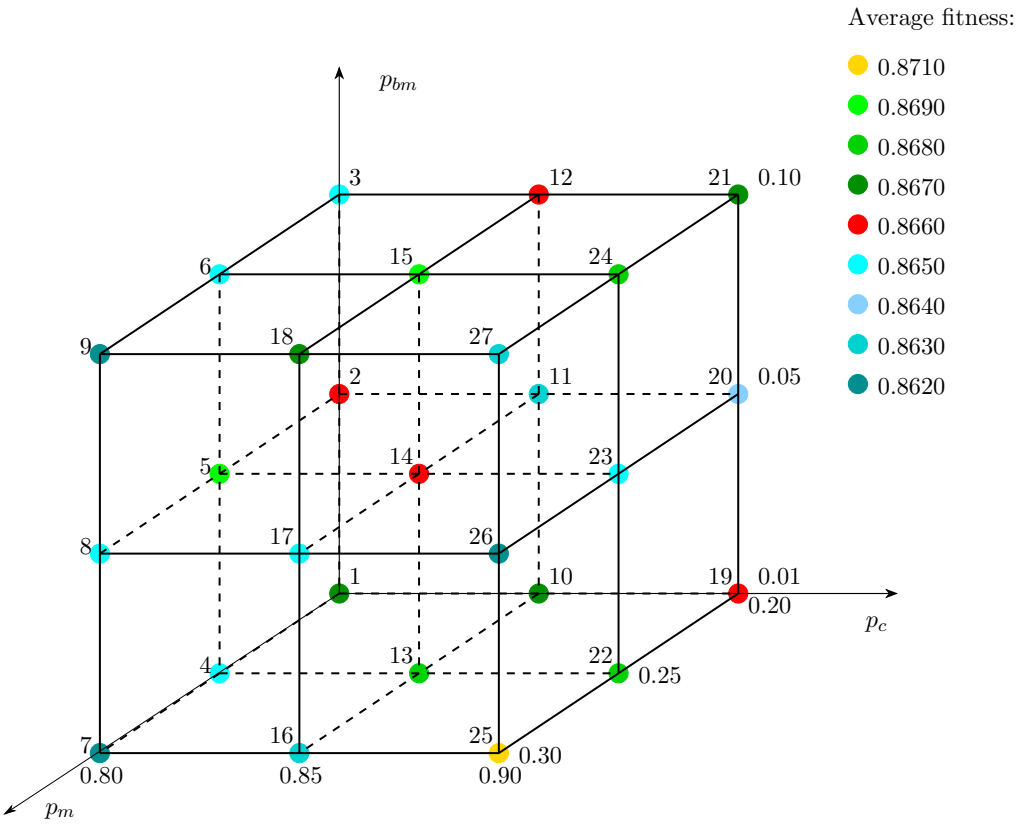


Figure 3.7: Second step of the GLA calibration

The average fitness values of each cube are slightly different from each other. Anyway, we can see that cube number 12 has the best average fitness function value: $\Phi_{c12}^{avg} = 0.8654$. We compute more detailed tests for the chosen cube number 12, following the rules described above. For each point of the new calibration space, we compute the fitness value with different seed values, the average fitness value and the standard deviation value (table 3.3). Figure 3.7 shows average values for all points of the second calibration space.

Point number 25 has the highest average value of the fitness function $\Phi_{25}^{avg} = 0.871$. It corresponds to the set of parameters $p_c = 0.9$, $p_m = 0.3$, $p_{bm} = 0.01$. Those are the calibrated GA parameters for the Gwee—Lim algorithm. Therefore, the optimal GA parameters such as crossover probability, mutation probability and bit mutation probability have been

Table 3.3: Average values of the fitness function at the second step of calibration of GLA

i	Φ_i	σ_i	p_c	p_m	p_{bm}
1	0.867	0.0110	0.80	0.20	0.01
2	0.866	0.0102	0.80	0.20	0.05
3	0.865	0.0128	0.80	0.20	0.10
4	0.865	0.0067	0.80	0.25	0.01
5	0.869	0.0070	0.80	0.25	0.05
6	0.865	0.0067	0.80	0.25	0.10
7	0.862	0.0087	0.80	0.30	0.01
8	0.865	0.0050	0.80	0.30	0.05
9	0.862	0.0060	0.80	0.30	0.10
10	0.867	0.0064	0.85	0.20	0.01
11	0.863	0.0046	0.85	0.20	0.05
12	0.866	0.0080	0.85	0.20	0.10
13	0.868	0.0087	0.85	0.25	0.01
14	0.866	0.0066	0.85	0.25	0.05
15	0.869	0.0083	0.85	0.25	0.10
16	0.863	0.0090	0.85	0.30	0.01
17	0.865	0.0067	0.85	0.30	0.05
18	0.867	0.0078	0.85	0.30	0.10
19	0.866	0.0092	0.90	0.20	0.01
20	0.864	0.0128	0.90	0.20	0.05
21	0.867	0.0046	0.90	0.20	0.10
22	0.868	0.0087	0.90	0.25	0.01
23	0.865	0.0067	0.90	0.25	0.05
24	0.868	0.0075	0.90	0.25	0.10
25	0.871	0.0083	0.90	0.30	0.01
26	0.862	0.0060	0.90	0.30	0.05
27	0.863	0.0078	0.90	0.30	0.10

obtained for the Gwee—Lim algorithm.

3.1.4 Examples of structures by the Gwee—Lim algorithm

In this section we provide four experiments as examples of work of the Gwee—Lim algorithm. The experiments are done with different structure sizes and various polyomino shapes. Following parameters were used:

- number of iterations: $K = 50$;
- population size: $P = 10$;
- seed: $s = 37$.

We used the calibrated values of p_c, p_m and p_{bm}. Figures 3.8 and 3.9 show the graphs of fitness function and its components in the first and second examples.

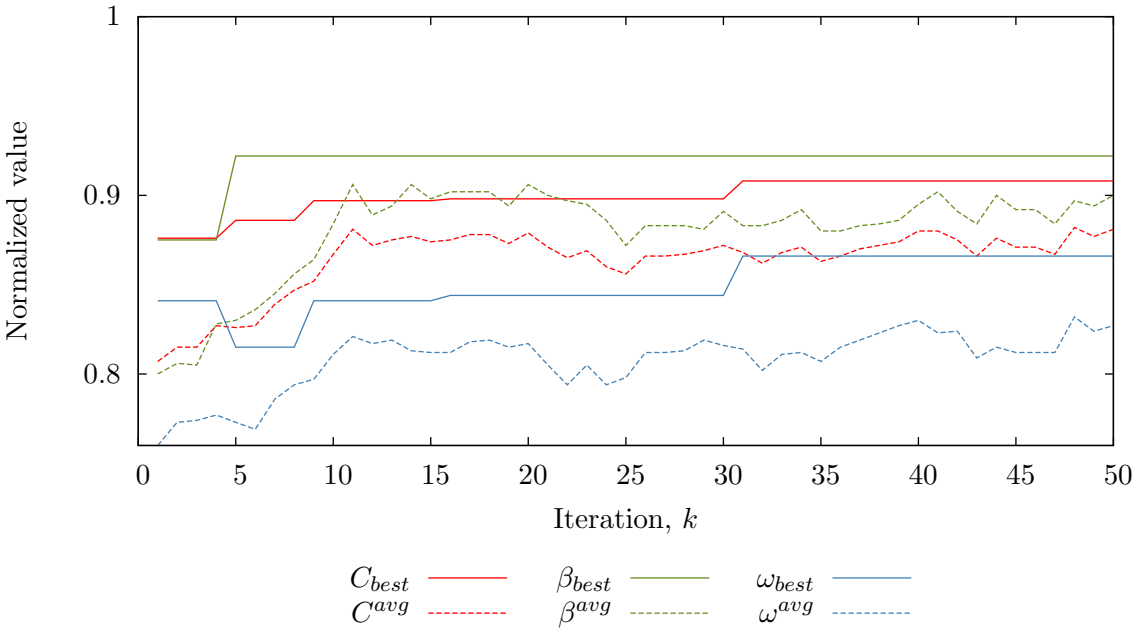


Figure 3.8: Fitness function in the first example

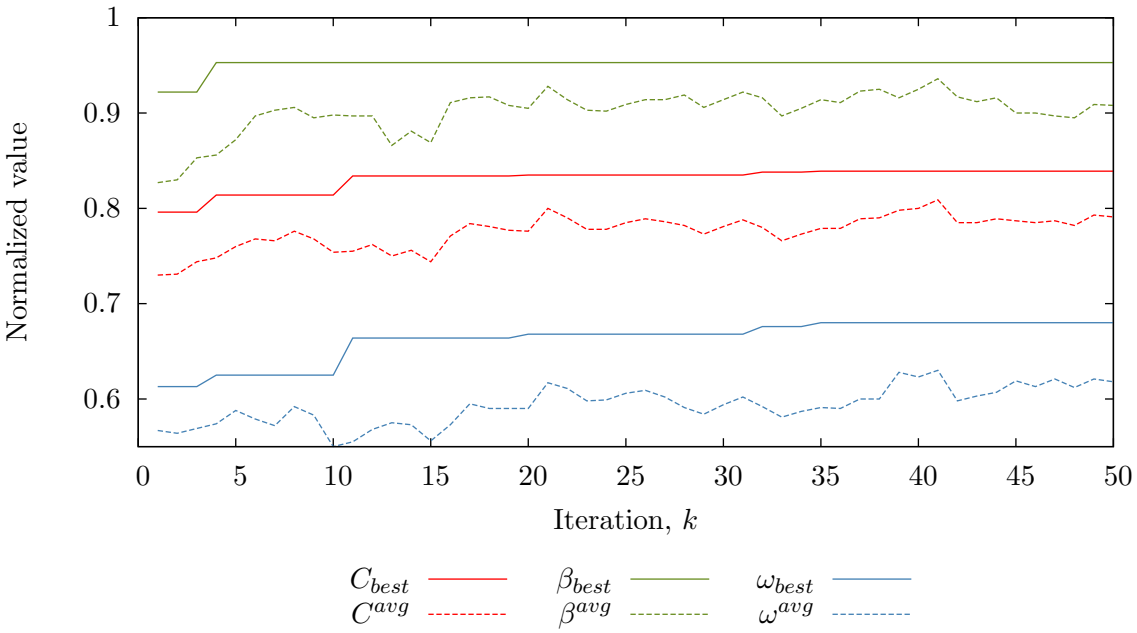


Figure 3.9: Fitness function in the second example

Example 1: Structure 16 × 16, L-tromino

Input parameters:

- structure size: $M = N = 16$;
- polyomino type: L-tromino.

Figure 3.10 shows the structure obtained by GLA with those parameters. Numerical results are provided in table 3.4.

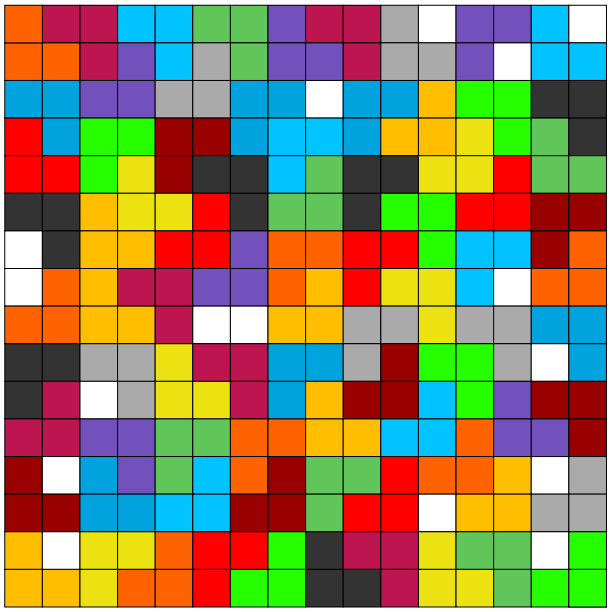


Figure 3.10: Structure of the array in the first example

Table 3.4: Output data of the first example

Parameter	Value
Number of polyominoes α	80
Number of boundary edges β	59
Number of common edges ω	263
Number of holes H	16
Fullness of the structure A, %	93.75

Algorithm shows fair fullness of the structure.

Example 2: Structure 16 × 16, L-octomino

Input parameters:

- structure size: $M = N = 16$;
- polyomino type: L-octomino.

Figure 3.11 shows the structure obtained by GLA with those parameters. Numerical results are provided in table 3.5.

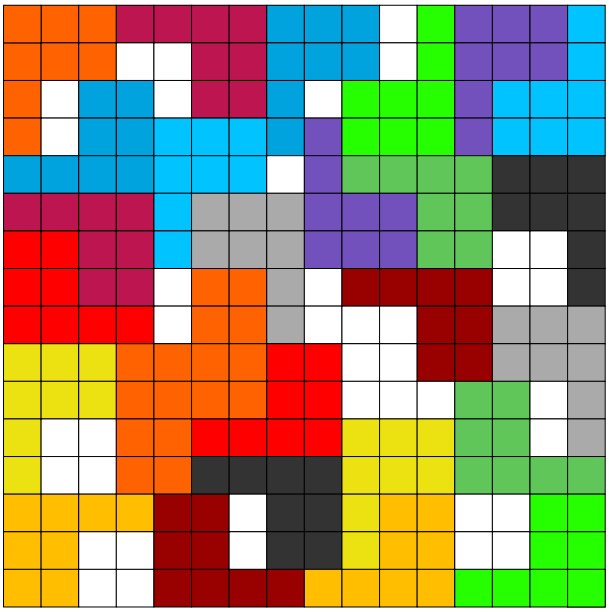


Figure 3.11: Structure of the array in the second example

Table 3.5: Output data of the second example

Parameter	Value
Number of polyominoes α	27
Number of boundary edges β	61
Number of common edges ω	113
Number of holes H	40
Fullness of the structure A, %	84.38

Larger size of polyominoes led to decrease in the fullness of the structure.

Example 3: Structure 32 × 32, L-tromino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-tromino.

Figure 3.12 shows the structure obtained by GLA with those parameters. Numerical results are provided in table 3.6.

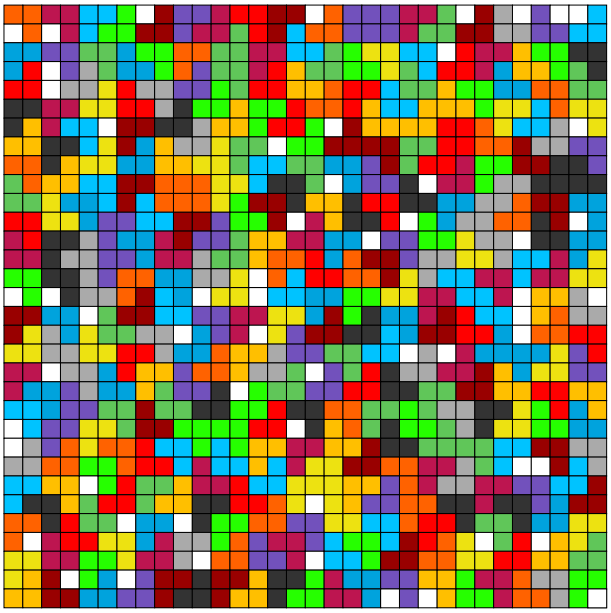


Figure 3.12: Structure of the array in the third example

Table 3.6: Output data of the third example

Parameter	Value
Number of polyominoes α	322
Number of boundary edges β	113
Number of common edges ω	1130
Number of holes H	58
Fullness of the structure A, %	94.34

In this example the fullness by L-tromino got higher because their relative dimensions became smaller.

Example 4: Structure 32 × 32, L-octomino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-octomino.

Figure 3.13 shows the structure obtained by GLA with those parameters. Numerical results are provided in table 3.7.

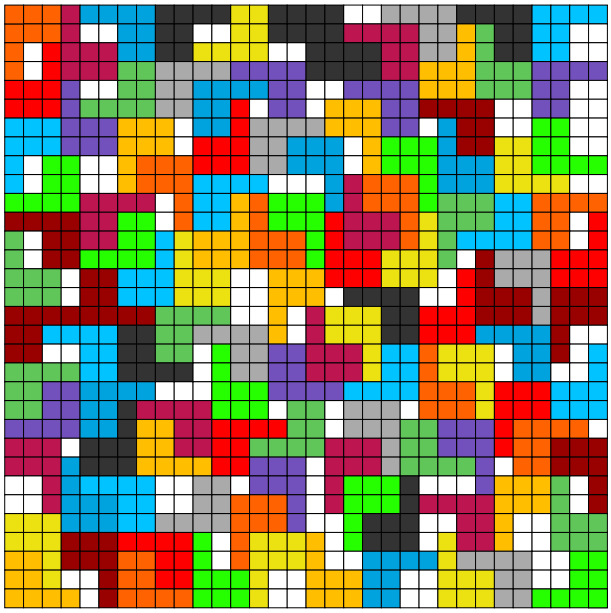


Figure 3.13: Structure of the array in the fourth example

Table 3.7: Output data of the fourth example

Parameter	Value
Number of polyominoes α	111
Number of boundary edges β	110
Number of common edges ω	560
Number of holes H	136
Fullness of the structure A, %	86.72

Similarly, the fullness of the structure has increased but still it is lower than 90%.

Thus, the questions were described that arise with application of the genetic algorithm and sequence of solving provided. The Gwee—Lim algorithm for tiling rectangular structures with polyominoes was described in details including circular placement principle and fitness function. Parameters have been calibrated and example results shown. The analysis of the examples revealed that this algorithm is not able to fill structures with polyominoes well and that it is necessary to develop another algorithm to solve this problem.

3.2 Development of the "Snowball" algorithm for optimization of structures of polyominoes

From the examples by the Gwee—Lim algorithm it can be seen that the fullness of structures composed of octominoes does not exceed 90%. In other words, more than 10% of the structures remains unused. With respect to the antenna arrays, this means a reduction of electrical dimensions, thus reducing the directivity. Using octominoes is also important, since grouping of elements in the subarrays of eight cells reduces the number of delay lines, respectively, eight times.

If we analyse the structure in figure 3.11, we can see that octominoes are relatively large polyominoes. In a limited space they begin to interfere with each other. With centripetal circular placement it is especially evident in the central region of the structures, where colonies of polyominoes growing from different angles meet.

In this regard, an attempt was made to develop another algorithm capable to overcome the above drawbacks.

3.2.1 Mathematical formulation

In this work we consider planar arrays which dimensions are defined as $M \times N$ elements on the x-y plane and inter-element spacing d_x and d_y along the x and y axis. The aperture is being tiled with subarrays of q elements, the maximum number of subarrays that can be used to tile the whole aperture is $Q = {}^{M \times N}/_{q}$. A set $\vec{S}$ of Q or less polyominoes with defined positions and orientations represent a particular structure of the array or, in other words, one of the possible solutions. Optimal solution $\vec{S}_{opt}$ in our case is the one that provides good filling of the aperture of the array together with low sidelobe level. The genetic algorithm works with chromosomes during search of the optimal solution, the final result is represented by a chromosome as well. A chromosome $\vec{x}$ is a vector of binary numbers. Evaluation of a fitness of a solution is performed by computation of the value of a fitness function C.

The algorithm uses such a term as a fitness effect ΔC. The fitness of a solution is the value of the fitness function given that solution:

$$C = f(\vec{x}). \tag{3.11}$$

Let us call the $\vec{x}_n$ a solution containing only n first polyominoes. The corresponding fitness value will be:

$$C_n = f(\vec{x}_n). \tag{3.12}$$

The fitness effect of m-th polyomino is defined as:

$$\Delta C_m = C_m - C_{m-1}. \tag{3.13}$$

It can be shown that the sum of all the fitness effects is equal to the fitness of a whole solution. Let us assume that there are α polyominoes in a whole solution:

$$\vec{x} = \vec{x}_\alpha, \tag{3.14}$$

$$C = f(\vec{x}) = f(\vec{x}_\alpha) = C_\alpha, \tag{3.15}$$

$$\sum_{i=1}^{\alpha} \Delta C_i = \Delta C_1 + \ldots + \Delta C_\alpha = (C_1 - 0) + ... + (C_\alpha - C_{\alpha-1}) = C_\alpha = C. \tag{3.16}$$

In order to evaluate the fitness function, the solution represented by a chromosome should first be decoded. In our case each solution is a particular structure of an array tiled with given shape of irregular polyomino. So the task is to transform a binary vector into a tiled structure. The decoding strategy and fitness function are two main components of our approach called "Snowball" algorithm (SA).

3.2.2 Decoding strategy

The decoding strategy is similar to one proposed by Gwee and Lim [38]. The main novelty of their strategy is a technique called circular placement. They assume that any tiling structure can be represented by a set of orientations $\vec{S} = \{\mu_1,\ \mu_2,\ \ldots \mu_Q\}$ of polyominoes which then are being placed using a predefined rule. Since we use a rectangular grid, there are 4 possible orientations of a polyomino. Also the polyomino can be flipped. Therefore, an asymmetric polyomino can have $N_\mu = 8$ orientations. A symmetric polyomino will have 4 orientations repeated twice. We dedicate $L_g = \log_2 N_\mu = 3$ bits to code N_μ orientations. The tiling strategy can be described in three steps. Corresponding flowchart can be seen in figure 3.14, where P — size of population, $i_{\max}$ — number of iterations. Let us consider a chromosome of length $L = L_g \times Q$:

$$\vec{x} = 100\ 001\ 011\ 111\ 010\ \ldots. \tag{3.17}$$

Step 1 — Extraction of genes

We assume, that the chromosome is a concatenation of orientations in a binary notation. Parts of the chromosomes (genes) that correspond to

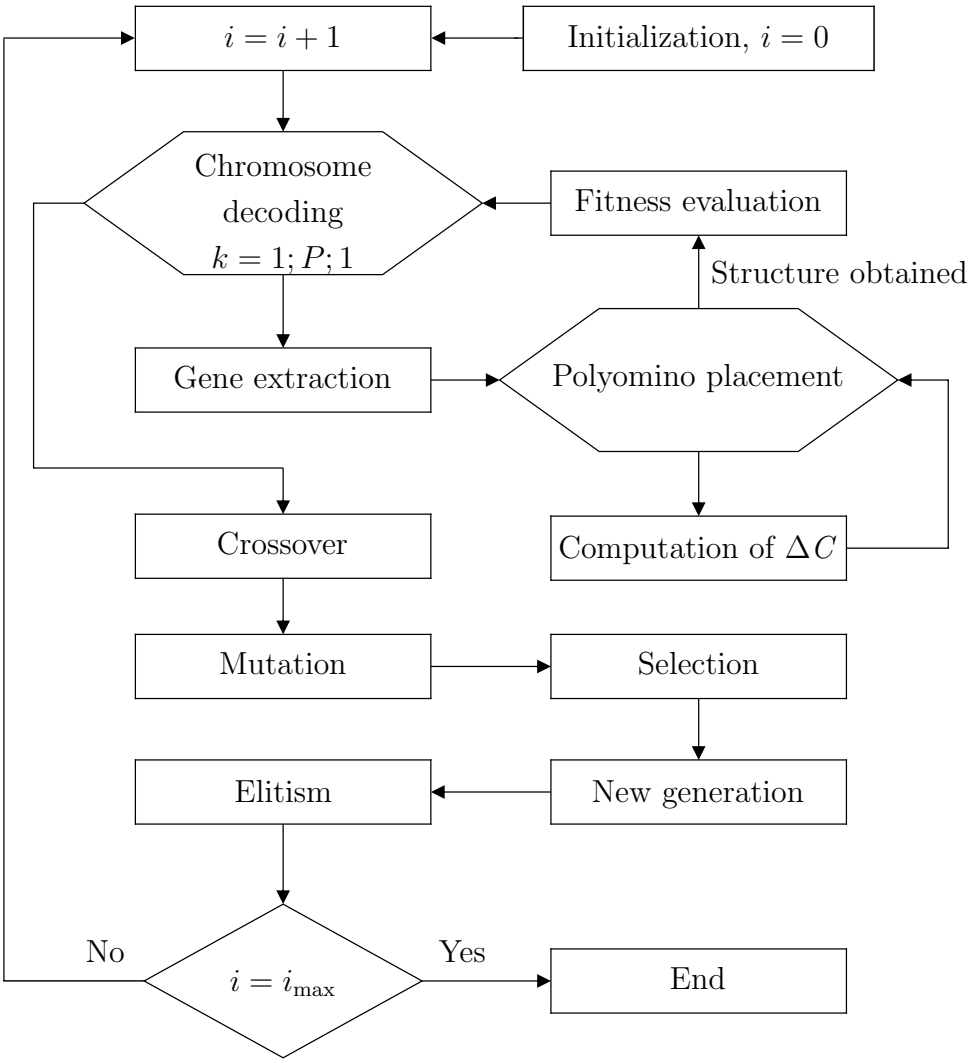


Figure 3.14: Flowchart of the "Snowball" algorithm

each particular polyomino should be extracted. We split the chromosome in blocks of L_g bits and convert them into orientations μ_i. As a result we obtain a vector of orientations, one per polyomino $\mu = \{4, 1, 3, 7, 2, \ldots\}$. As an example, 8 orientations of an L-shaped octomino are shown in figure 3.15.

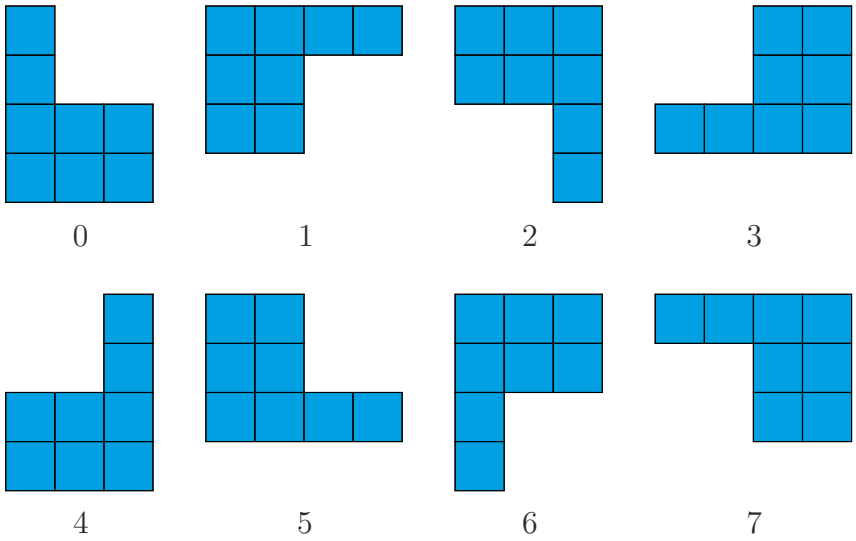


Figure 3.15: Orientations of L-shaped octomino with corresponding numbers

Step 2 — Placing the first polyomino

The main difference of SA is the order of polyomino placement in structure. In order to provide good tiling, polyominoes are placed starting from center towards borders. Such placement method got name centrifugal placement. The first polyomino is put in the center of the area. The coordinates of the barycentre of the first polyomino in case of $M \times N$ array are defined as:

$$X_C = \left\lfloor \frac{M}{2} \right\rfloor, \tag{3.18}$$

$$Y_C = \left\lfloor \frac{N}{2} \right\rfloor. \tag{3.19}$$

Step 3 — Placing other polyominoes

We place other polyominoes close to the first one and choose the position depending on the fitness effect (3.13). Each polyomino is being checked at all positions (to be correct, the barycentre of a polyomino is being put at all the possible positions) in a defined order (figure 3.16). This order is based on the rule that the priority in tiling is given to the central part

of the array. If a position is suitable, the fitness effect is calculated. A suitable position means that the polyomino does not cross the borders of the array and does not overlap any existing polyominoes. The polyomino is put at the position with the largest fitness effect.

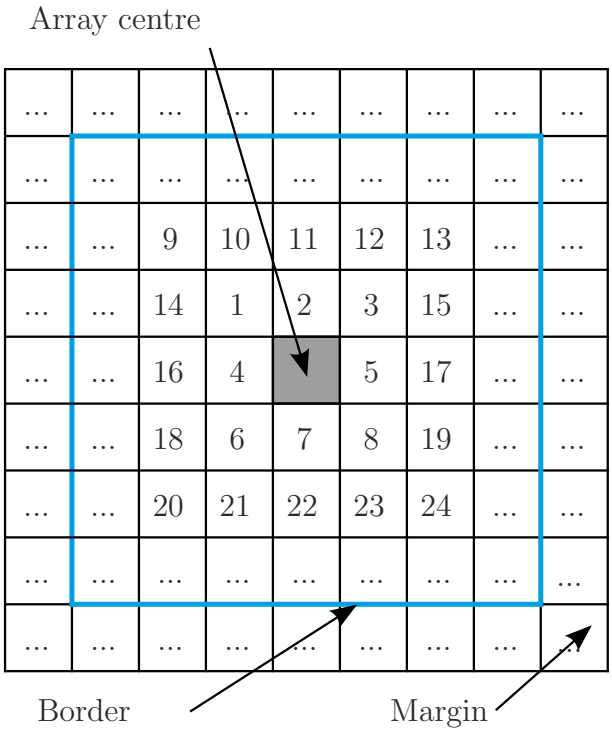


Figure 3.16: Search order of suitable positions

For better tiling of the periphery we introduce margins of $B = 2$ elements around the array (figure 3.16). Margins allow the algorithm to put additional polyominoes into the structure, that will cover those areas which can't accommodate the whole polyomino. New size of the array becomes $M = (M+2B)$, $N = (N+2B)$. After tiling all the elements placed outside the border are eliminated. Eliminating or cutting the elements may not suit all practical areas, but it does not affect badly the performance of the antenna arrays. Our chromosome example (3.17) will result in a structure shown in figure 3.17.

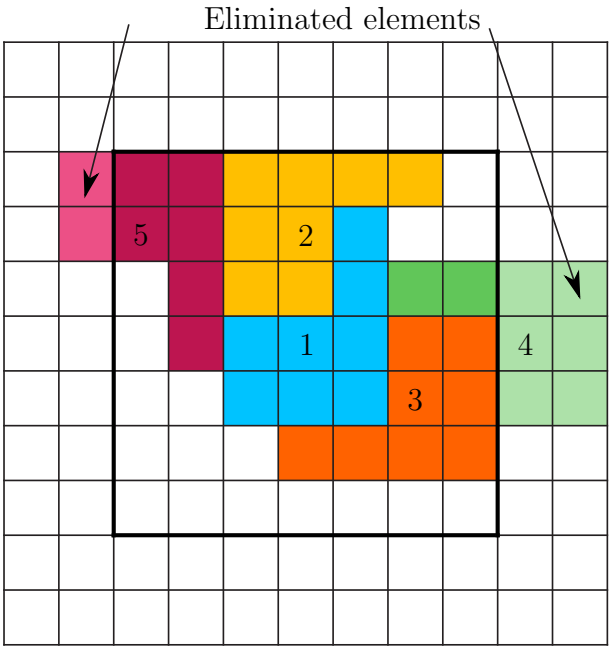


Figure 3.17: Chromosome (3.17) decoding result

3.2.3 Fitness function

For the snowball algorithm we use a similar fitness function to one proposed by Gwee and Lim (3.4). Since the polyominoes can cross the border there is no need to consider the number of boundary edges. Two criteria are left:

- α — number of polyominoes successfully placed in the structure without overlapping;
- ω — number of common edges between all polyominoes.

For solution $\vec{x}$ the fitness function is:

$$C\left(\vec{x}\right) = \psi_\alpha \bar{\alpha}\left(\vec{x}\right) + \psi_\omega \bar{\omega}\left(\vec{x}\right), \tag{3.20}$$

Values of weighting coefficients are chosen proportionally to ones proposed by Gwee and Lim:

$$\begin{aligned} \psi_\alpha &= 0.4, \\ \psi_\omega &= 0.6. \end{aligned} \tag{3.21}$$

3.2.4 Calibration of GA parameters

Likewise the Gwee—Lim algorithm, it is needed to perform a calibration of parameters of the genetic algorithm for the snowball algorithm. That includes probability of crossover p_c, probability of mutation p_m and probability of mutation of a bit p_{bm}. In this paragraph we will describe the process of calibration of these parameters in order to achieve the maximal efficiency of the algorithm [34].

We consider same ranges recommended by Weile and Michielssen [90], as for GLA: $0.6 < p_c < 0.9$, $0.1 < p_m < 0.3$, $0.001 < p_{bm} < 0.1$, and use the same calibration space (figure 3.5). The calibration process runs in two steps at different scales. At first step, in order to walk through all the combinations of parameters, the ranges were discretized with bigger intervals:

$$\begin{aligned} p_c &\in \{0.6,\ 0.7,\ 0.8,\ 0.9\}, \\ p_m &\in \{0.1,\ 0.2,\ 0.3\}, \\ p_{bm} &\in \{0.001,\ 0.01,\ 0.1\}. \end{aligned} \tag{3.22}$$

Performing the tests for all those combinations, we achieve 36 test results (table 3.8). We are dealing with a pseudo random number generator (PRNG) which should be initialized to a starting state using a particular value called seed s. If PRNG is initialized always with the same seed value, it will produce always the same sequence of random values. For this reason, we compute tests with different values of seed for each point of the initial calibration space. We use $N_s = 10$ different values of s: $s \in \{10,\ 20,\ \ldots,\ 100\}$. In total we get N_s solutions for each point i in the space: $\Phi_i^s \in \left\{\Phi_i^{10},\ \Phi_i^{20},\ \ldots,\ \Phi_i^{100}\right\}$. Then, for each point of the initial calibration space we compute the average fitness value (3.8) and standard deviation (3.9):

For more accurate calibration we use a zooming approach. Initial calibration space is divided into 12 cubes. The average value of the fitness

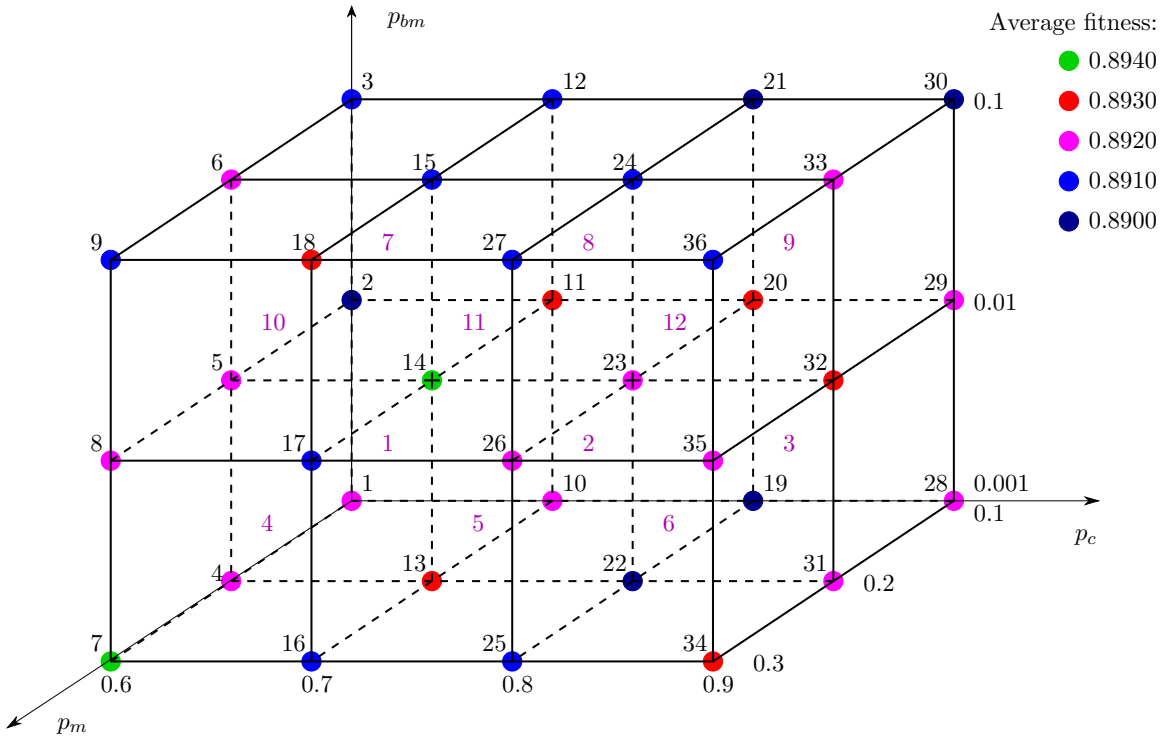


Figure 3.18: The first step of SA calibration

function Φ_{cj}^{avg} is calculated among 8 vertices belonging to the same cube j, where $j = 1, \ldots, 12$. The a cube j_M is selected with the highest average value $\Phi_{cj_M}^{avg} \geqslant \Phi_{cj}^{avg}$ $\forall j$ and considered as a calibration space for the second step.

Let the chosen cube be defined by intervals $p_c \in [a_c; b_c]$, $p_m \in [a_m; b_m]$, $p_{bm} \in [a_{bm}; b_{bm}]$. Parameter ranges for the second step will be expressed as follows:

$$\begin{aligned} p_c &\in \left\{a_c; \tfrac{a_c+b_c}{2}; b_c\right\}, \\ p_m &\in \left\{a_m; \tfrac{a_m+b_m}{2}; b_m\right\}, \\ p_{bm} &\in \left\{a_{bm}; \tfrac{a_{bm}+b_{bm}}{2}; b_{bm}\right\}. \end{aligned} \tag{3.23}$$

Now there are $3 \times 3 \times 3 = 27$ points. As before, the fitness function is calculated at every point with 10 different seeds. The average values are computed and the one with the highest average is chosen. Its coordinates will be the optimal parameters of GA.

Table 3.8: Average fitness function values at the first step of SA calibration

i	Φ_i	σ_i	p_c	p_m	p_{bm}	i	Φ_i	σ_i	p_c	p_m	p_{bm}
1	0.8920	0.0040	0.6	0.1	0.001	19	0.8900	0.0045	0.8	0.1	0.001
2	0.8900	0.0000	0.6	0.1	0.01	20	0.8930	0.0046	0.8	0.1	0.01
3	0.8910	0.0030	0.6	0.1	0.1	21	0.8900	0.0000	0.8	0.1	0.1
4	0.8920	0.0040	0.6	0.2	0.001	22	0.8900	0.0000	0.8	0.2	0.001
5	0.8920	0.0040	0.6	0.2	0.01	23	0.8920	0.0040	0.8	0.2	0.01
6	0.8920	0.0040	0.6	0.2	0.1	24	0.8910	0.0030	0.8	0.2	0.1
7	0.8940	0.0049	0.6	0.3	0.001	25	0.8910	0.0070	0.8	0.3	0.001
8	0.8920	0.0040	0.6	0.3	0.01	26	0.8920	0.0040	0.8	0.3	0.01
9	0.8910	0.0030	0.6	0.3	0.1	27	0.8910	0.0030	0.8	0.3	0.1
10	0.8920	0.0040	0.7	0.1	0.001	28	0.8920	0.0040	0.9	0.1	0.001
11	0.8930	0.0046	0.7	0.1	0.01	29	0.8920	0.0040	0.9	0.1	0.01
12	0.8910	0.0030	0.7	0.1	0.1	30	0.8900	0.0000	0.9	0.1	0.1
13	0.8930	0.0046	0.7	0.2	0.001	31	0.8920	0.0060	0.9	0.2	0.001
14	0.8940	0.0049	0.7	0.2	0.01	32	0.8930	0.0046	0.9	0.2	0.01
15	0.8910	0.0030	0.7	0.2	0.1	33	0.8920	0.0040	0.9	0.2	0.1
16	0.8910	0.0030	0.7	0.3	0.001	34	0.8930	0.0064	0.9	0.3	0.001
17	0.8910	0.0030	0.7	0.3	0.01	35	0.8920	0.0040	0.9	0.3	0.01
18	0.8930	0.0046	0.7	0.3	0.1	36	0.8910	0.0030	0.9	0.3	0.1

All the experiments use the following parameters:

- number of elements along axis x: $M = 64$;
- number of elements along axis y: $N = 64$;
- type of polyomino: L-shaped octomino;
- number of individuals: $P = 10$;
- number of iterations: $K = 100$.

Figure 3.18 shows the values of the fitness function at each point of the initial calibration space. Table 3.9 provides average values of the fitness function for each of the 12 cubes.

Table 3.9: Average fitness function values of cubes at the first step of SA calibration

j	Φ_{cj}^{avg}	$a_c \div b_c$	$a_m \div b_m$	$a_{bm} \div b_{bm}$
1	0.8922	0.6 ÷ 0.7	0.1 ÷ 0.2	0.1 ÷ 0.01
2	0.8921	0.7 ÷ 0.8	0.1 ÷ 0.2	0.1 ÷ 0.01
3	0.8917	0.8 ÷ 0.9	0.1 ÷ 0.2	0.1 ÷ 0.01
4	0.8923	0.6 ÷ 0.7	0.1 ÷ 0.3	0.1 ÷ 0.01
5	0.8917	0.7 ÷ 0.8	0.1 ÷ 0.3	0.1 ÷ 0.01
6	0.8919	0.8 ÷ 0.9	0.1 ÷ 0.3	0.1 ÷ 0.01
7	0.8917	0.6 ÷ 0.7	0.1 ÷ 0.2	0.1 ÷ 0.1
8	0.8919	0.7 ÷ 0.8	0.1 ÷ 0.2	0.1 ÷ 0.1
9	0.8916	0.8 ÷ 0.9	0.1 ÷ 0.2	0.1 ÷ 0.1
10	0.8920	0.6 ÷ 0.7	0.1 ÷ 0.3	0.1 ÷ 0.1
11	0.8919	0.7 ÷ 0.8	0.1 ÷ 0.3	0.1 ÷ 0.1
12	0.8918	0.8 ÷ 0.9	0.1 ÷ 0.3	0.1 ÷ 0.1

The average fitness values of each cube are slightly different from each other. Anyway, we can see that cube number 4 has the best average fitness function value: $\Phi_{c4}^{avg} = 0.8923$. We compute more detailed tests for the chosen cube number 4, following the rules described above. For each point of the new calibration space, we compute the fitness value with different seed values, the average fitness value and the standard deviation value (table 3.10). Figure 3.19 shows average values for all points of the second calibration space.

Point number 11 has the highest average value of the fitness function $\Phi_{11}^{avg} = 0.8950$. It corresponds to the set of parameters $p_c = 0.65$, $p_m = 0.2$, $p_{bm} = 0.005$. Those are the calibrated GA parameters for the snowball algorithm. Therefore, the optimal GA parameters such as crossover probability, mutation probability and bit mutation probability have been obtained for the snowball algorithm. Those values will be used in further experiments. The accuracy of values is enough for this kind of tasks, because the fitness function will contain a criterion, nonlinearly connected with the structure.

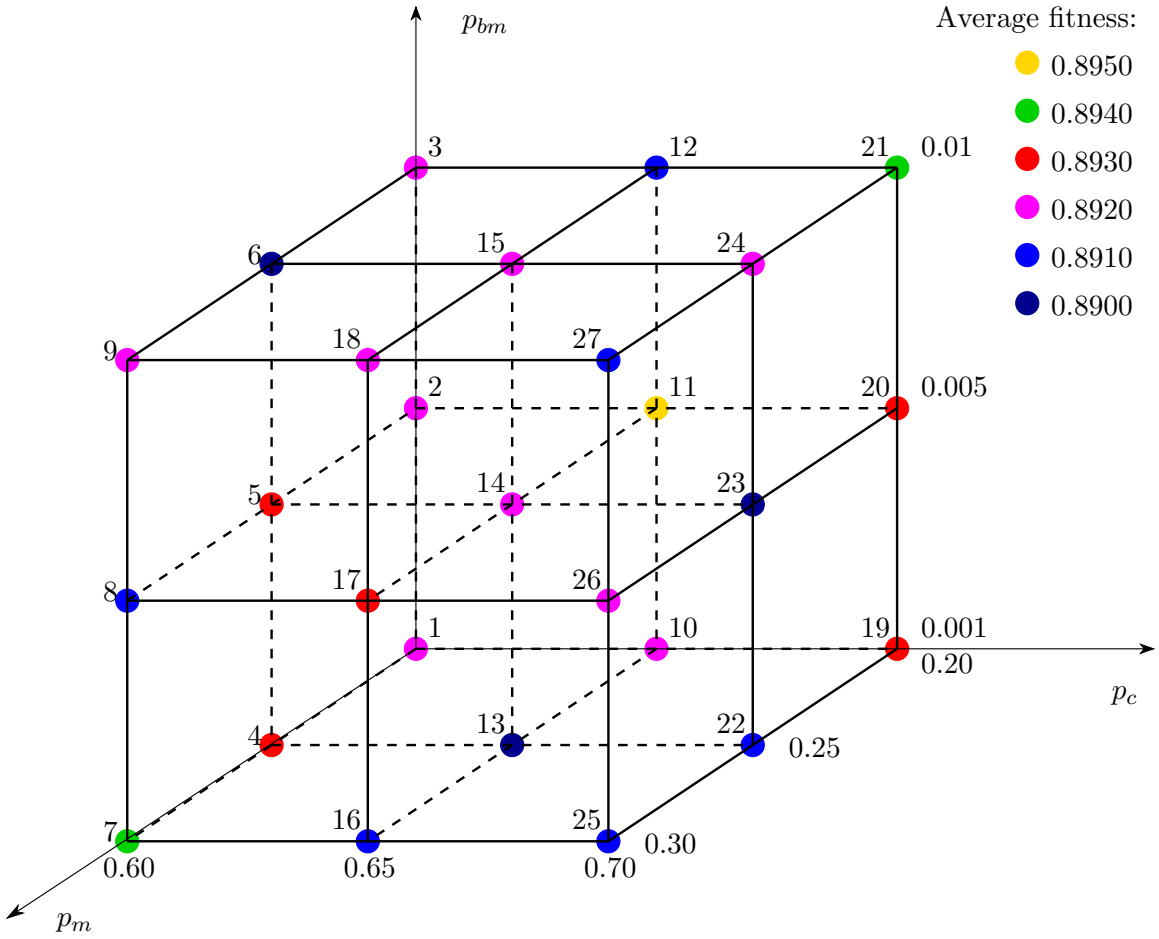


Figure 3.19: Second step of the SA calibration

Its existence will decrease current values of the weighting coefficients by such criteria, as number of successfully placed polyominoes in a structure and number of common edges.

3.2.5 Examples of structures by the "Snowball" algorithm

In this section we provide four experiments as examples of work of the snowball algorithm. The experiments are done with different structure sizes and various polyomino shapes. We used the calibrated values of p_c, p_m and p_{bm}. Figure 3.20 shows the graph of a fitness function and its components in the fourth example.

Table 3.10: Average values of the fitness function at the second step of calibration of SA

i	Φ_i	σ_i	p_c	p_m	p_{bm}
1	0.892	0.001	0.60	0.20	0.001
2	0.892	0.001	0.60	0.20	0.005
3	0.892	0.001	0.60	0.20	0.01
4	0.893	0.006	0.60	0.25	0.001
5	0.893	0.005	0.60	0.25	0.005
6	0.890	0.005	0.60	0.25	0.01
7	0.894	0.005	0.60	0.30	0.001
8	0.891	0.003	0.60	0.30	0.005
9	0.892	0.004	0.60	0.30	0.01
10	0.892	0.004	0.65	0.20	0.001
11	0.895	0.005	0.65	0.20	0.005
12	0.891	0.003	0.65	0.20	0.01
13	0.890	0.005	0.65	0.25	0.001
14	0.892	0.004	0.65	0.25	0.005

i	Φ_i	σ_i	p_c	p_m	p_{bm}
15	0.892	0.004	0.65	0.25	0.01
16	0.891	0.005	0.65	0.30	0.001
17	0.893	0.005	0.65	0.30	0.005
18	0.892	0.004	0.65	0.30	0.01
19	0.893	0.005	0.70	0.20	0.001
20	0.893	0.005	0.70	0.20	0.005
21	0.894	0.005	0.70	0.20	0.01
22	0.891	0.003	0.70	0.25	0.001
23	0.890	0.001	0.70	0.25	0.005
24	0.892	0.004	0.70	0.25	0.01
25	0.891	0.003	0.70	0.30	0.001
26	0.892	0.004	0.70	0.30	0.005
27	0.891	0.004	0.70	0.30	0.01

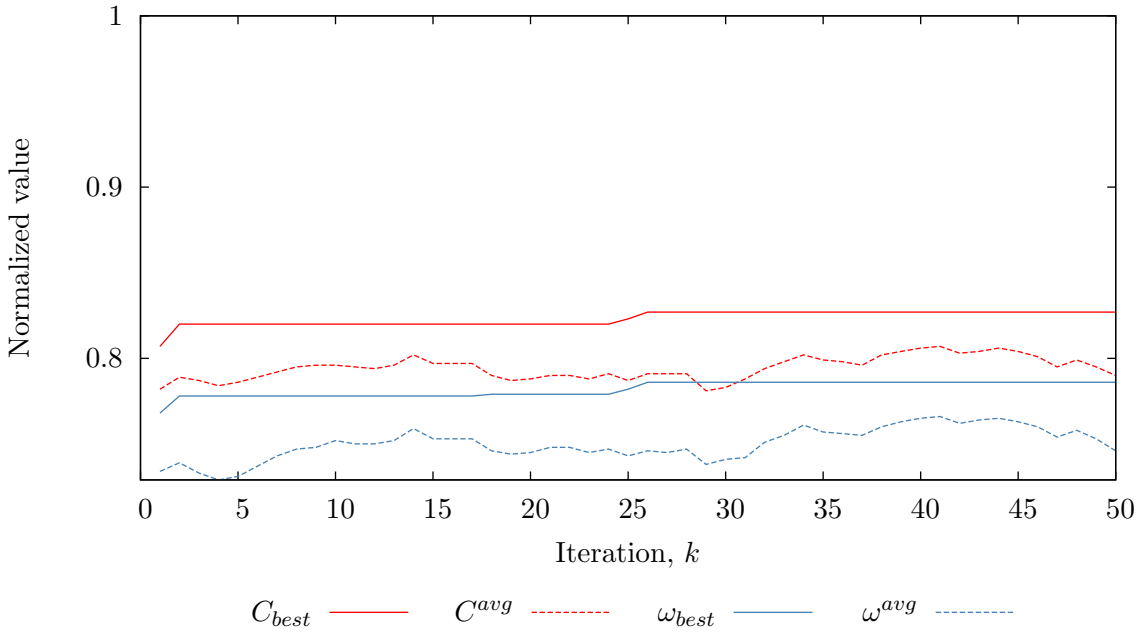


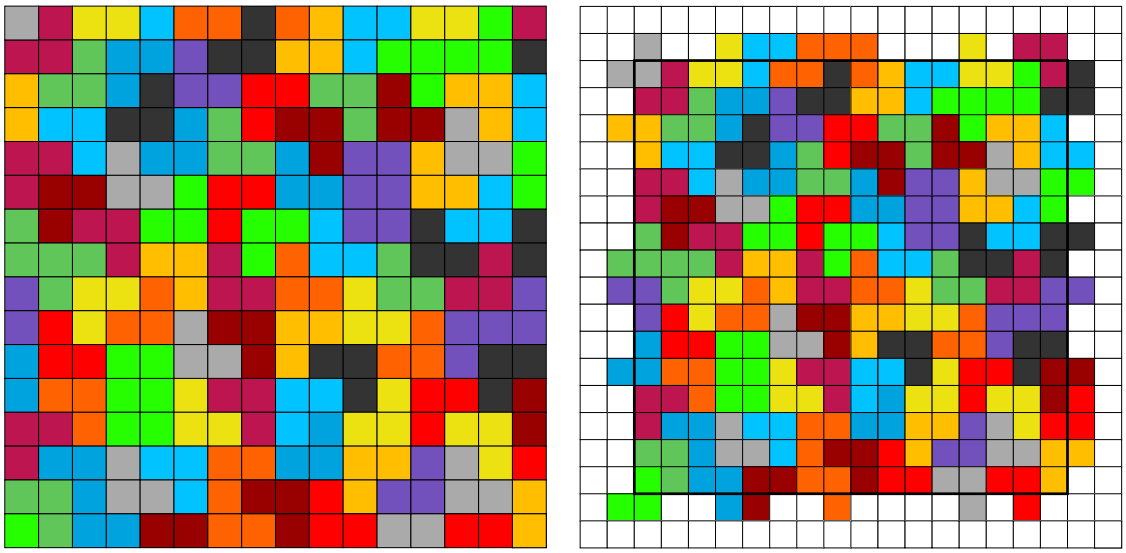
Figure 3.20: Fitness function in the fourth example

Example 1: Structure 16×16, L-tromino

Input parameters:

- structure size: $M = N = 16$;
- polyomino type: L-tromino.

Figure 3.21 shows the structure obtained by GLA with those parameters. Numerical results are provided in table 3.11.



(a) (b)

Figure 3.21: Structure of the array in the first example without (a) and with margins (b)

Table 3.11: Output data of the first example

Parameter	Value
Number of polyominoes α	96
Number of common edges ω	320
Number of holes H	0
Fullness of the structure A, %	100

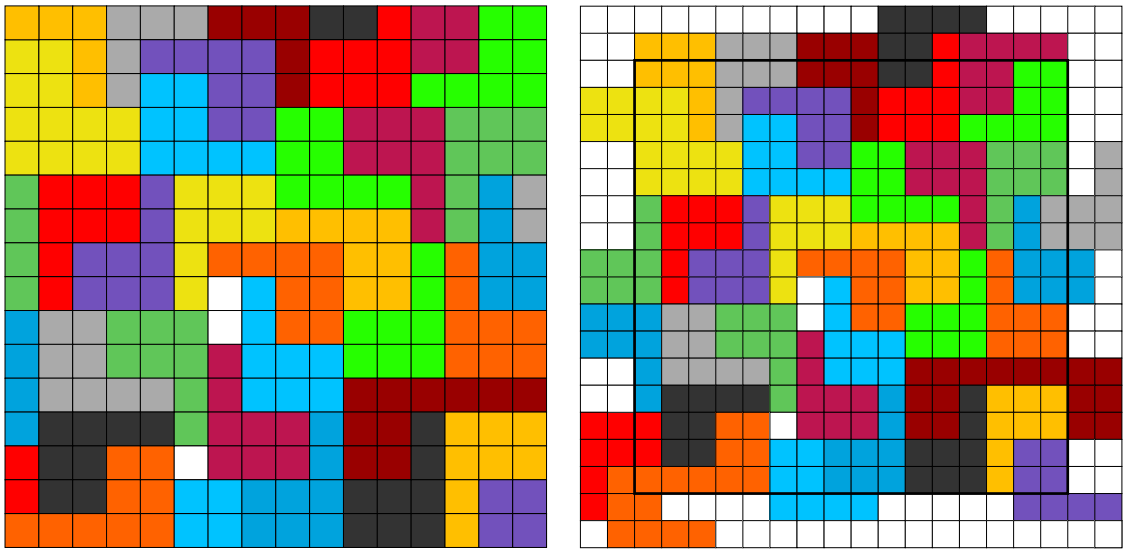
In this example the algorithm was able to fill the structure completely.

Example 2: Structure 16×16, L-octomino

Input parameters:

- structure size: $M = N = 16$;
- polyomino type: L-octomino.

Figure 3.22 shows the structure obtained by GLA with those parameters. Numerical results are provided in table 3.12.



(a) (b)

Figure 3.22: Structure of the array in the second example without (a) and with margins (b)

Table 3.12: Output data of the second example

Parameter	Value
Number of polyominoes α	40
Number of common edges ω	202
Number of holes H	3
Fullness of the structure A, %	98.83

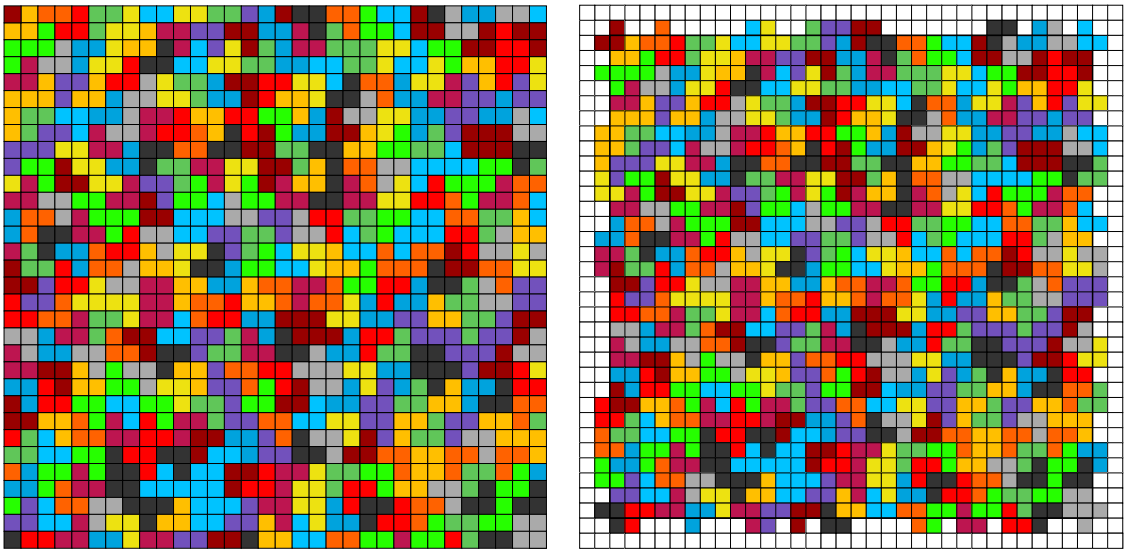
This example shows only three holes that is 13 times less then in a similar example by GLA.

Example 3: Structure 32×32, L-tromino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-tromino.

Figure 3.23 shows the structure obtained by GLA with those parameters. Numerical results are provided in table 3.13.



(a) (b)

Figure 3.23: Structure of the array in the third example without (a) and with margins (b)

Table 3.13: Output data of the third example

Parameter	Value
Number of polyominoes α	363
Number of common edges ω	1323
Number of holes H	0
Fullness of the structure A, %	100

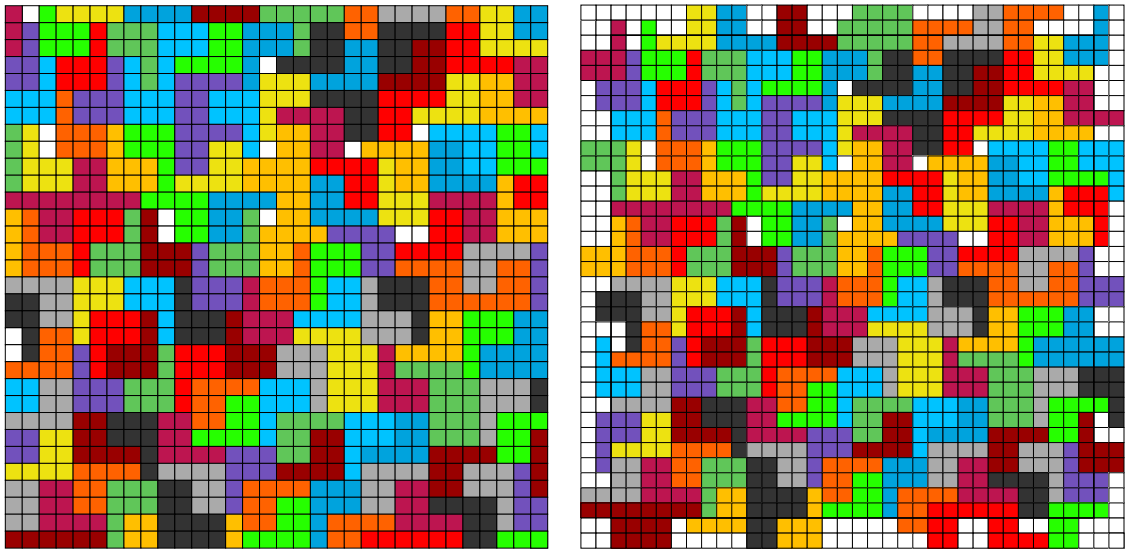
In this example too use of L-trominoes avoided any uncovered cells.

Example 4: Structure 32×32, L-octomino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-octomino.

Figure 3.24 shows the structure obtained by GLA with those parameters. Numerical results are provided in table 3.14.



(a) (b)

Figure 3.24: Structure of the array in the fourth example without (a) and with margins (b)

Table 3.14: Output data of the fourth example

Parameter	Value
Number of polyominoes α	144
Number of common edges ω	835
Number of holes H	14
Fullness of the structure A, %	98.63

In the fourth there are 14 holes, that is less than 2% of the structure.

Therefore, the "Snowball" algorithm was developed, that differs by the way of polyomino placement that provides better structure filling in comparison with the Gwee—Lim algorithm. Parameters of the algorithm have been calibrated. Examples have been obtained and shown. Analysis of the examples shows that structure fullness grew up to 98–100%.

3.3 Comparison of the Gwee—Lim and "Snowball" algorithms by the fullness of structures

In this paragraph we show the comparison of the functioning of the Gwee—Lim and "Snowball" algorithms. The following parameters were used:

- structure size: $M = N = \{20,\ 25,\ 30,\ 35,\ 40,\ 45,\ 50,\ 55,\ 60\}$;
- number of iterations: $K = 50$;
- population size: $P = 10$;
- elitism: on;
- seed: $s = \{10,\ 20,\ 30,\ 40,\ 50,\ 60,\ 70\}$.

Obtained fullness values for each structure size was averaged over seven different seeds of PRNG. Each algorithm used its calibrated parameters.

Figure 3.25 shows the graph of the average fullness of structures by L-trominoes. Numerical data is provided in table 3.15. It can be seen that unlike GLA, SA demonstrates constantly high fullness of structures regardless its size.

Figure 3.26 shows the graph of the average fullness of structures by L-octominoes. Numerical data is provided in table 3.16. Due to big size of L-octominoes the fullness of both algorithms decreased, but the difference between GLA and SA only increased, which says about advantage of the SA in structure tiling.

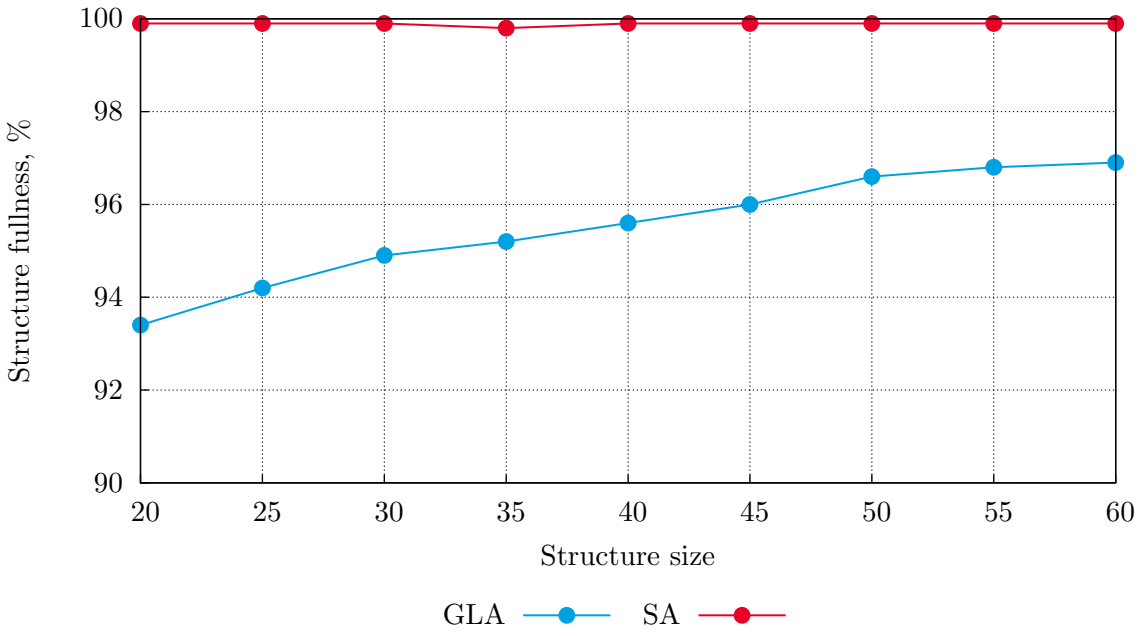


Figure 3.25: Fullness of structures of different sizes by L-trominoes

As the final comparison of GLA and SA we list several test cases on tiling a square structure of 32×32 elements with polyominoes of different types, shown in figure 1.9. Figure 3.27 presents the graph of average fullness of structures with different polyominoes. Numerical data is provided in table 3.17.

As it is seen from the graph, the Gwee—Lim algorithm tends to decrease the fullness with the growth of the polyomino size. Again, SA shows better results. there is an evident problem with tiling by C-shaped octominoes. Figures 3.28 and 3.29 show examples of such structures with C-octominoes. It is visible, that the "C" shape does not let polyominoes to lie close to each other and leads to many holes. In the upper part of figure 3.29 there is a tiled area of high density, but it is characterized by repeated patterns of polyominoes' relevant placement. In the fourth chapter it will be shown, that such patterns have negative effect on the radiation pattern leading to grating lobes. Therefore, using C-octominoes does not seem prospective

Table 3.15: Fullness of structures of different sizes by L-trominoes

Structure size $M = N$	Fullness by GLA, %	Fullness by SA, %
20	93.4	99.9
25	94.2	99.9
30	94.9	99.9
35	95.2	99.8
40	95.6	99.9
45	96.0	99.9
50	96.6	99.9
55	96.8	99.9
60	96.9	99.9

Table 3.16: Fullness of structures of different sizes by L-octominoes

Structure size $M = N$	Fullness by GLA, %	Fullness by SA, %
20	86.2	98.6
25	86.1	98.1
30	87.1	98.0
35	87.2	98.2
40	87.7	98.0
45	87.7	97.9
50	88.2	97.8
55	88.4	97.9
60	88.7	97.9

and will not be considered further.

Table 3.17: Fullness of a 32×32 structure with different polyominoes

Polyomino type	Fullness by GLA, %	Fullness by SA, %
L-tromino	95.0	99.9
L-tetromino	90.6	99.5
S-tetromino	89.8	99.4
T-tetromino	89.7	97.7
L-octomino	86.9	98.0
C-octomino	70.3	84.9
Pu-octomino	84.1	96.6

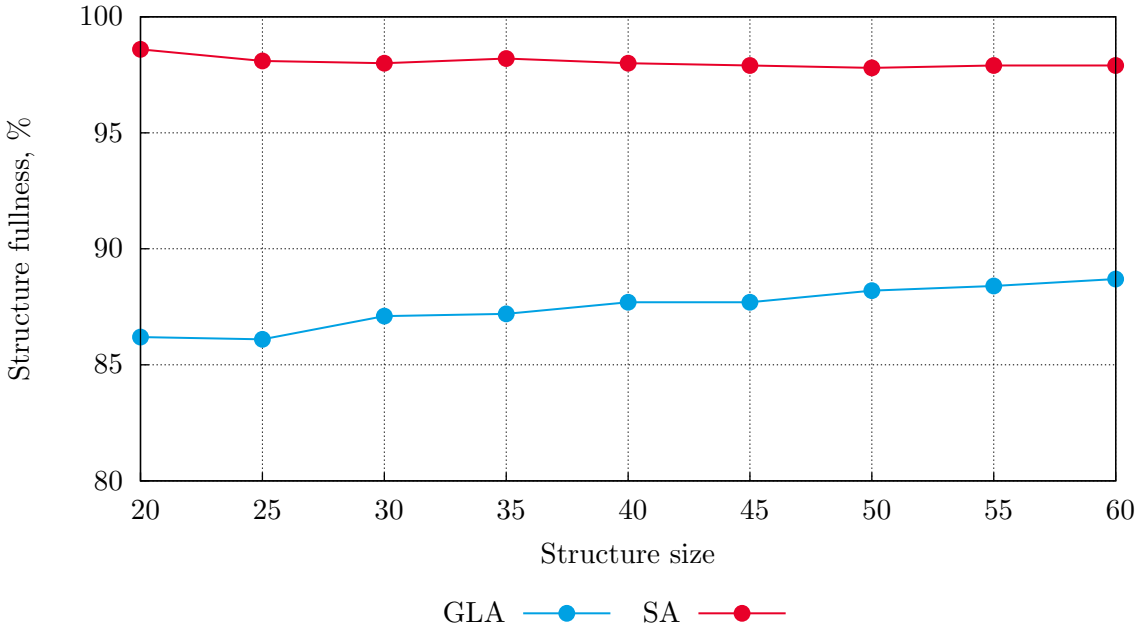


Figure 3.26: Fullness of structures of different sizes by L-octominoes

Thus, the comparative analysis of functioning of two algorithms has been performed from the point of view of fullness of the obtained structures. The comparison has shown that the snowball algorithm tiles the structures in average 10% better, than the Gwee—Lim algorithm.

3.4 Tiling with two shapes of polyominoes simultaneously

In order to increase the fullness of structures, tiled with large polyominoes, in this work we propose to tile structures with two shapes of polyominoes simultaneously. The first shape appears to be a large polyomino, mainly octomino. The second shape is a smaller polyomino, for example, tromino or tetromino. The use of two shapes in the process of filling the geometric structure leads to geometric symbiosis of polyominoes, when small polyominoes fill hard to reach areas and large ones at the same time get more

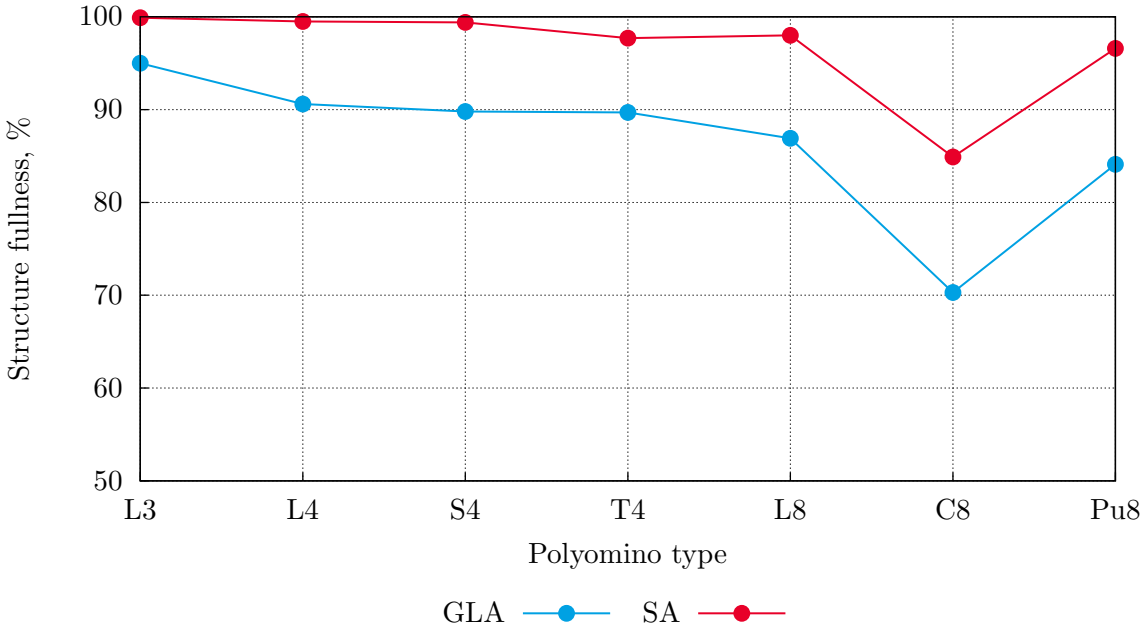


Figure 3.27: Fullness of a 32×32 structure with different polyominoes

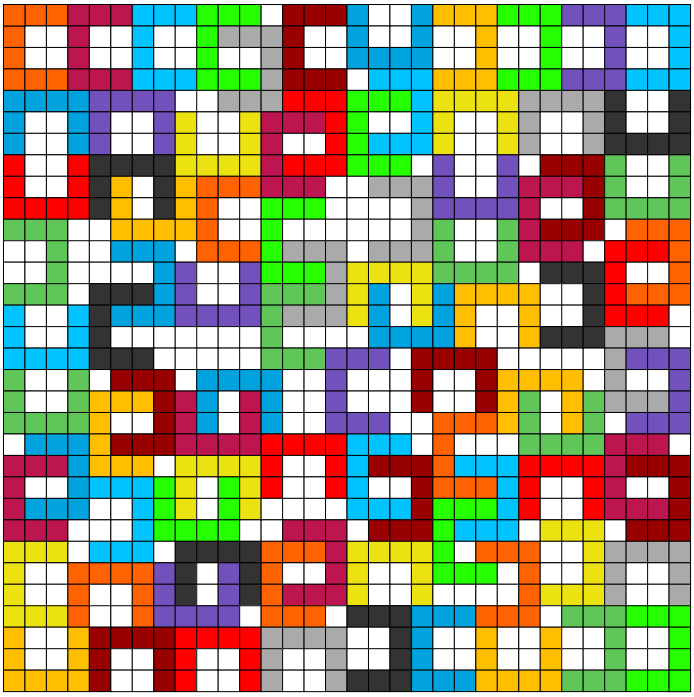


Figure 3.28: A 32×32 structure, tiled by GLA with C-octominoes

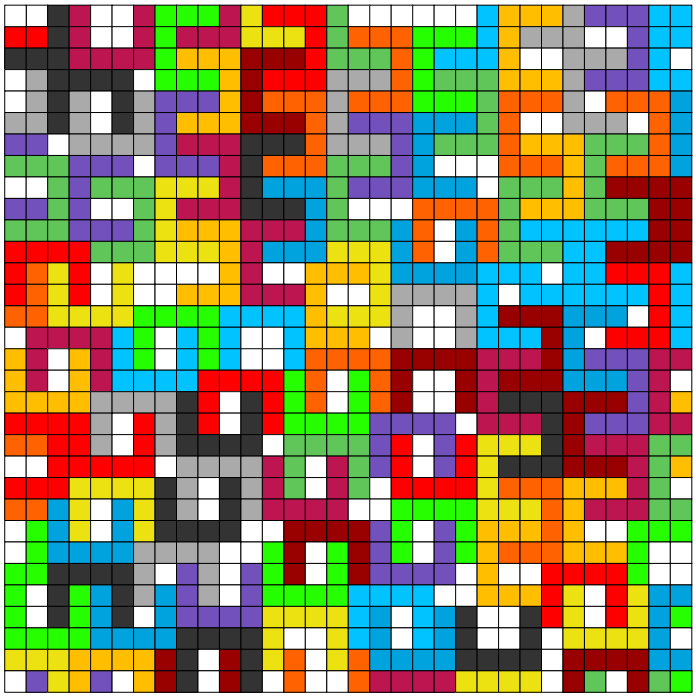


Figure 3.29: A 32×32 structure, tiled by SA with C-octominoes

options for placement.

The selection of one of two shapes is done by the genetic algorithm. For this reason the length of genes was increased from 3 to 4 bits, where the fourth bit defines the shape of polyomino (figure 3.30).

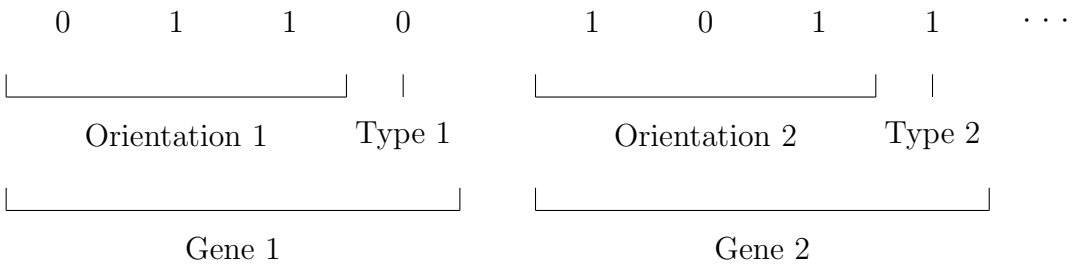


Figure 3.30: Extended genes in a chromosome

Further, examples of structures obtained by two algorithms with two shapes of polyominoes are listed. At the end, the analysis of results is done and the optimal pair is being chosen.

3.4.1 Examples of structures by GLA with two shapes of polyominoes

In this paragraph we present four experiments as examples of structures by the Gwee—Lim algorithm with two shapes of polyominoes. Experiments were held with different structure sizes and different pairs of polyominoes.

We used calibrated values of p_c, p_m and p_{bm}. Figure 3.31 shows fitness function and its components for the second example.

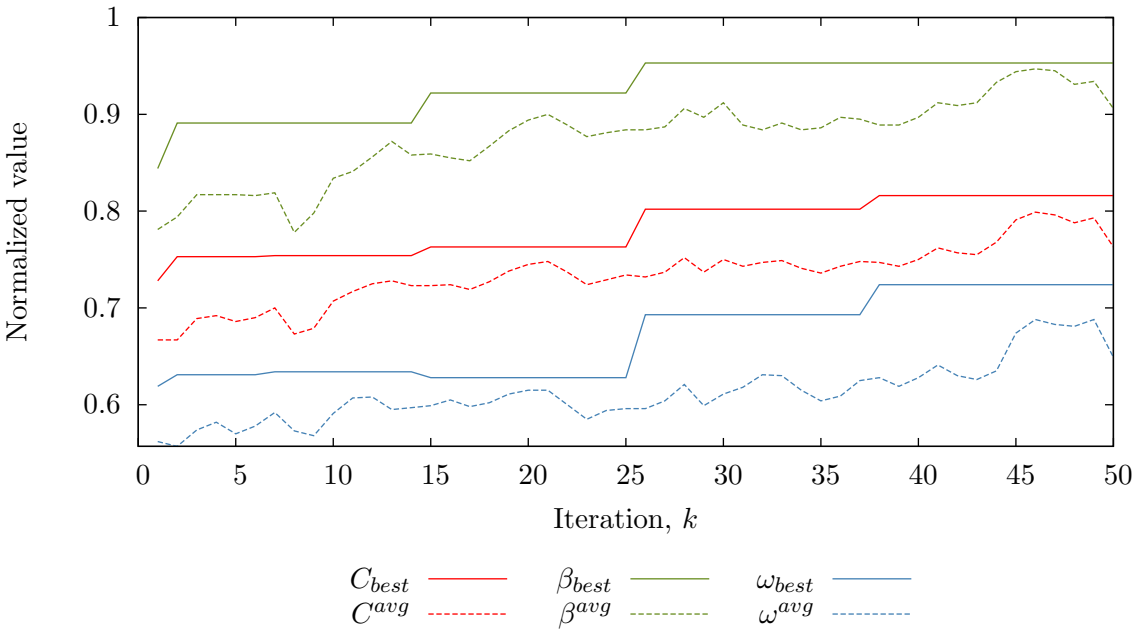


Figure 3.31: Fitness function in the second example

Example 1: Structure 16 × 16, L-tromino and L-octomino

Input parameters:

- structure size: $M = N = 16$;
- polyomino type: L-tromino and L-octomino.

Figure 3.32 shows the structure obtained by the GLA with these parameters. Numerical results are provided in table 3.18.

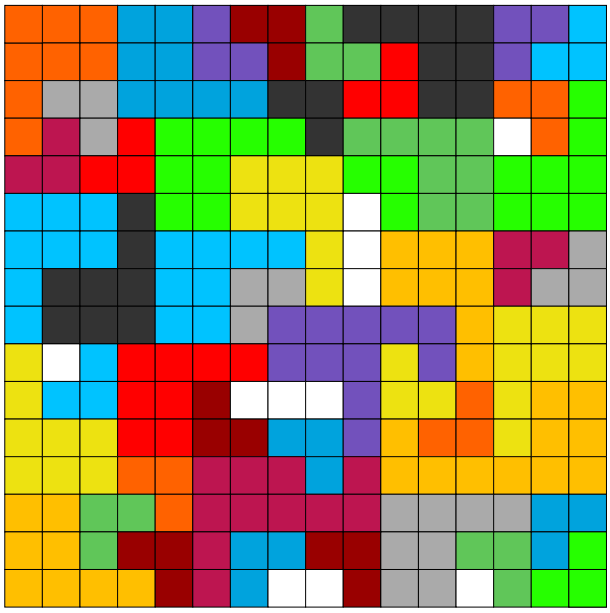


Figure 3.32: Array structure in the first example

Table 3.18: Output data of the first example

Parameter	Value
Number of polyominoes α	50
Number of boundary edges β	61
Number of common edges ω	211
Number of holes H	11
Fullness of structure A, %	95.70

Structure fullness has grown in comparison with structure of only L-octominoes.

Example 2: Structure 16 × 16, L-tetromino and L-octomino

Input parameters:

- structure size: $M = N = 16$;
- polyomino type: L-tetromino and L-octomino.

Figure 3.33 shows the structure obtained by the GLA with these parameters. Numerical results are provided in table 3.19.

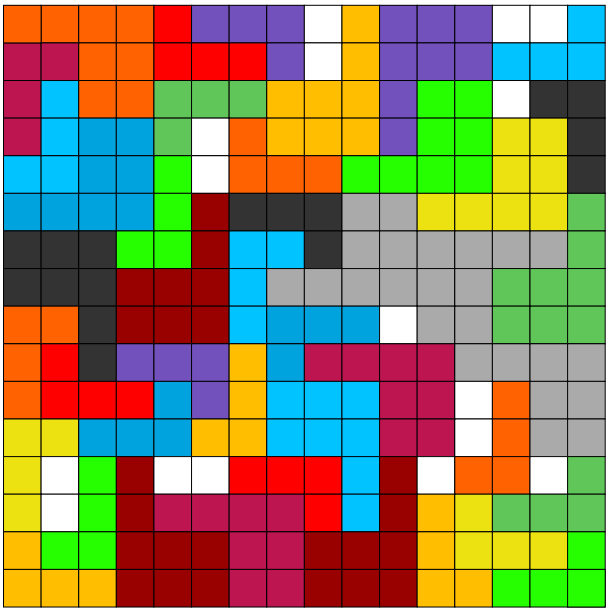


Figure 3.33: Array structure in the second example

Table 3.19: Output data of the second example

Parameter	Value
Number of polyominoes α	43
Number of boundary edges β	61
Number of common edges ω	194
Number of holes H	16
Fullness of structure A, %	93.75

The fullness decreased, because tetrominoes are larger than trominoes.

Example 3: Structure 32 × 32, L-tromino and L-octomino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-tromino and L-octomino.

Figure 3.34 shows the structure obtained by the GLA with these parameters. Numerical results are provided in table 3.20.

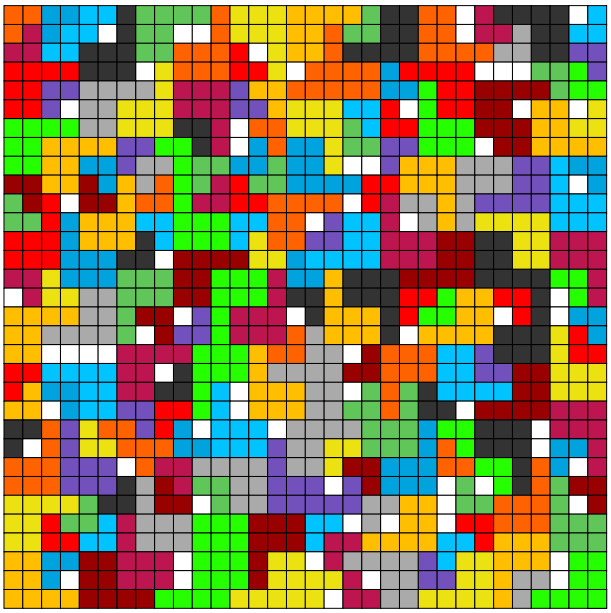


Figure 3.34: Array structure in the third example

Table 3.20: Output data of the third example

Parameter	Value
Number of polyominoes α	183
Number of boundary edges β	124
Number of common edges ω	791
Number of holes H	70
Fullness of structure A, %	93.16

The decrease in fullness of larger structures indicates that GLA is not efficient in working with two shapes.

Example 4: Structure 32 × 32, L-tetromino and L-octomino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-tetromino and L-octomino.

Figure 3.35 shows the structure obtained by the GLA with these parameters. Numerical results are provided in table 3.21.

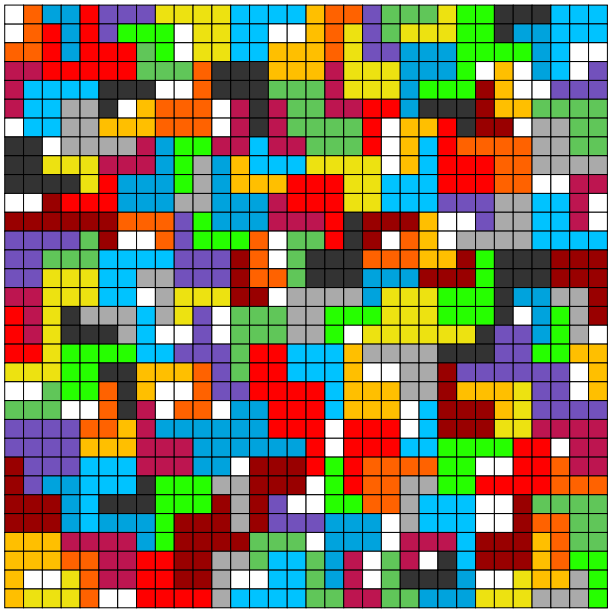


Figure 3.35: Array structure in the fourth example

Table 3.21: Output data of the fourth example

Parameter	Value
Number of polyominoes α	167
Number of boundary edges β	116
Number of common edges ω	782
Number of holes H	88
Fullness of structure A, %	91.41

In case of tetrominoes we also see a decrease in fullness of a larger structure.

3.4.2 Examples of structures by SA with two shapes of polyominoes

In this paragraph we present four experiments as examples of structures by the snowball algorithm with two shapes of polyominoes. Experiments were held with different structure sizes and different pairs of polyominoes. The following parameters were used:

- number of iterations: $K = 50$;
- population size: $P = 10$;

We used calibrated values of p_c, p_m and p_{bm} with seed $s = 37$. Figure 3.36 shows fitness function and its components for the fourth example. It is seen that the fitness function possesses lower values. That is because the maximal number of common edges in a structure is normalized by the biggest such number, that is achievable by only the smaller polyomino.

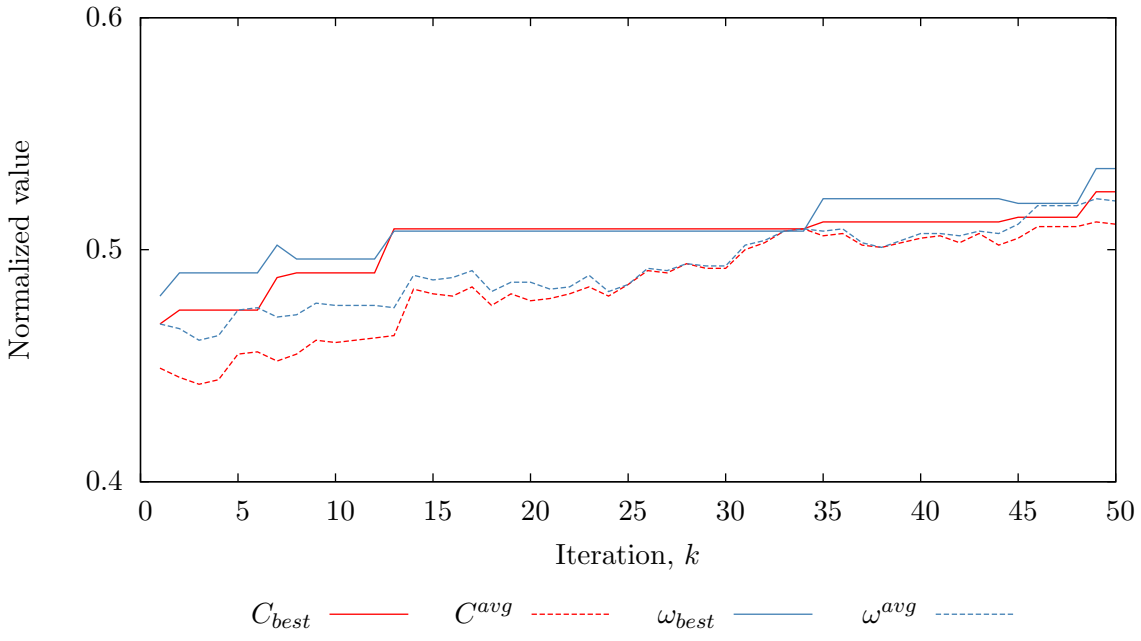


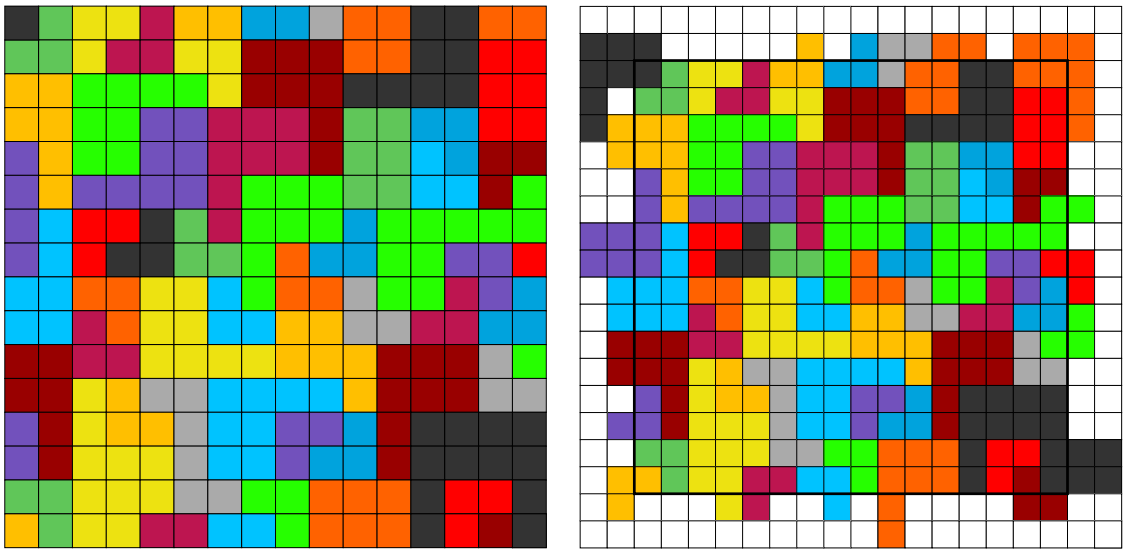
Figure 3.36: Fitness function in the first example

Example 1: Structure 16 × 16, L-tromino and L-octomino

Input parameters:

- structure size: $M = N = 16$;
- polyomino type: L-tromino and L-octomino.

Figure 3.37 shows the structure obtained by the SA with these parameters. Numerical results are provided in table 3.22.



(a) (b)

Figure 3.37: Array structure in the first example without (a) and with (b) margins

Table 3.22: Output data of the first example

Parameter	Value
Number of polyominoes α	68
Number of common edges ω	263
Number of holes H	0
Fullness of structure A, %	100

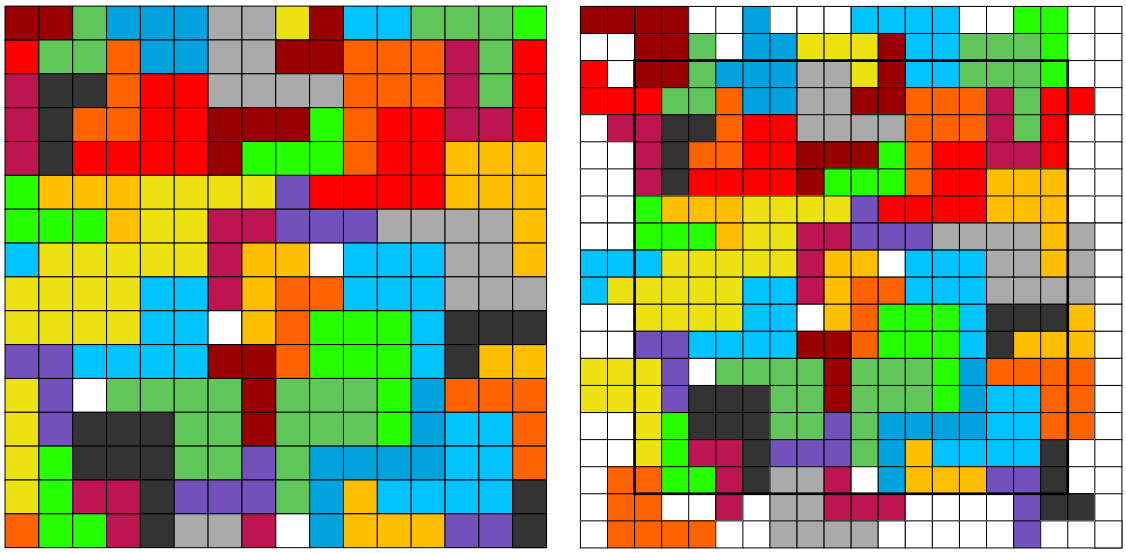
Using two shapes, SA was able to fill the structure completely.

Example 2: Structure 16 × 16, L-tetromino and L-octomino

Input parameters:

- structure size: $M = N = 16$;
- polyomino type: L-tetromino and L-octomino.

Figure 3.38 shows the structure obtained by the SA with these parameters. Numerical results are provided in table 3.23.



(a) (b)

Figure 3.38: Array structure in the second example without (a) and with (b) margins

Table 3.23: Output data of the second example

Parameter	Value
Number of polyominoes α	58
Number of common edges ω	241
Number of holes H	4
Fullness of structure A, %	98.44

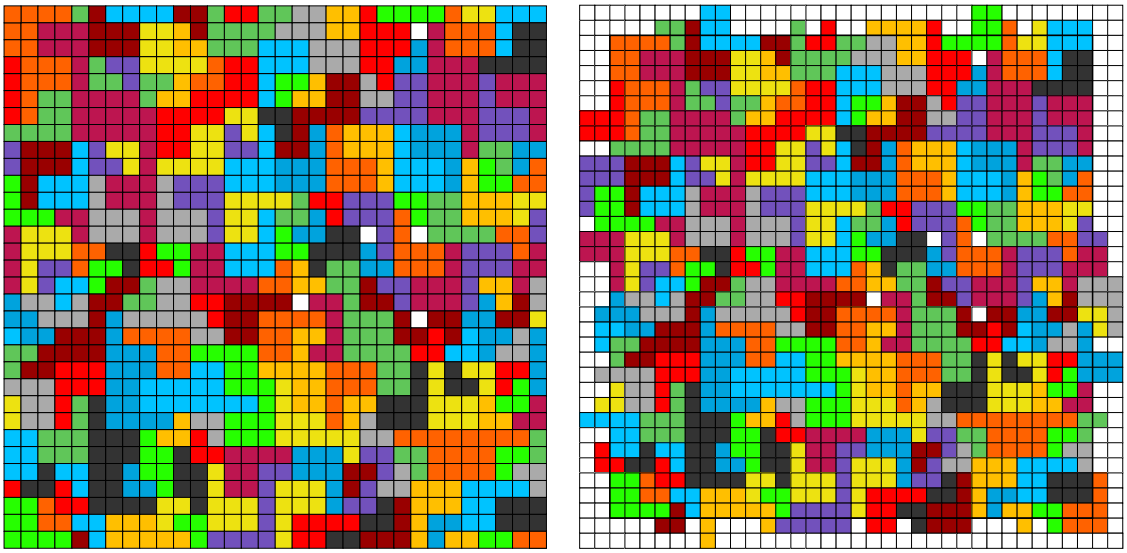
In this example four holes appeared in the structure due to tetrominoes.

Example 3: Structure 32 × 32, L-tromino and L-octomino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-tromino and L-octomino.

Figure 3.39 shows the structure obtained by the SA with these parameters. Numerical results are provided in table 3.24.



(a) (b)

Figure 3.39: Array structure in the third example without (a) and with (b) margins

Table 3.24: Output data of the third example

Parameter	Value
Number of polyominoes α	216
Number of common edges ω	1003
Number of holes H	5
Fullness of structure A, %	99.51

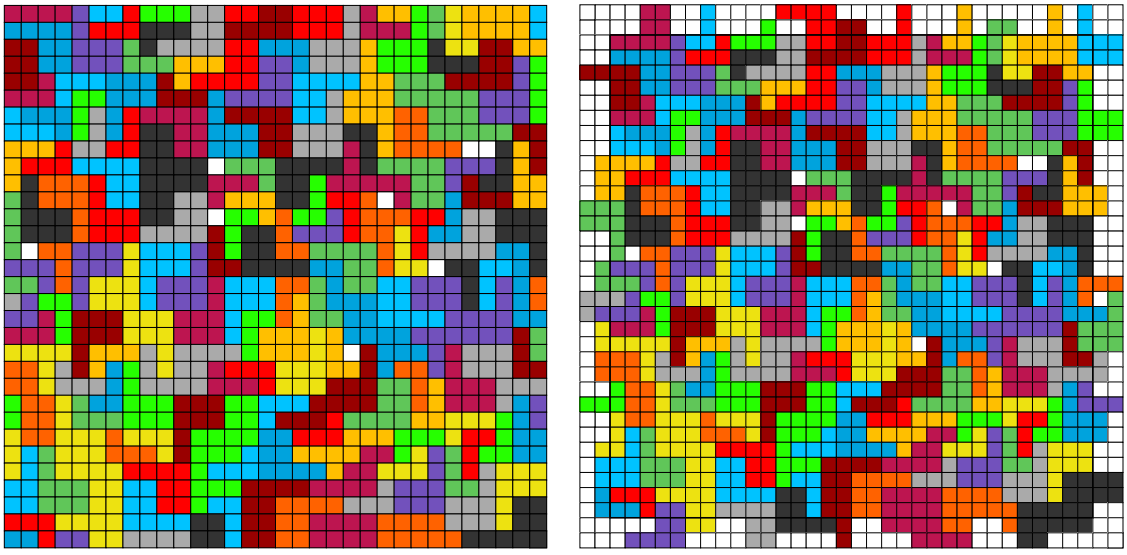
This example shows that as the structure size grows, the probability of holes increases.

Example 4: Structure 32 × 32, L-tetromino and L-octomino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-tetromino and L-octomino.

Figure 3.40 shows the structure obtained by the SA with these parameters. Numerical results are provided in table 3.25.



(a) (b)

Figure 3.40: Array structure in the fourth example without (a) and with (b) margins

Table 3.25: Output data of the fourth example

Parameter	Value
Number of polyominoes α	194
Number of common edges ω	986
Number of holes H	8
Fullness of structure A, %	99.22

In the fourth example with tetromino the number of holes increased.

Thus, we have described tiling of structures with two shapes of polyominoes simultaneously and corresponding adjustments in the algorithms.

Examples of structures by both algorithms were shown. Use of two shapes increased fullness of structures for both algorithms.

3.5 Comparison of the Gwee—Lim and "Snowball" algorithms by the fullness of structures with two shapes of polyominoes

In this paragraph we show the comparison of the functioning of the Gwee—Lim and "Snowball" algorithms with two shapes of polyominoes. The following parameters were used:

- structure size: $M = N = \{20,\ 25,\ 30,\ 35,\ 40,\ 45,\ 50,\ 55,\ 60\}$;
- number of iterations: $K = 50$;
- population size: $P = 10$;
- elitism: on;
- seed: $s = \{10,\ 20,\ 30,\ 40,\ 50,\ 60,\ 70\}$.

Obtained fullness values for each structure size was averaged over seven different seeds of PRNG. Each algorithm used its calibrated parameters.

Figure 3.41 shows the graph of the average fullness of structures by L-trominoes and L-octominoes. For comparison the graph also contains results obtained by tiling with one shape of polyomino (figures 3.25 and 3.26). We can see, that structures tiled with L-trominoes and L-octominoes at the same time take intermediate position at fullness between the structures, tiled with each of those polyominoes separately. And SA demonstrates less dispersion than GLA. Numerical data is provided in table 3.26.

Figure 3.42 shows the graph of the average fullness of structures by L-octominoes and L-tetrominoes. Numerical data is provided in table 3.26. Fullness of structures by SA is higher than those by GLA by 7–9%.

As the final analysis of polyomino pairs, we list several test cases on tiling a square structure of 32×32 elements with different pairs of polyominoes. Figure 3.43 presents the graph of average fullness of structures with different pairs of polyominoes. Numerical data is provided in table 3.27.

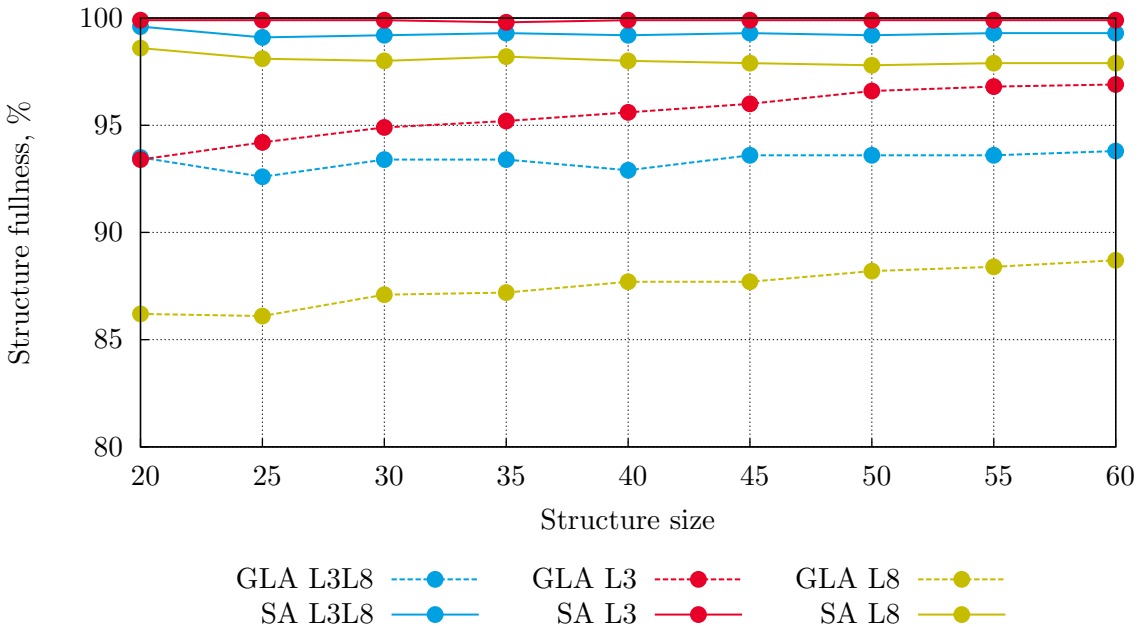


Figure 3.41: Fullness of structures of different sizes by L-trominoes and L-octominoes

The graph shows that both algorithms demonstrate the best tiling by using L-tromino and L-octomino in pair. For this reason, further in the work only this pair will be considered while synthesizing antenna arrays with two shapes of subarrays.

Therefore, the comparative analysis of results by Gwee—Lim and snowball algorithms was performed from the point of view of structure fullness by tiling with two shapes of polyominoes simultaneously. The comparison showed that the snowball algorithm fills structures by 6–10% better than Gwee—Lim algorithm.

Table 3.26: Fullness of structures of different sizes with different pairs of polyominoes

Structure size $M = N$	L-tromino and L-octomino		L-tetromino and L-octomino	
	Fullness by GLA, %	Fullness by SA, %	Fullness by GLA, %	Fullness by SA, %
20	93.5	99.6	90.2	99.3
25	92.6	99.1	90.6	99.1
30	93.4	99.2	90.1	98.8
35	93.4	99.3	89.9	98.7
40	92.9	99.2	90.6	99.0
45	93.6	99.3	90.9	99.0
50	93.6	99.2	90.8	98.8
55	93.6	99.3	91.5	99.0
60	93.8	99.3	91.5	98.9

Table 3.27: Fullness of a 32×32 structure with different pairs of polyominoes

Polyomino type	Fullness by GLA, %	Fullness by SA, %
L-tromino and L-octomino	93.2	99.2
L-tromino and Pu-octomino	90.8	98.9
L-tetromino and L-octomino	90.5	98.8
L-tetromino and Pu-octomino	88.8	98.5
S-tetromino and L-octomino	87.6	98.5
S-tetromino and Pu-octomino	87.2	98.4
T-tetromino and L-octomino	87.7	97.8
T-tetromino and Pu-octomino	87.7	97.6

3.6 Chapter 3 conclusions

1. The questions were described that arise with application of the genetic algorithm and sequence of solving provided. The Gwee—Lim algorithm for tiling rectangular structures with polyominoes was described in details including circular placement principle and fitness function. Parameters have been calibrated and example results shown. The analysis of the examples revealed that this algorithm is not able

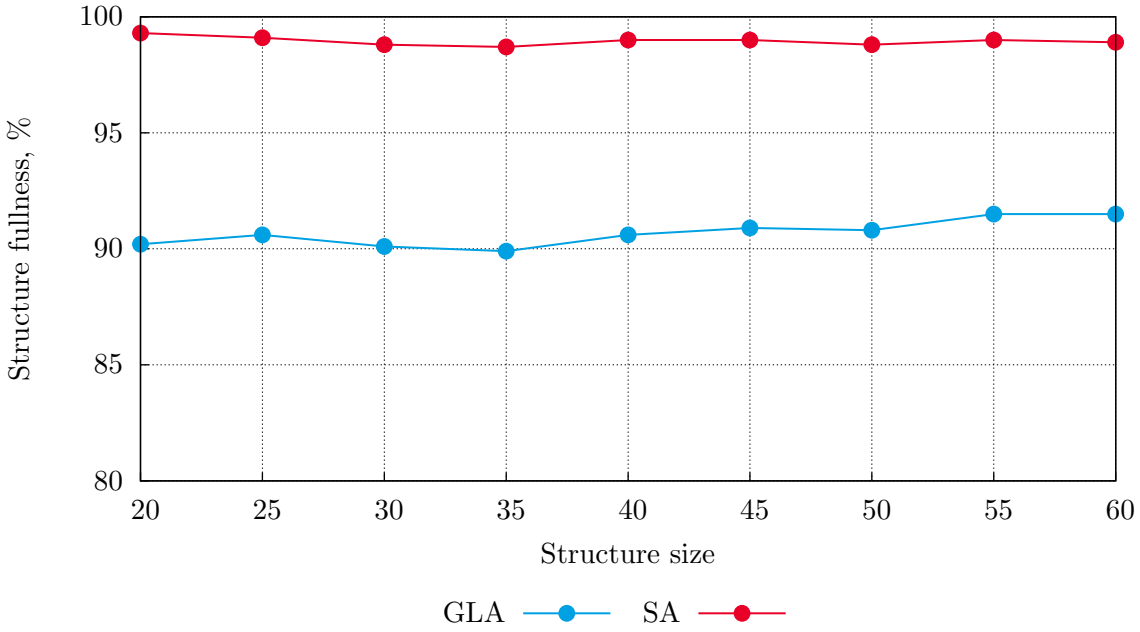


Figure 3.42: Fullness of structures of different sizes by L-tetrominoes and L-octominoes

to fill structures with polyominoes well and that it is necessary to develop another algorithm to solve this problem.

2. The "Snowball" algorithm was developed, that differs by the way of polyomino placement that provides better structure filling in comparison with the Gwee—Lim algorithm. Parameters of the algorithm have been calibrated. Examples have been obtained and shown. Analysis of the examples shows that structure fullness grew up to 98–100%.

3. The comparative analysis of functioning of two algorithms has been performed from the point of view of fullness of the obtained structures. The comparison has shown that the snowball algorithm tiles the structures in average 10% better, than the Gwee—Lim algorithm.

4. We have described tiling of structures with two shapes of polyominoes simultaneously and corresponding adjustments in the algorithms. Examples of structures by both algorithms were shown. Use of two

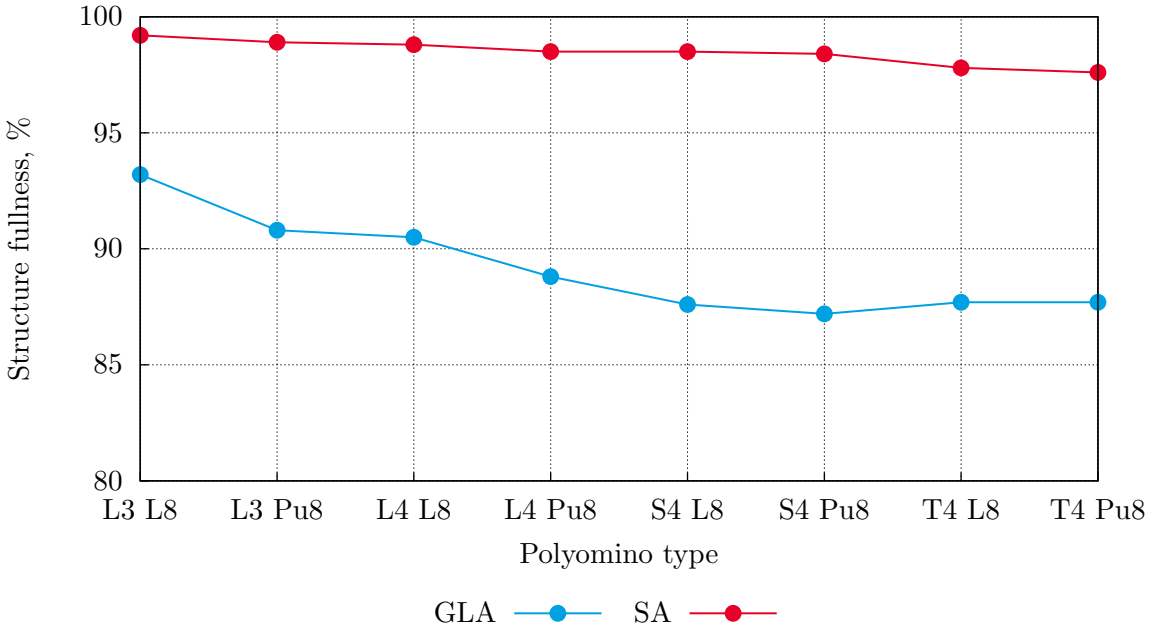


Figure 3.43: Fullness of a 32 × 32 structure with different pairs of polyominoes

shapes increased fullness of structures for both algorithms.

5. The comparative analysis of results by Gwee—Lim and snowball algorithms was performed from the point of view of structure fullness by tiling with two shapes of polyominoes simultaneously. The comparison showed that the snowball algorithm fills structures by 6–10% better than Gwee—Lim algorithm.

Chapter 4

Application of the optimization algorithms to antenna array design

In this chapter we apply the developed methods and algorithms for the design of phased arrays structures. First we show examples of radiation patterns of structures obtained using the Gwee—Lim algorithm. Similar experiments were carried out with the "Snowball" algorithm. Then, a comparative analysis of the two algorithms in terms of the characteristics of antenna arrays is presented. Main results of the chapter are published by author in journals and conference proceedings [17, 7, 1].

All results of this chapter, namely the structures of arrays, their fullness, patterns and sidelobe levels are obtained using the developed software. Software implements the developed mathematical model, describing the structure of phased arrays composed of subarrays in shape of polyominoes, developed optimization method for polyomino placement in structure by calculating the autocorrelation function, the snowball algorithm and the Gwee—Lim algorithm. The software contains a procedural engine that simulates the radiation patterns of PAA. Figure 4.1 shows the structure of the developed software. Arrows denote data exchange between the blocks. Software is written in C language and has a command line interface.

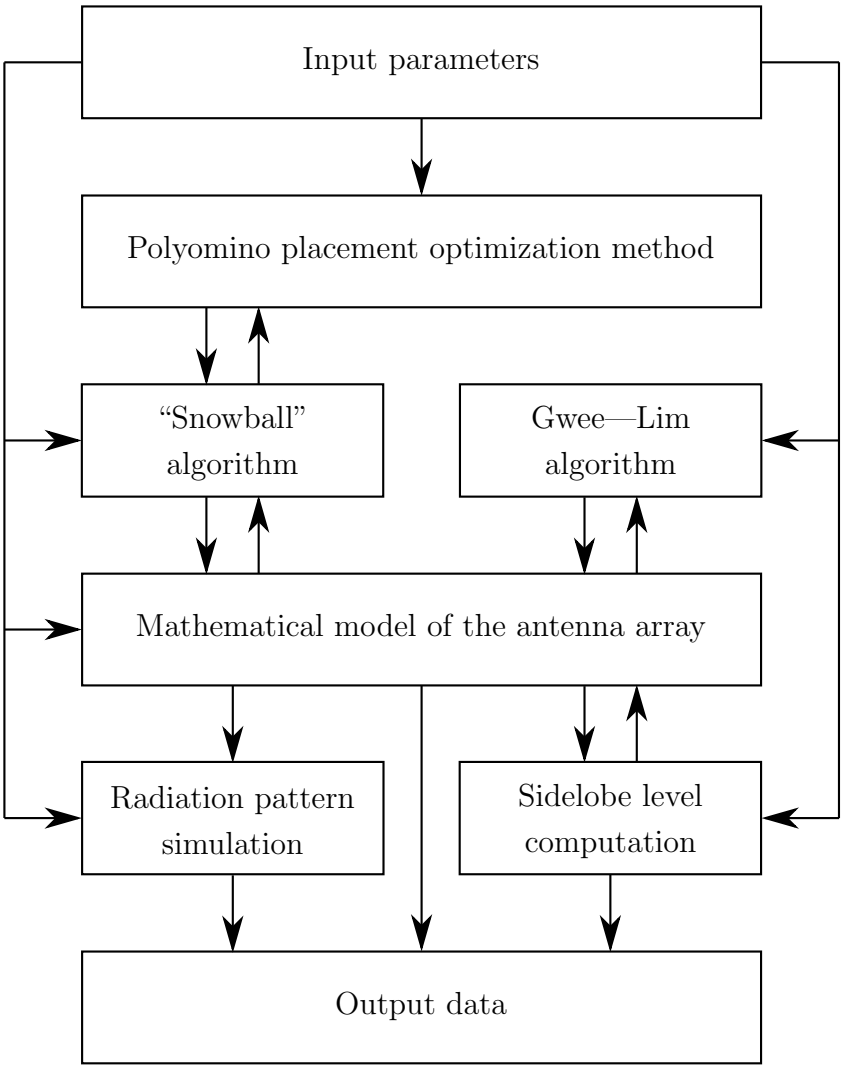


Figure 4.1: Structure of the software

4.1 Applying Gwee—Lim algorithm to phased antenna array optimization

In this paragraph we present examples of radiation patterns of those arrays, which were obtained by the Gwee—Lim algorithm. In total there are four examples with the following parameters:

- number of iterations: $K = 50$;
- population size: $P = 10$;

- elitism: on;
- seed: $s = 37$;
- SLL optimization: $r = 1.3$.

Calibrated values p_c, p_m and p_{bm} were used. Figure 4.2 shows the graph of the fitness function in the second example.

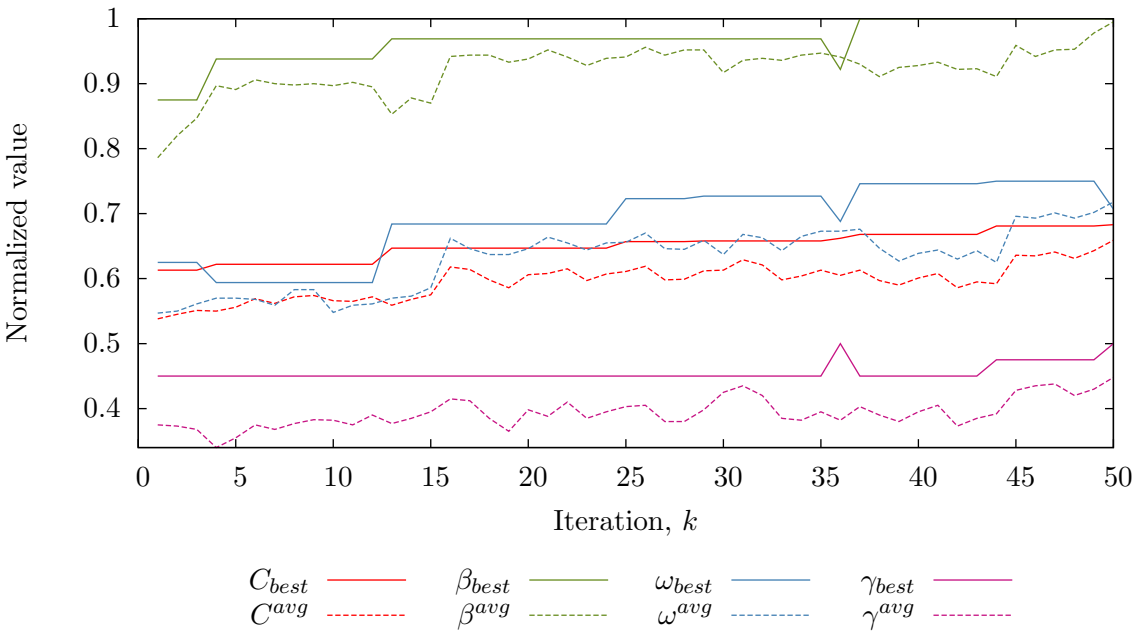


Figure 4.2: Fitness function in the second example

Example 1: Structure 16 × 16, L-tromino

Input parameters:

- structure size: $M = N = 16$;
- polyomino type: L-tromino;

Figure 4.3 shows the radiation pattern obtained by GLA with those parameters. Numerical results are provided in table 4.1.

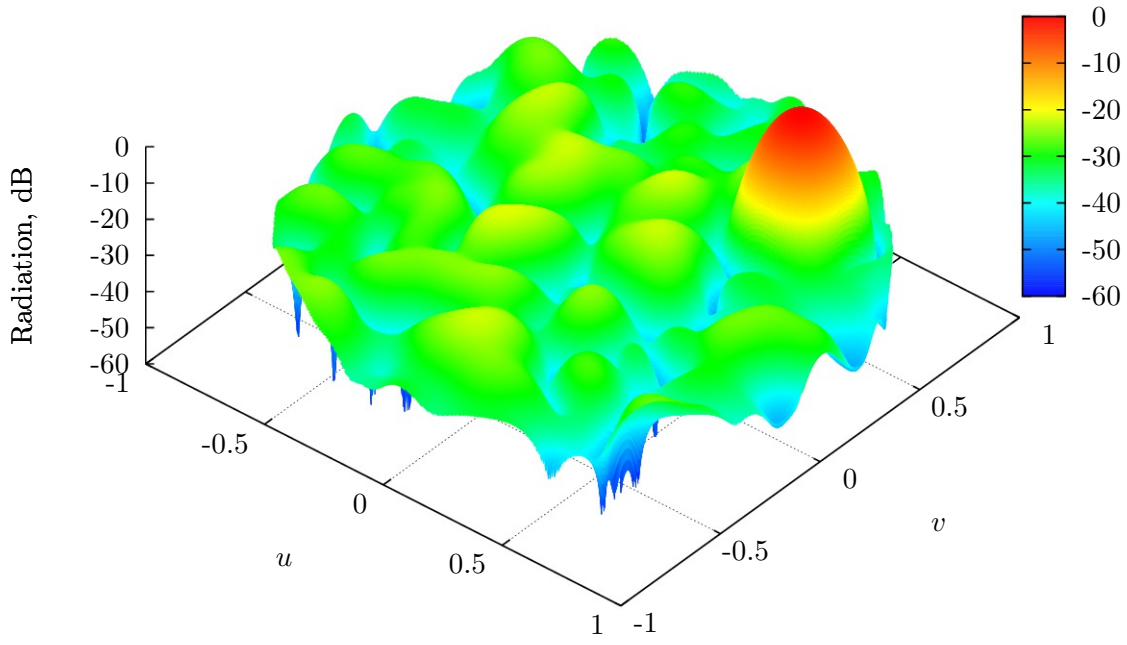


Figure 4.3: Radiation pattern in the first example at $r = 1.3$

Table 4.1: Output data of the first example

Parameter	Value
Number of polyominoes α	79
Fullness of the structure A, %	92.58
Sidelobe level γ at $r = 1.300$, dB	-21.13
Sidelobe level γ at $r = 1.818$, dB	-16.86

Example 2: Structure 16 × 16, L-octomino

Input parameters:

- structure size: $M = N = 16$;
- polyomino type: L-octomino;

Figure 4.4 shows the radiation pattern obtained by GLA with those parameters. Numerical results are provided in table 4.2.

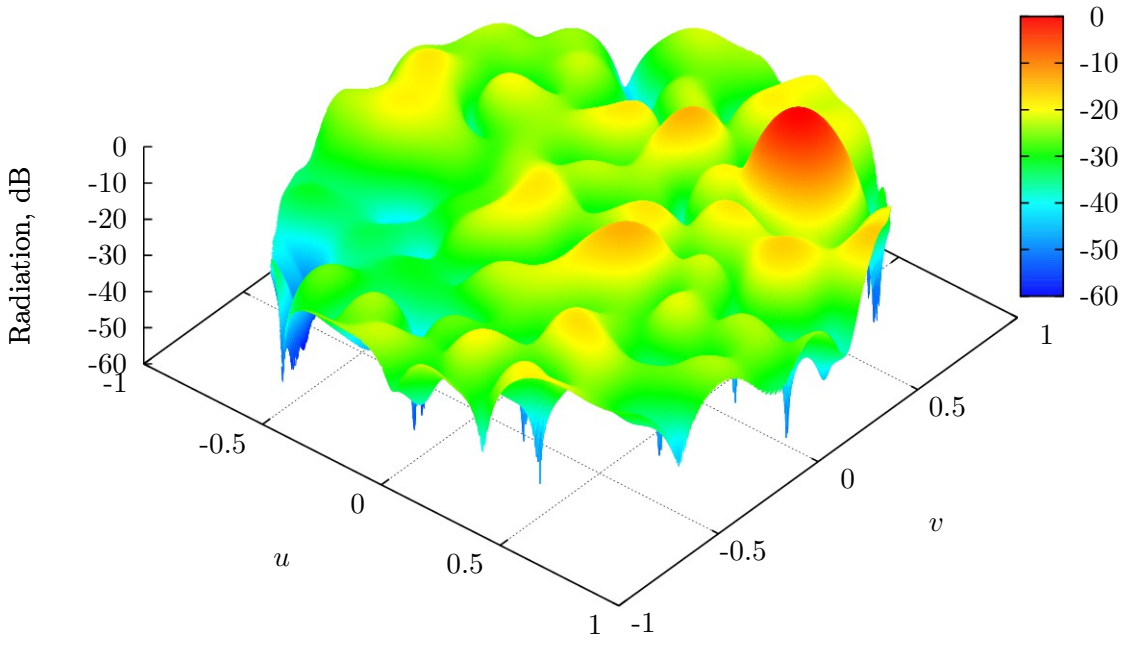


Figure 4.4: Radiation pattern in the second example at $r = 1.3$

Table 4.2: Output data of the second example

Parameter	Value
Number of polyominoes α	26
Fullness of the structure A, %	81.25
Sidelobe level γ at $r = 1.300$, dB	-18.16
Sidelobe level γ at $r = 1.818$, dB	-10.10

Example 3: Structure 32 × 32, L-tromino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-tromino;

Figure 4.5 shows the radiation pattern obtained by GLA with those parameters. Numerical results are provided in table 4.3.

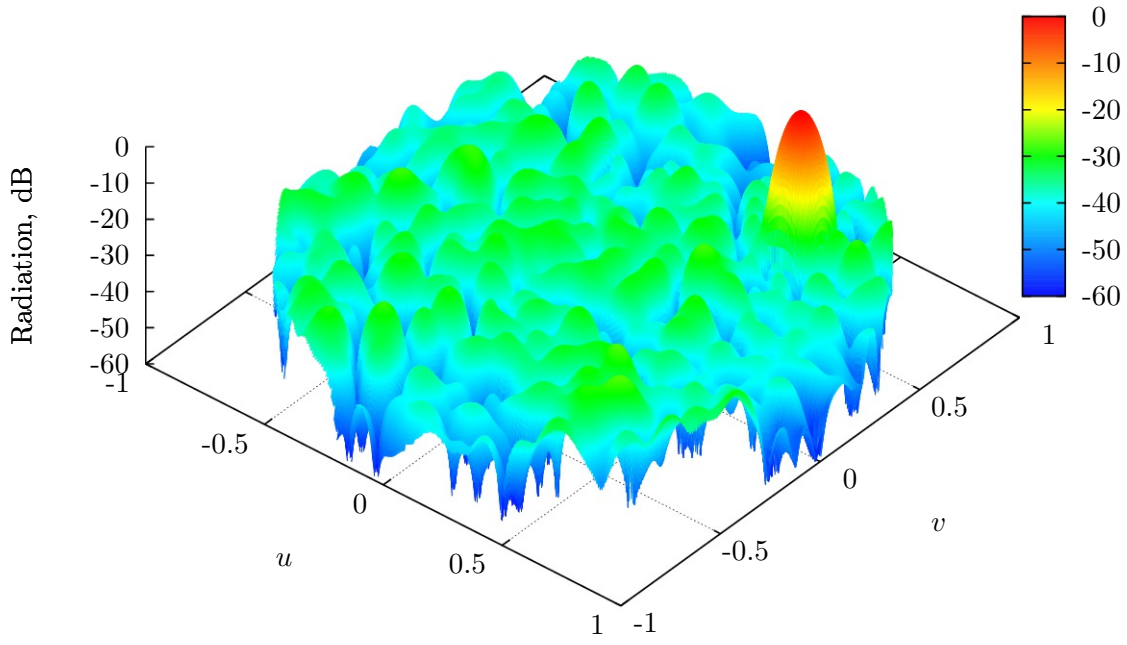


Figure 4.5: Radiation pattern in the third example at $r = 1.3$

Table 4.3: Output data of the third example

Parameter	Value
Number of polyominoes α	323
Fullness of the structure A, %	94.63
Sidelobe level γ at $r = 1.300$, dB	-27.65
Sidelobe level γ at $r = 1.818$, dB	-20.96

Example 4: Structure 32 × 32, L-octomino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-octomino;

Figure 4.6 shows the radiation pattern obtained by GLA with those parameters. Numerical results are provided in table 4.4.

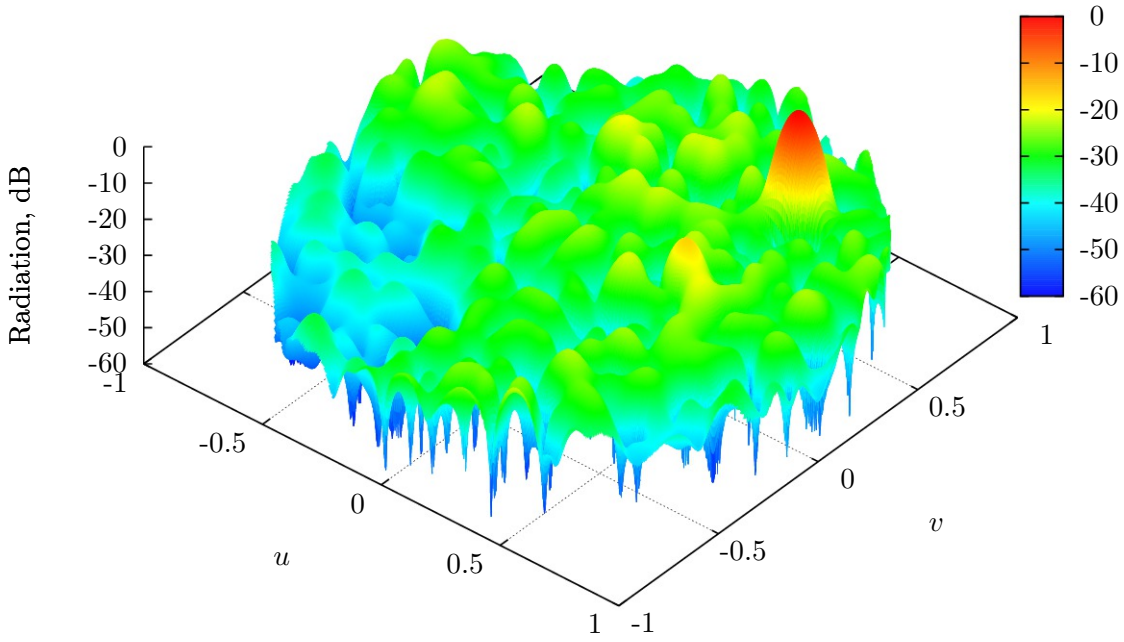


Figure 4.6: Radiation pattern in the fourth example at $r = 1.3$

Table 4.4: Output data of the fourth example

Parameter	Value
Number of polyominoes α	106
Fullness of the structure A, %	82.81
Sidelobe level γ at $r = 1.300$, dB	-21.45
Sidelobe level γ at $r = 1.818$, dB	-13.51

Therefore, examples of radiation patterns and parameters of antenna arrays were listed, which were obtained by the Gwee—Lim algorithm. The analysis shows that sidelobe level at band $r = 1.818$ is too high for structures tiled with L-octomino.

4.2 Applying "Snowball" algorithm to phased antenna array optimization

In this paragraph we present examples of radiation patterns of those arrays, which were obtained by the snowball algorithm. In total there are four examples. The same parameters were used as in the previous paragraph.

Figure 4.7 shows the graph of fitness function and its components in the second example.

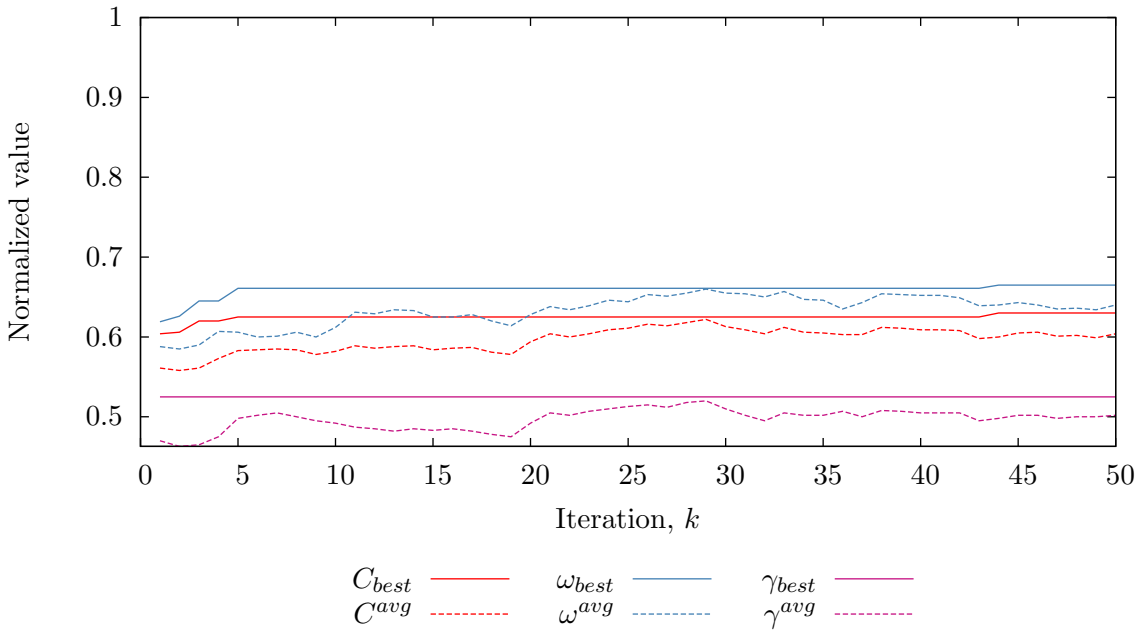


Figure 4.7: Fitness function in the second example

Example 1: Structure 16 × 16, L-tromino

Input parameters:

- structure size: $M = N = 16$;
- polyomino type: L-tromino;

Figure 4.8 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 4.5.

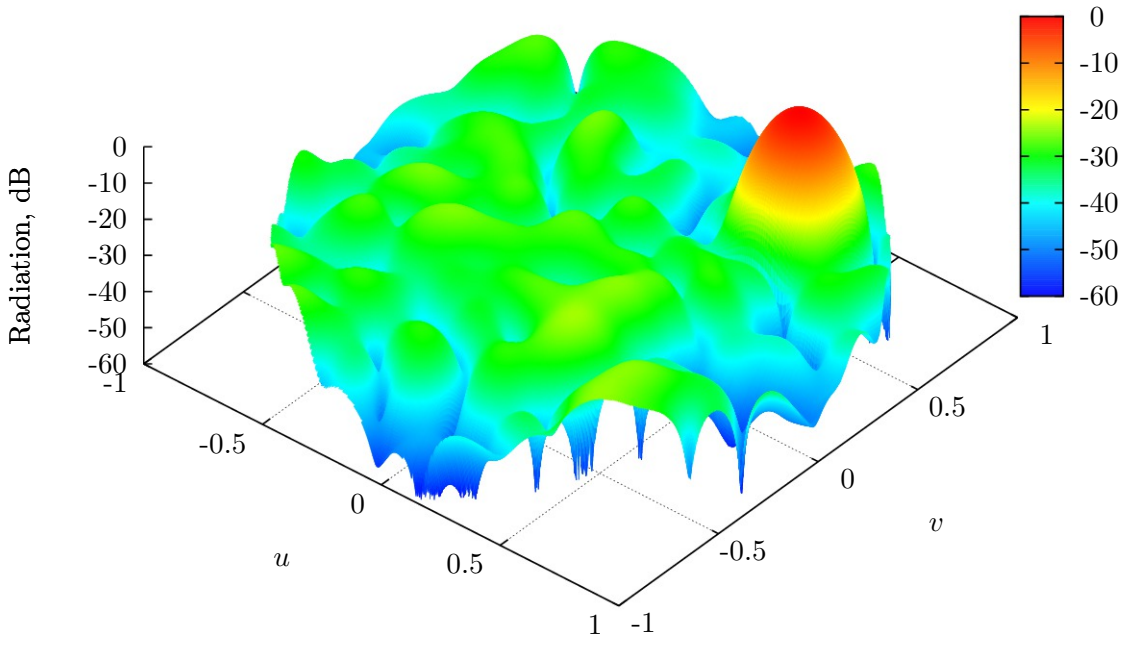


Figure 4.8: Radiation pattern in the first example at $r = 1.3$

Table 4.5: Output data of the first example

Parameter	Value
Number of polyominoes α	92
Fullness of the structure A, %	98.83
Sidelobe level γ at $r = 1.300$, dB	-25.08
Sidelobe level γ at $r = 1.818$, dB	-17.85

Example 2: Structure 16 × 16, L-octomino

Input parameters:

- structure size: $M = N = 16$;
- polyomino type: L-octomino;

Figure 4.9 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 4.6.

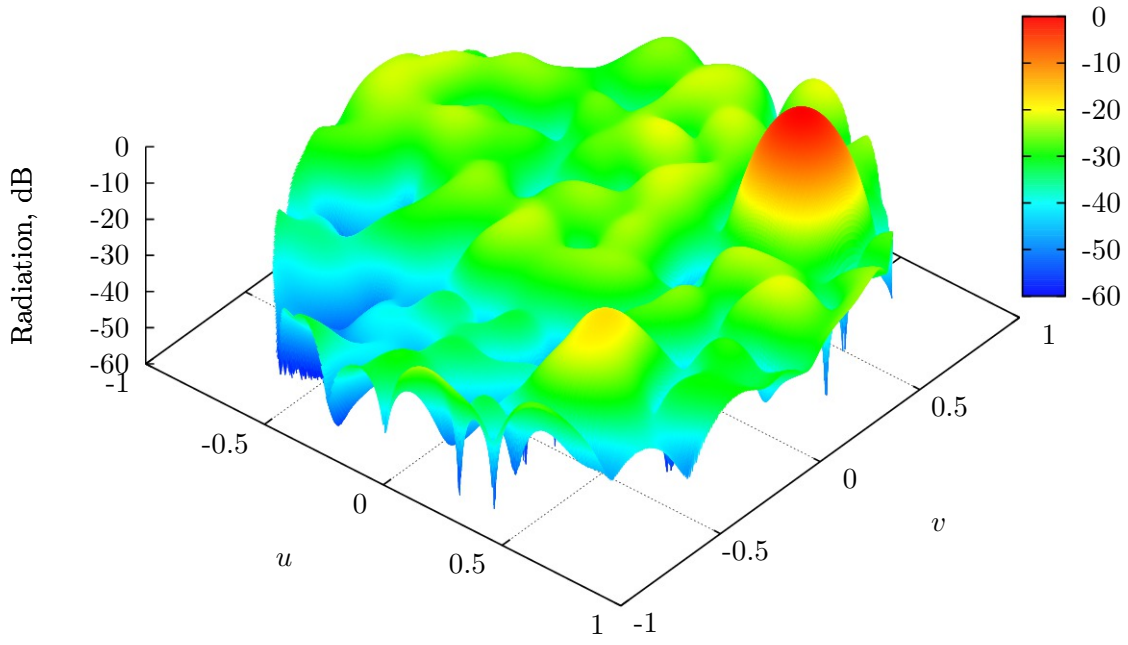


Figure 4.9: Radiation pattern in the second example at $r = 1.3$

Table 4.6: Output data of the second example

Parameter	Value
Number of polyominoes α	41
Fullness of the structure A, %	98.83
Sidelobe level γ at $r = 1.300$, dB	-21.05
Sidelobe level γ at $r = 1.818$, dB	-13.16

Example 3: Structure 32 × 32, L-tromino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-tromino;

Figure 4.10 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 4.7.

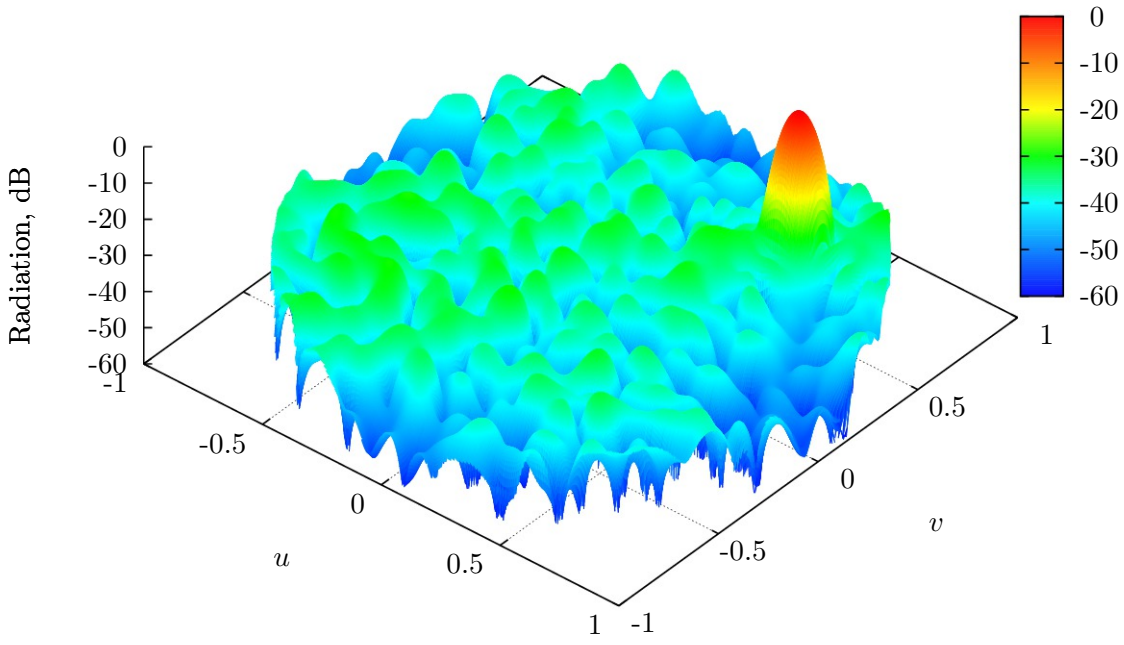


Figure 4.10: Radiation pattern in the third example at $r = 1.3$

Table 4.7: Output data of the third example

Parameter	Value
Number of polyominoes α	360
Fullness of the structure A, %	99.71
Sidelobe level γ at $r = 1.300$, dB	-29.18
Sidelobe level γ at $r = 1.818$, dB	-22.36

Example 4: Structure 32 × 32, L-octomino

Input parameters:

- structure size: $M = N = 32$;
- polyomino type: L-octomino;

Figure 4.11 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 4.8.

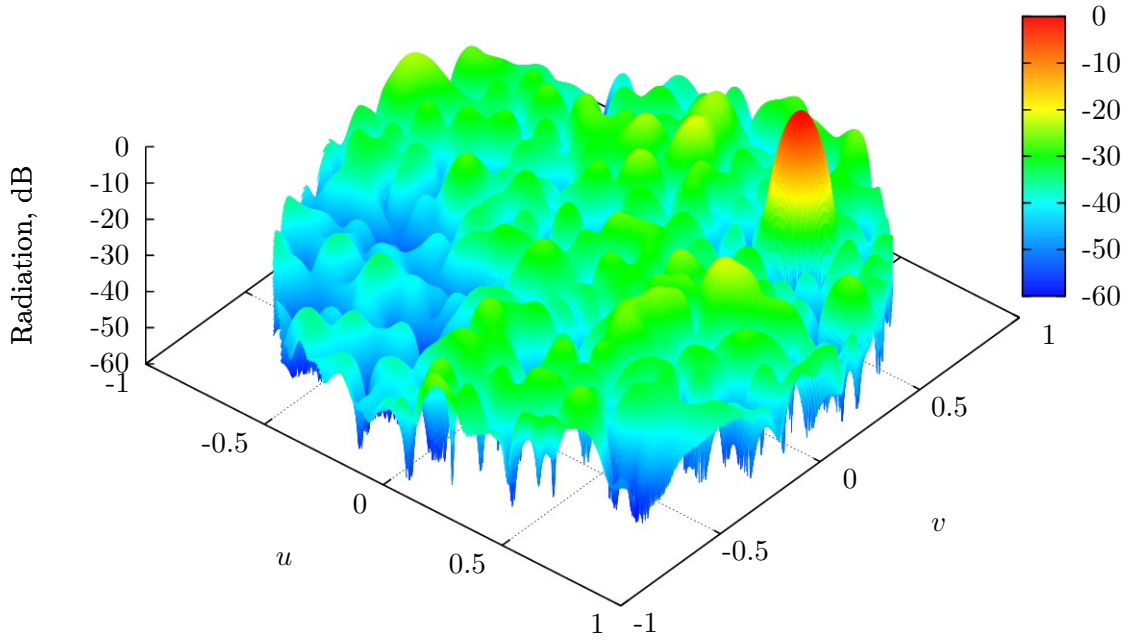


Figure 4.11: Radiation pattern in the fourth example at $r = 1.3$

Table 4.8: Output data of the fourth example

Parameter	Value
Number of polyominoes α	142
Fullness of the structure A, %	97.17
Sidelobe level γ at $r = 1.300$, dB	-23.18
Sidelobe level γ at $r = 1.818$, dB	-16.18

Therefore, examples of radiation patterns and parameters of antenna arrays were listed, which were obtained by the snowball algorithm. The analysis shows that the snowball algorithm can synthesize structures with better sidelobe level suppression in the radiation pattern.

4.3 Comparison of the Gwee—Lim and "Snowball" algorithms by the sidelobe level

In this paragraph we perform a comparison of the Gwee—Lim and "Snowball" algorithms from the point of view of radiation pattern forming. The following parameters were used:

- structure size: $M = N = \{20,\ 25,\ 30,\ 35,\ 40,\ 45,\ 50,\ 55,\ 60\}$;
- number of iterations: $K = 50$;
- population size: $P = 10$;
- elitism: on;
- seed: $s = \{10,\ 20,\ 30,\ 40,\ 50,\ 60,\ 70\}$.

Obtained value of the sidelobe level for each structure size was averaged over seven different seeds of PRNG. Each algorithm used its calibrated parameters.

Figure 4.12 shows the graph of average sidelobe level of structures tiled with L-trominoes. SLL was optimized for $r = 1.3$. A graph of SLL for the same structures but optimized for $r = 1.818$ is shown in figure 4.13. Numerical data is provided in table 4.10. It is seen that SA demonstrates better SLL for structures of all sizes.

Figure 4.14 shows the graph of average sidelobe level of structures tiled with L-octominoes. SLL was optimized for $r = 1.3$. A graph of SLL for

the same structures but optimized for $r = 1.818$ is shown in figure 4.15. Numerical data is provided in table 4.12. Here again SA outperforms GLA and results for $r = 1.818$ got better.

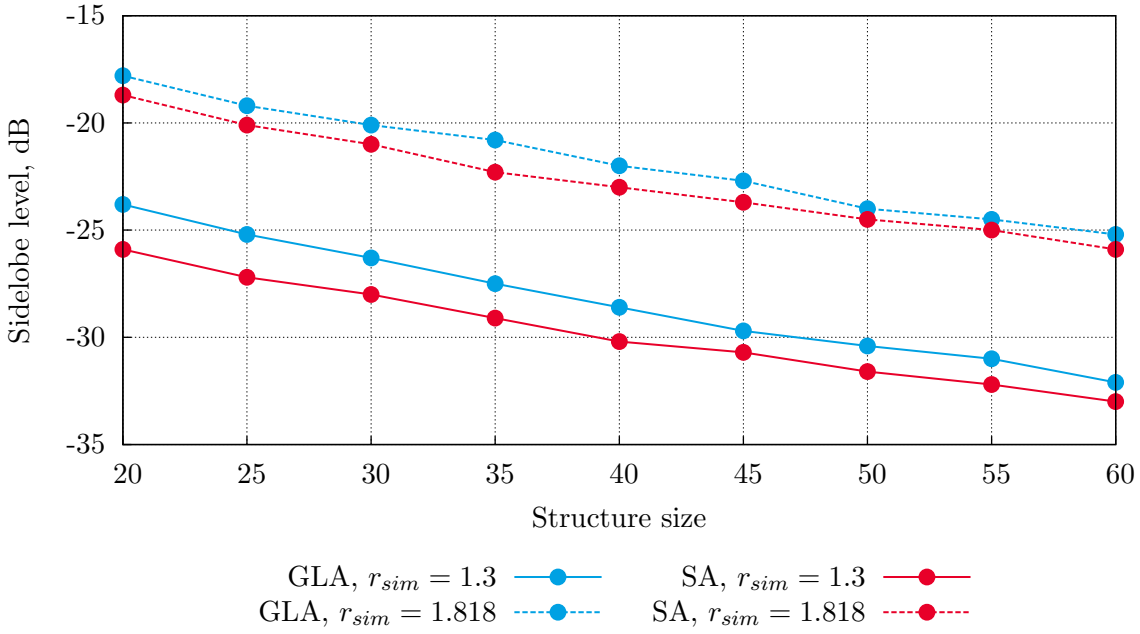


Figure 4.12: SLL of structures tiled with L-trominoes, $r_{opt} = 1.3$

Table 4.9: SLL of structures tiled with L-trominoes, $r_{opt} = 1.3$

Structure size	GLA		SA	
	$r_{sim} = 1.3$, dB	$r_{sim} = 1.818$, dB	$r_{sim} = 1.3$, dB	$r_{sim} = 1.818$, dB
20	−23.8	−17.8	−25.9	−18.7
25	−25.2	−19.2	−27.2	−20.1
30	−26.3	−20.1	−28.0	−21.0
35	−27.5	−20.8	−29.1	−22.3
40	−28.6	−22.0	−30.2	−23.0
45	−29.7	−22.7	−30.7	−23.7
50	−30.4	−24.0	−31.6	−24.5
55	−31.0	−24.5	−32.2	−25.0
60	−32.1	−25.2	−33.0	−25.9

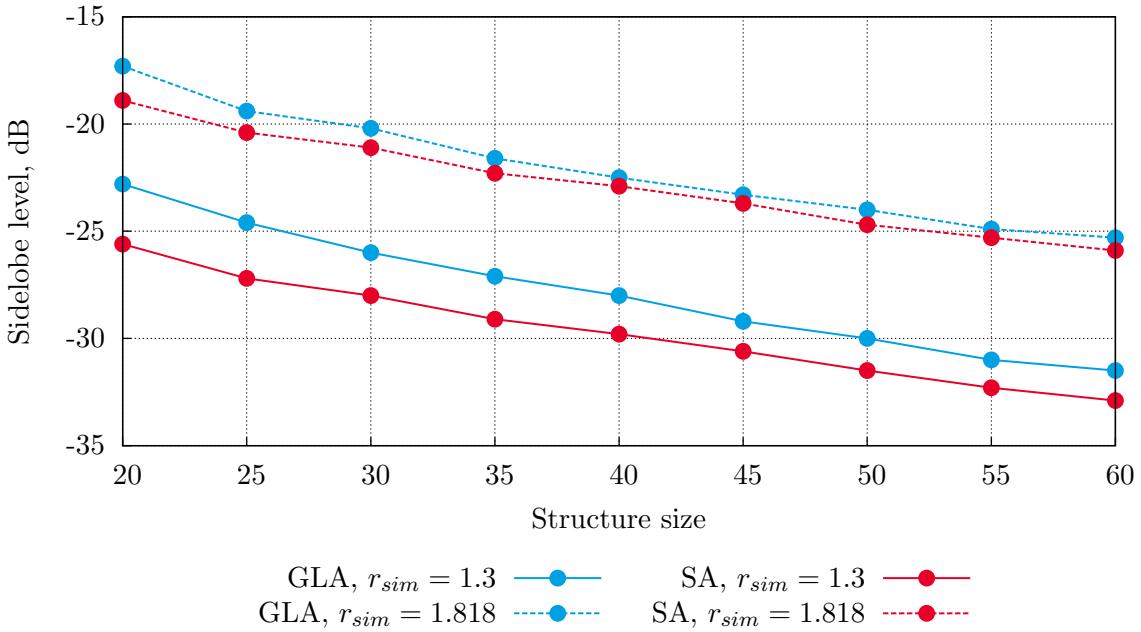


Figure 4.13: SLL of structures tiled with L-trominoes, $r_{opt} = 1.818$

Table 4.10: SLL of structures tiled with L-trominoes, $r_{opt} = 1.818$

Structure size	GLA		SA	
	$r_{sim} = 1.3$, dB	$r_{sim} = 1.818$, dB	$r_{sim} = 1.3$, dB	$r_{sim} = 1.818$, dB
20	−22.8	−17.3	−25.6	−18.9
25	−24.6	−19.4	−27.2	−20.4
30	−26.0	−20.2	−28.0	−21.1
35	−27.1	−21.6	−29.1	−22.3
40	−28.0	−22.5	−29.8	−22.9
45	−29.2	−23.3	−30.6	−23.7
50	−30.0	−24.0	−31.5	−24.7
55	−31.0	−24.9	−32.3	−25.3
60	−31.5	−25.3	−32.9	−25.9

This analysis is the main and final in the research work. It proves several important points that have been stated in previous chapters:

1. The developed snowball algorithm is able to optimize complex struc-

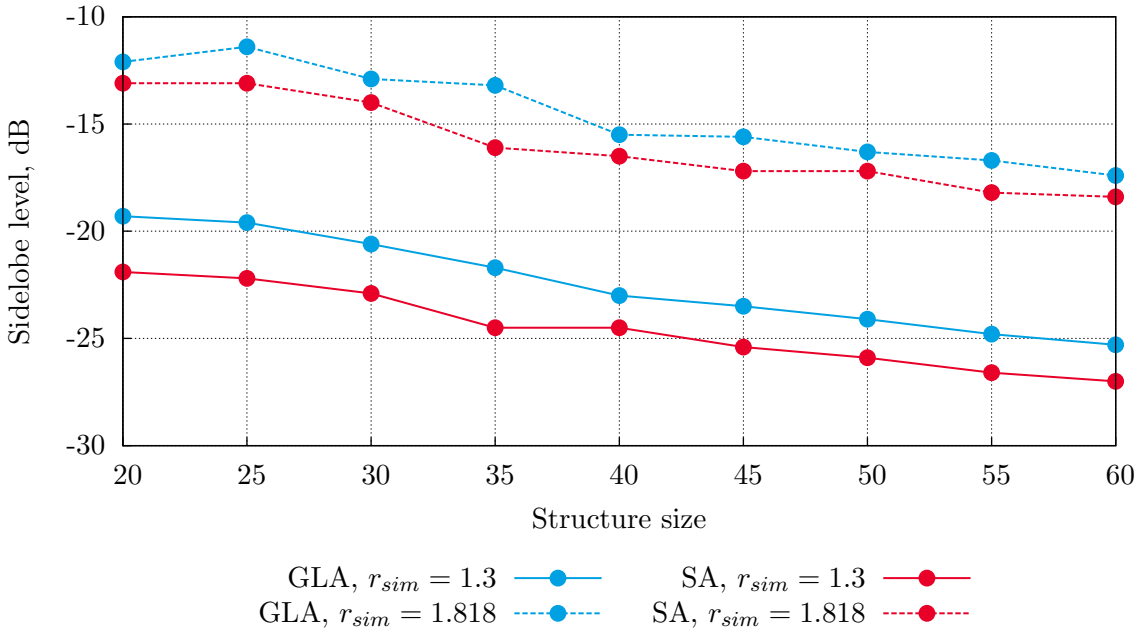


Figure 4.14: SLL of structures tiled with L-octominoes, $r_{opt} = 1.3$

Table 4.11: SLL of structures tiled with L-octominoes, $r_{opt} = 1.3$

Structure size	GLA		SA	
	$r_{sim} = 1.3$, dB	$r_{sim} = 1.818$, dB	$r_{sim} = 1.3$, dB	$r_{sim} = 1.818$, dB
20	−19.3	−12.1	−21.9	−13.1
25	−19.6	−11.4	−22.2	−13.1
30	−20.6	−12.9	−22.9	−14.0
35	−21.7	−13.2	−24.5	−16.1
40	−23.0	−15.5	−24.5	−16.5
45	−23.5	−15.6	−25.4	−17.2
50	−24.1	−16.3	−25.9	−17.2
55	−24.8	−16.7	−26.6	−18.2
60	−25.3	−17.4	−27.0	−18.4

tures tiled with polyominoes. At the same time fullness of structures increases as well as side lobes are suppressed better in the radiation pattern.

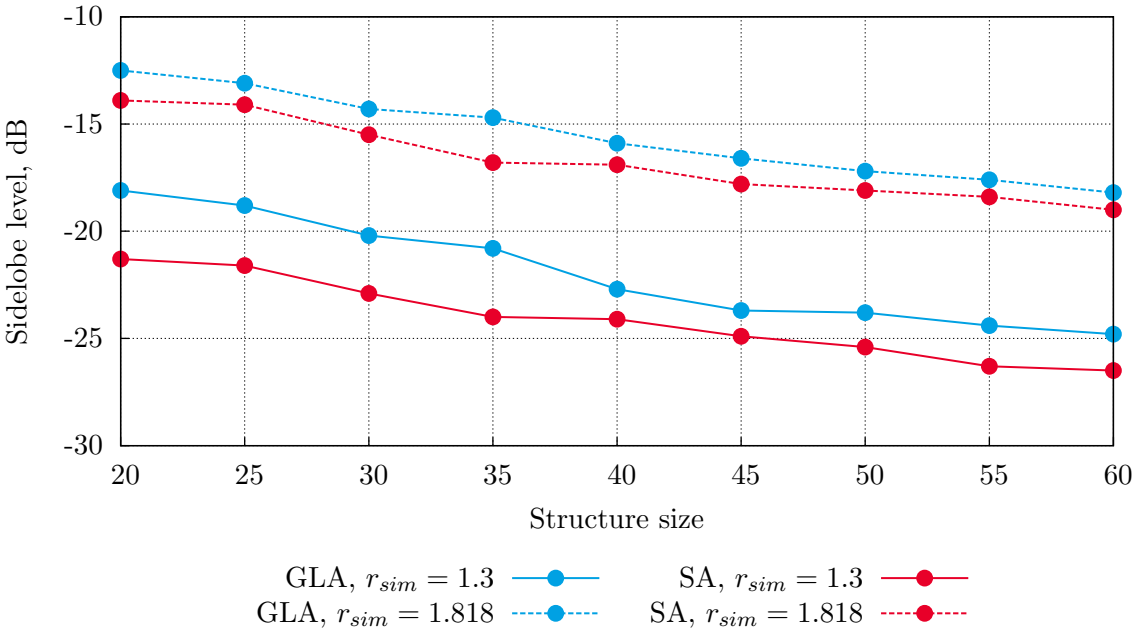


Figure 4.15: SLL of structures tiled with L-octominoes, $r_{opt} = 1.818$

Table 4.12: SLL of structures tiled with L-octominoes, $r_{opt} = 1.818$

Structure size	GLA		SA	
	$r_{sim} = 1.3$, dB	$r_{sim} = 1.818$, dB	$r_{sim} = 1.3$, dB	$r_{sim} = 1.818$, dB
20	−18.1	−12.5	−21.3	−13.9
25	−18.8	−13.1	−21.6	−14.1
30	−20.2	−14.3	−22.9	−15.5
35	−20.8	−14.7	−24.0	−16.8
40	−22.7	−15.9	−24.1	−16.9
45	−23.7	−16.6	−24.9	−17.8
50	−23.8	−17.2	−25.4	−18.1
55	−24.4	−17.6	−26.3	−18.4
60	−24.8	−18.2	−26.5	−19.0

2. SA outperforms GLA by all values for all structure sizes.

3. Sidelobe level decreases with the growth of structure size.

4. At a wider band $r = 1.818$ SLL is higher than at band $r = 1.3$ for one

and the same structure.

Finally, figure 4.16 shows the graph that compares averaged values of sidelobe levels by Gwee—Lim algorithm, snowball algorithm and results obtained by Robert Mailloux [59, 58] for L-octomino.

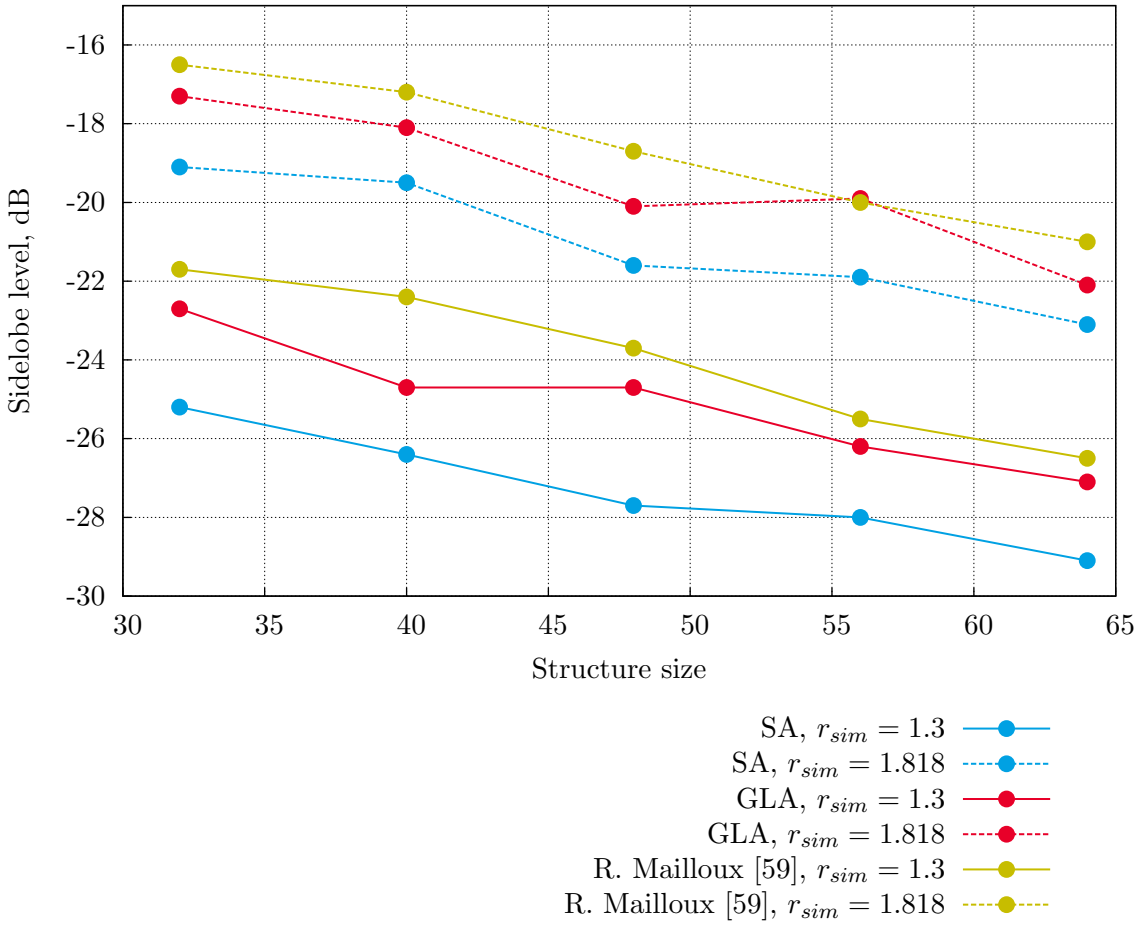


Figure 4.16: SLL comparison for structures of different sizes of L-octominoes, obtained by GLA, SA and R. Mailloux [59]

Therefore, there was done the final comparative analysis of the Gwee—Lim algorithm and the "Snowball" algorithm. The analysis from the point of view of radiation pattern forming and level of sidelobes has shown that the developed algorithm is able to synthesize antenna arrays with characteristics, outperforming existing analogues, including ones by the GLA

4.4 Analysis of steering capabilities

In this paragraph we present examples of radiation patterns of arrays, obtained by Snowball and steered at different angles. In total there are six examples. The following settings were applied:

- aperture size: $M = N = 32$;
- polyomino type: L-octomino;
- number of iterations: $K = 50$;
- population size: $P = 20$;
- seed: $s = 37$.

Calibrated values of p_c, p_m and p_{bm} were used. Figure 4.17 shows the graph of fitness function and its components in the first example.

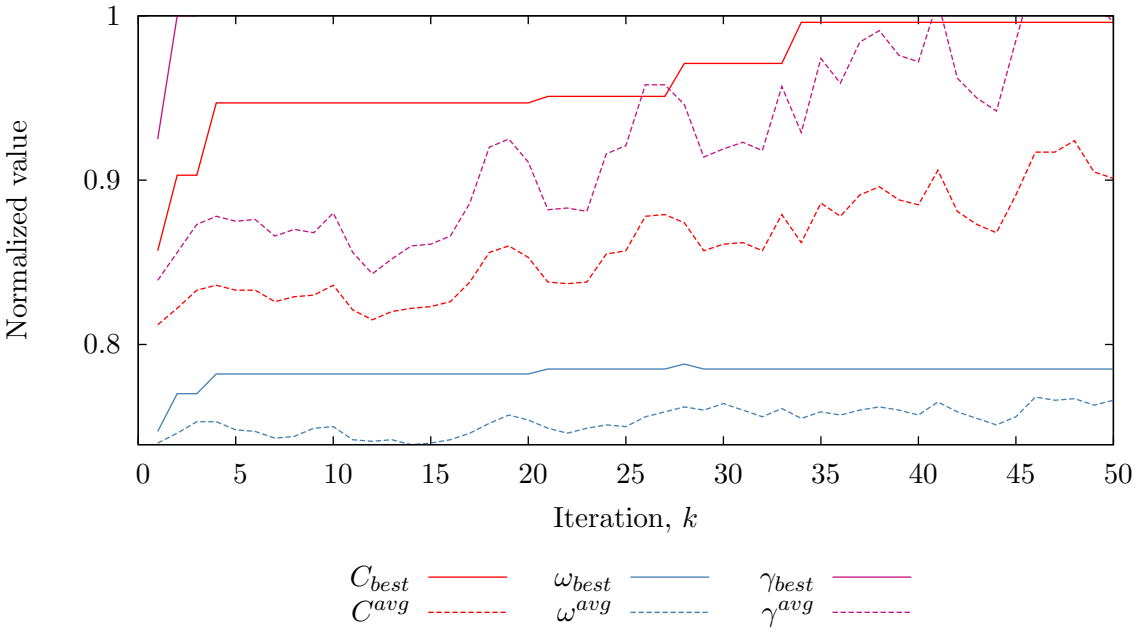


Figure 4.17: Fitness function in the first example

Example 1: steering at $(0°; 0°)$

Input parameters:

- steering: $(0°; 0°)$;
- SLL optimization: $r = 1.3$.

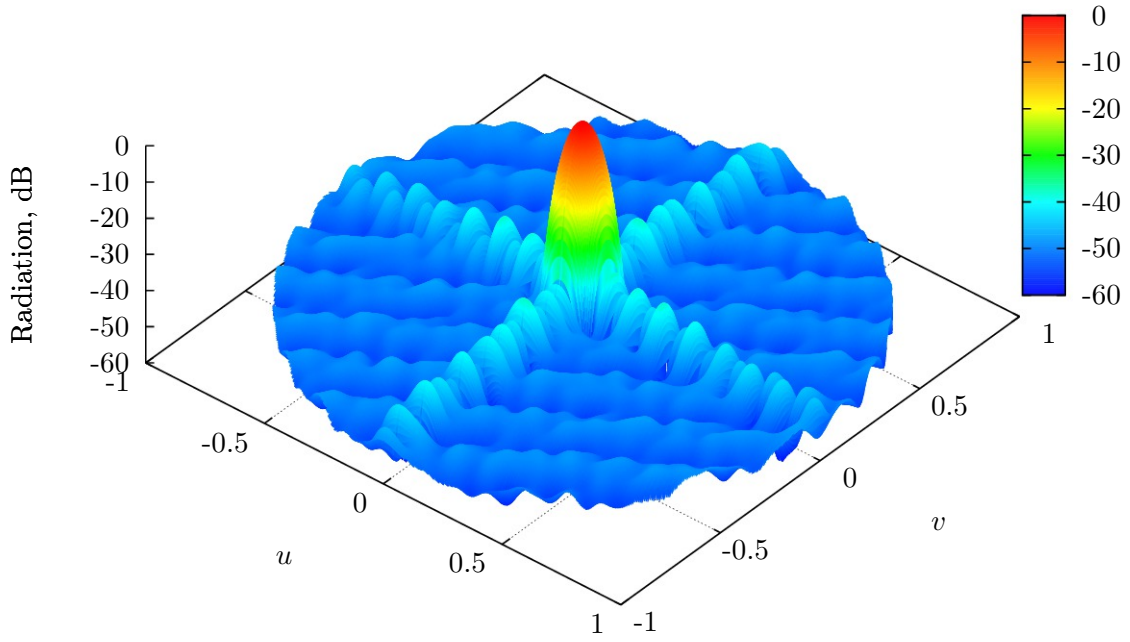


Figure 4.18: Radiation pattern in the first example at $r = 1.3$

Table 4.13: Output data of the first example

Parameter	Value
Number of polyominoes α	140
Fullness of the structure A, %	98.93
Sidelobe level γ at $r = 1.300$, dB	-47.22

Example 2: steering at $(15°; 15°)$

Input parameters:

- steering: $(15°; 15°)$;
- SLL optimization: $r = 1.3$.

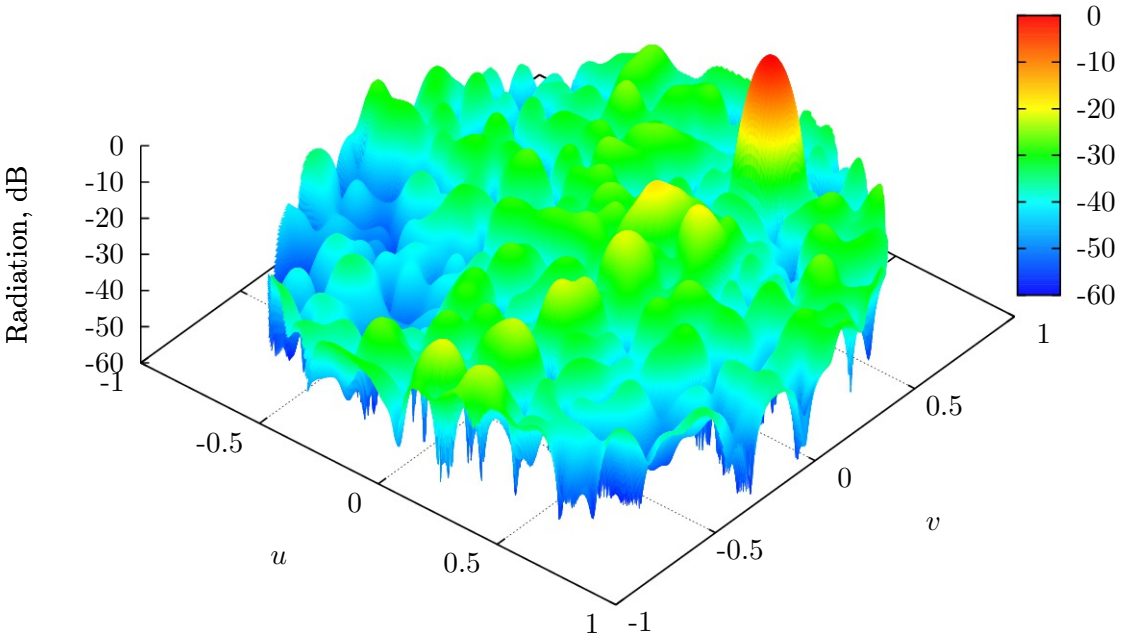


Figure 4.19: Radiation pattern in the second example at $r = 1.3$

Table 4.14: Output data of the second example

Parameter	Value
Number of polyominoes α	140
Fullness of the structure A, %	97.95
Sidelobe level γ at $r = 1.300$, dB	-25.20

Example 3: steering at $(30°; 30°)$

Input parameters:

- steering: $(30°; 30°)$;
- SLL optimization: $r = 1.3$.

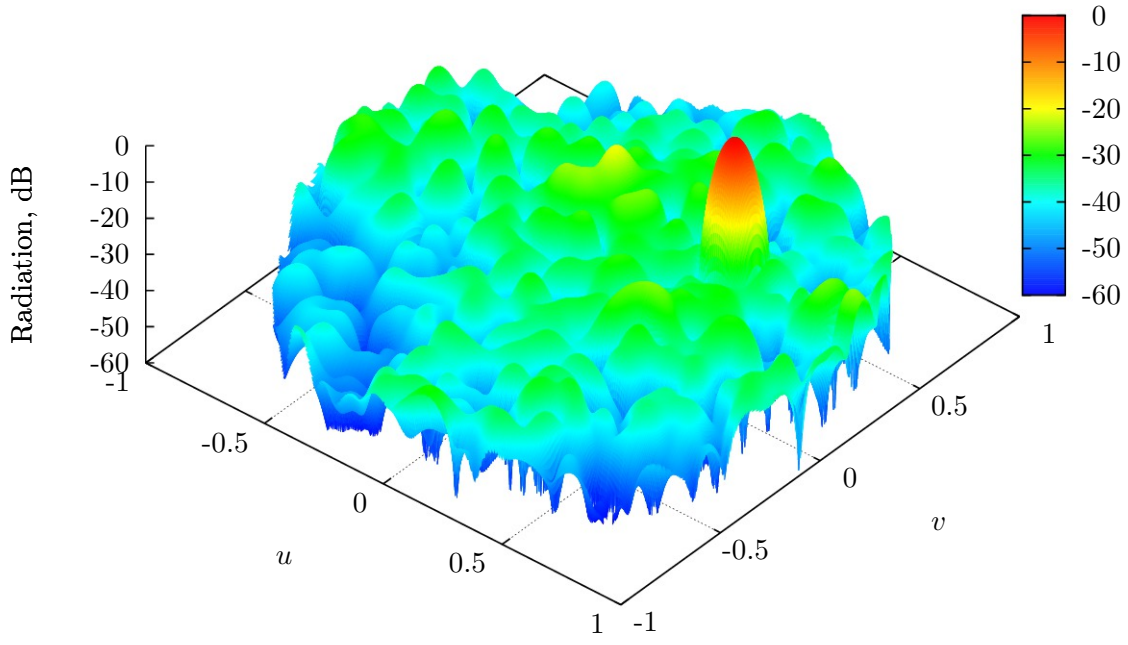


Figure 4.20: Radiation pattern in the third example at $r = 1.3$

Table 4.15: Output data of the third example

Parameter	Value
Number of polyominoes α	141
Fullness of the structure A, %	97.27
Sidelobe level γ at $r = 1.300$, dB	-28.00

Example 4: steering at $(45°; 45°)$

Input parameters:

- steering: $(45°; 45°)$;
- SLL optimization: $r = 1.3$.

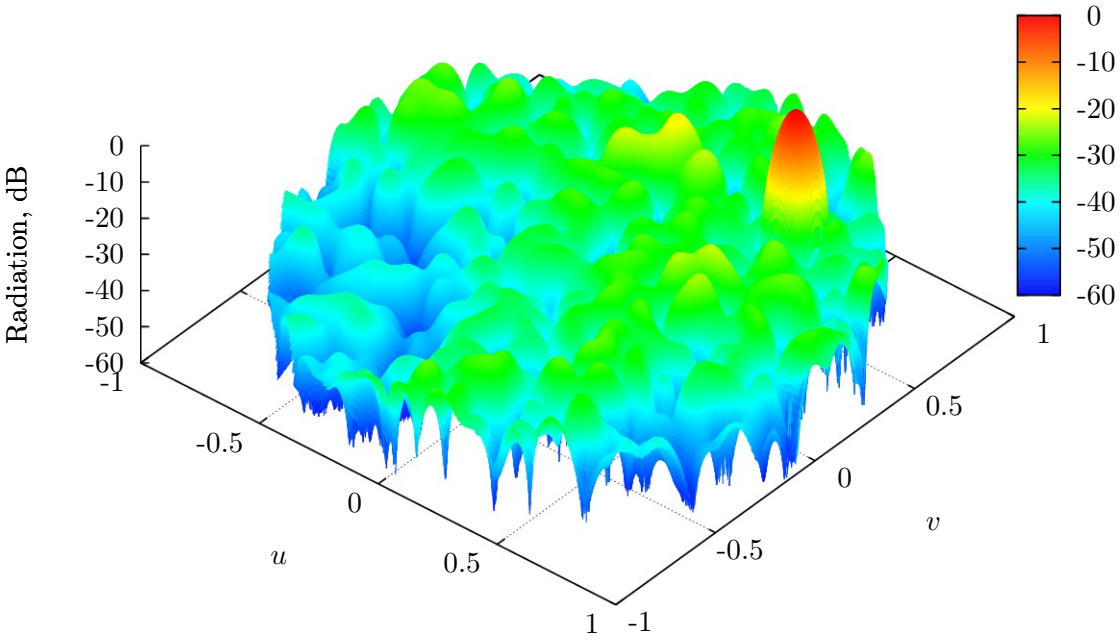


Figure 4.21: Radiation pattern in the fourth example at $r = 1.3$

Table 4.16: Output data of the fourth example

Parameter	Value
Number of polyominoes α	142
Fullness of the structure A, %	98.44
Sidelobe level γ at $r = 1.300$, dB	-25.05

Example 5: steering at $(60°; 60°)$

Input parameters:

- steering: $(60°; 60°)$;
- SLL optimization: $r = 1.3$.

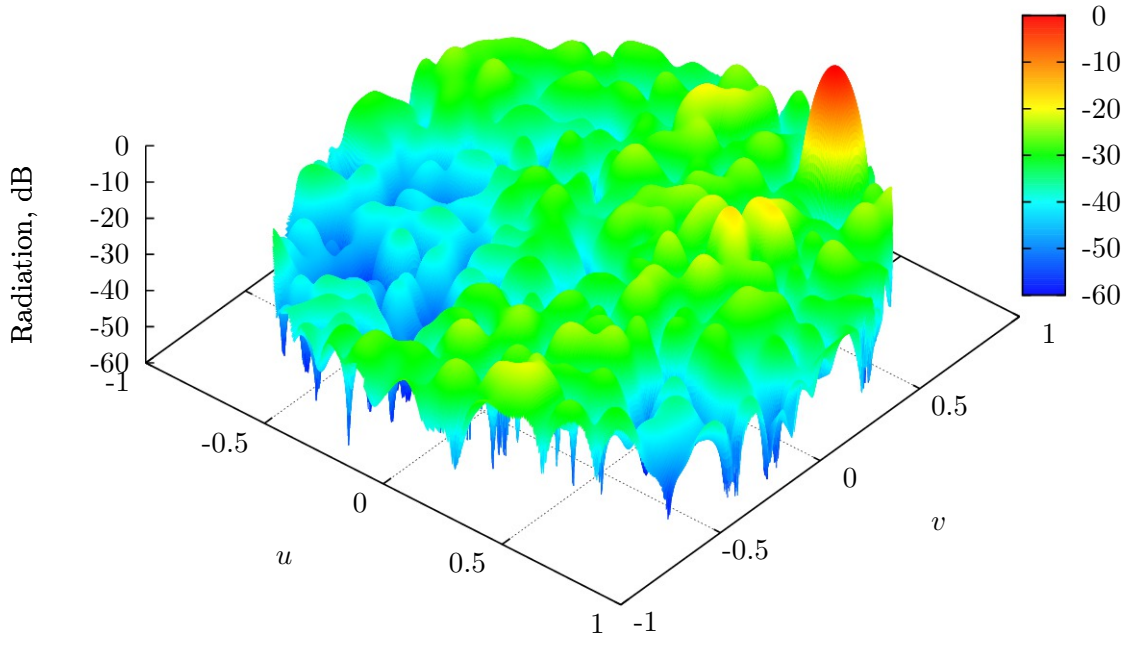


Figure 4.22: Radiation pattern in the fifth example at $r = 1.3$

Table 4.17: Output data of the fifth example

Parameter	Value
Number of polyominoes α	139
Fullness of the structure A, %	96.88
Sidelobe level γ at $r = 1.300$, dB	-23.08

Example 6: steering at $(75°; 75°)$

Input parameters:

- steering: $(75°; 75°)$;
- SLL optimization: $r = 1.3$.

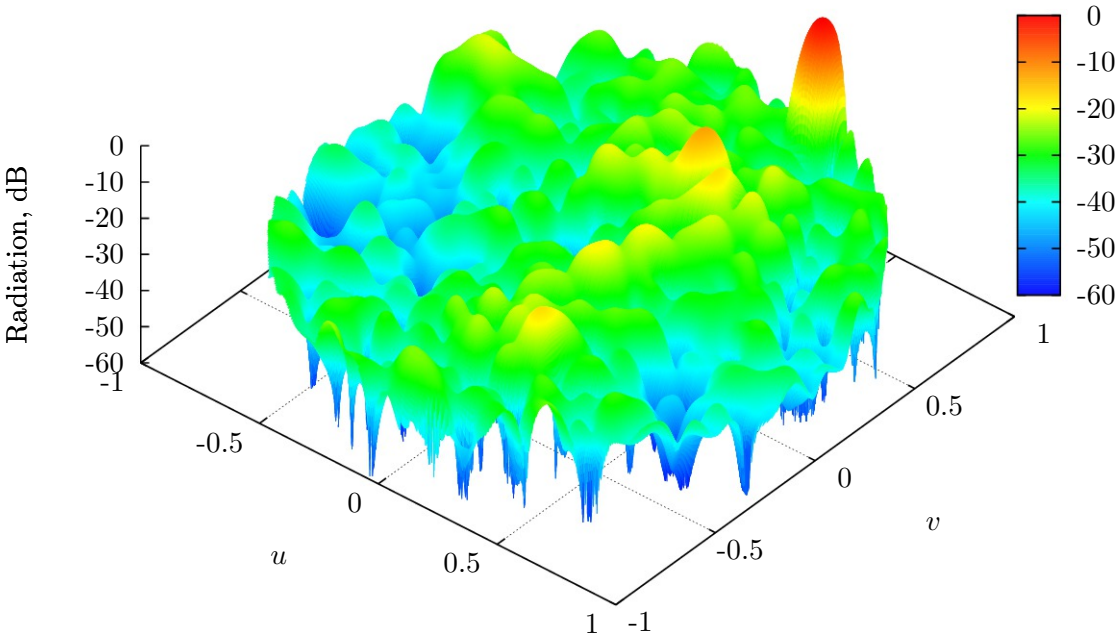


Figure 4.23: Radiation pattern in the sixth example at $r = 1.3$

Table 4.18: Output data of the sixth example

Parameter	Value
Number of polyominoes α	143
Fullness of the structure A, %	97.36
Sidelobe level γ at $r = 1.300$, dB	-20.59

Next, figure 4.24 presents the graph of sidelobe levels for 32×32 structures, averaged over seven different seeds $s = 10, 20, \ldots, 70$. The graph demonstrates a logarithmic rise of SLL.

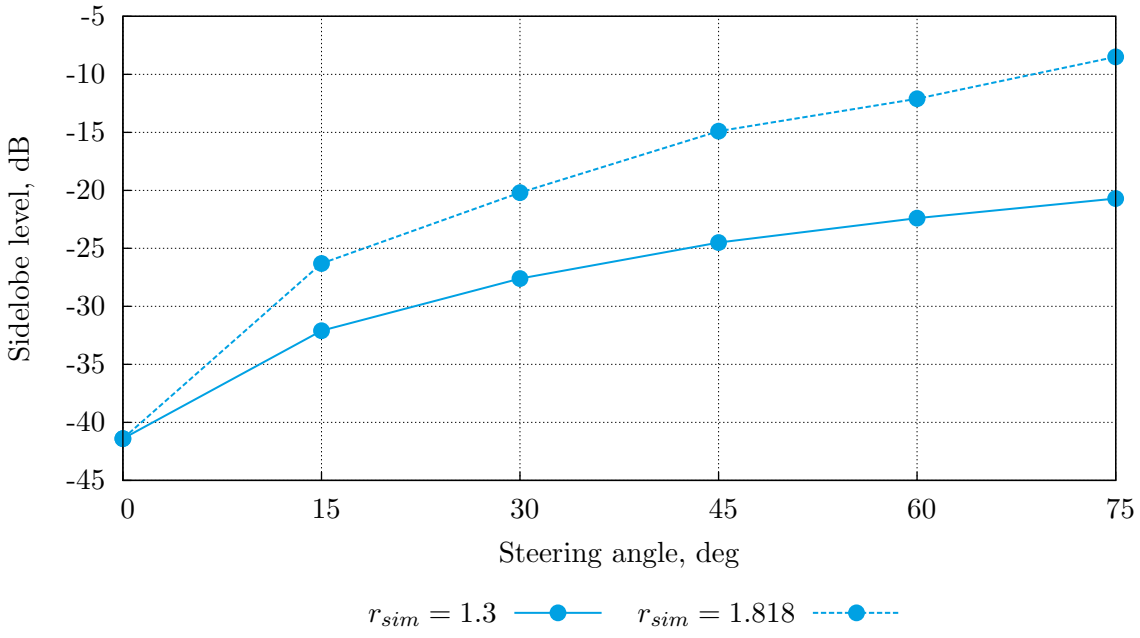


Figure 4.24: SLL at different steering angles

4.5 Analysis of bandwidth capabilities

In this paragraph we present examples of radiation patterns of 32×32 arrays, obtained by Snowball, optimized and simulated at different bandwidths. In total there are eight examples. The following settings were applied:

- polyomino type: L-tromino;
- number of iterations: $K = 50$;
- population size: $P = 20$;
- seed: $s = 37$.

Calibrated values of p_c, p_m and p_{bm} were used.

Example 1: bandwidth 2:1

Input parameters:

- SLL optimization: $r = 1.333$.

Figure 4.25 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 4.19.

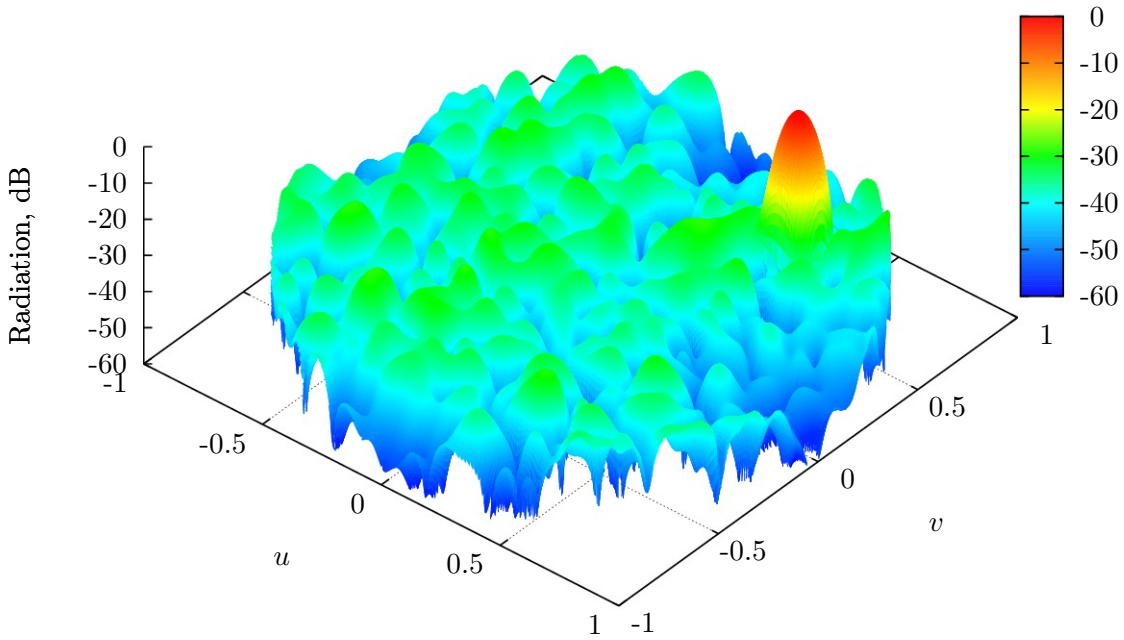


Figure 4.25: Radiation pattern in the first example at $r = 1.333$

Table 4.19: Output data of the first example

Parameter	Value
Number of polyominoes α	364
Fullness of the structure A, %	99.90
Sidelobe level γ at $r = 1.333$, dB	-28.34

Example 2: bandwidth 3:1

Input parameters:

- SLL optimization: $r = 1.500$.

Figure 4.26 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 4.20.

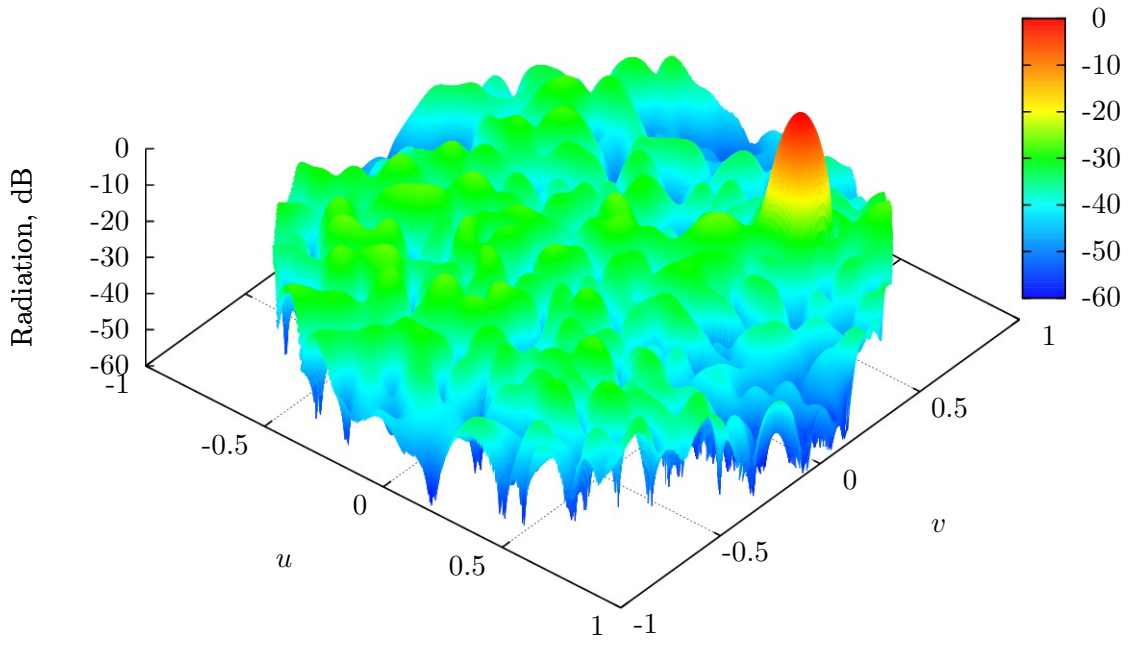


Figure 4.26: Radiation pattern in the second example at $r = 1.500$

Table 4.20: Output data of the second example

Parameter	Value
Number of polyominoes α	362
Fullness of the structure A, %	99.71
Sidelobe level γ at $r = 1.500$, dB	-26.03

Example 3: bandwidth 4:1

Input parameters:

- SLL optimization: $r = 1.600$.

Figure 4.27 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 4.21.

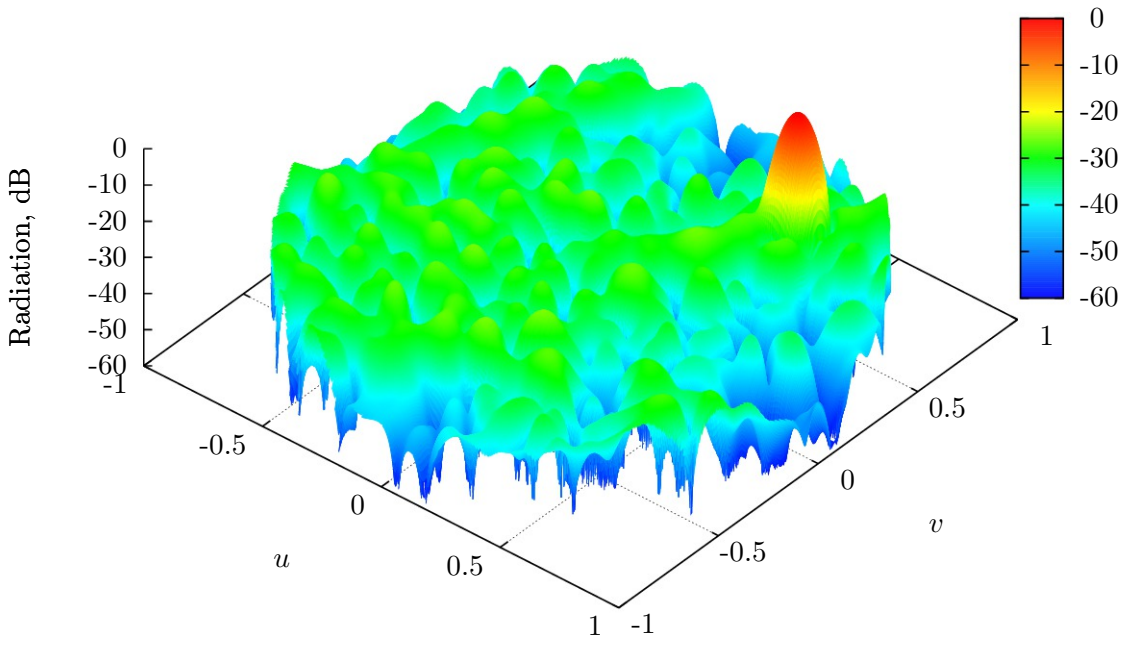


Figure 4.27: Radiation pattern in the third example at $r = 1.600$

Table 4.21: Output data of the third example

Parameter	Value
Number of polyominoes α	363
Fullness of the structure A, %	100.00
Sidelobe level γ at $r = 1.600$, dB	-25.19

Example 4: bandwidth 5:1

Input parameters:

- SLL optimization: $r = 1.667$.

Figure 4.28 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 4.22.

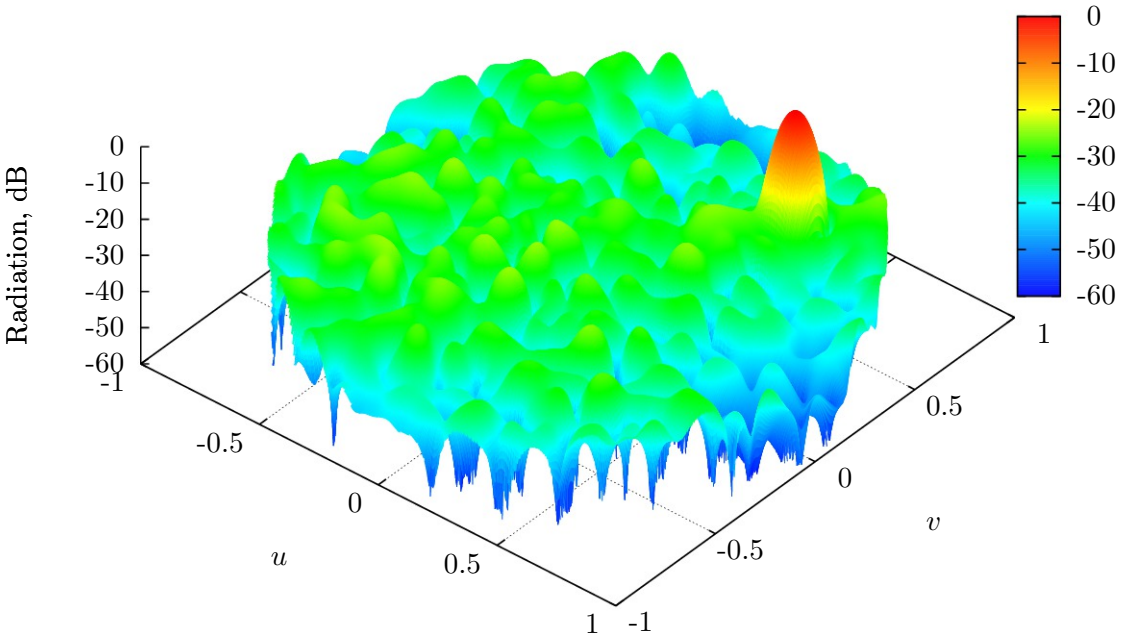


Figure 4.28: Radiation pattern in the fourth example at $r = 1.667$

Table 4.22: Output data of the fourth example

Parameter	Value
Number of polyominoes α	359
Fullness of the structure A, %	99.90
Sidelobe level γ at $r = 1.667$, dB	-24.10

Example 5: bandwidth 7:1

Input parameters:

- SLL optimization: $r = 1.750$.

Figure 4.29 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 4.23.

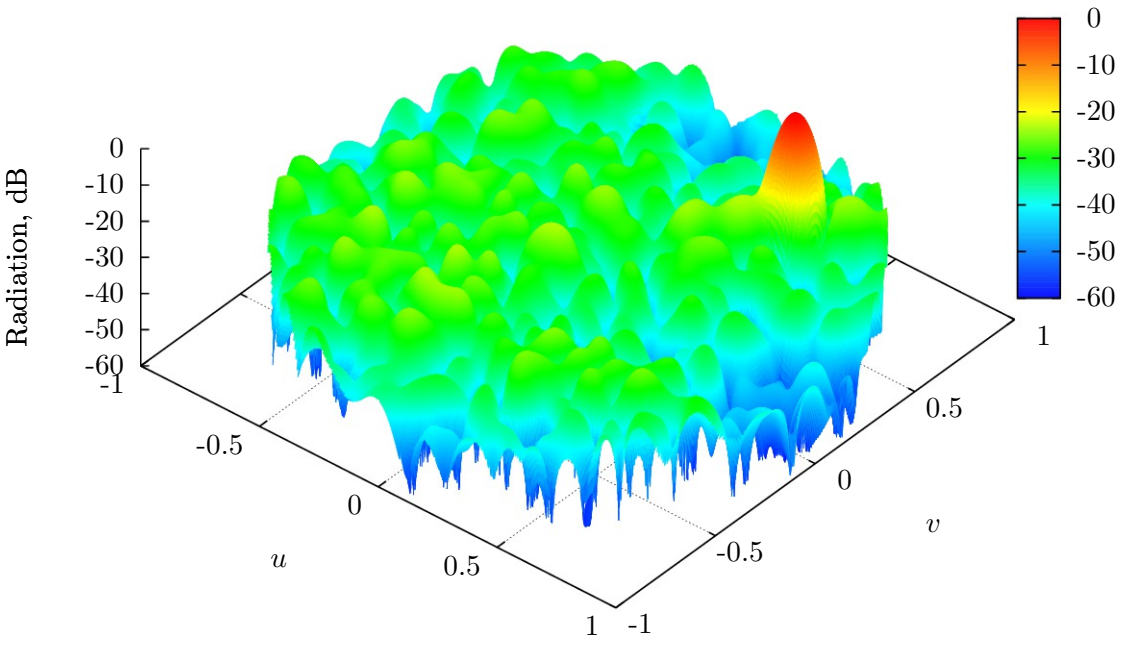


Figure 4.29: Radiation pattern in the fifth example at $r = 1.750$

Table 4.23: Output data of the fifth example

Parameter	Value
Number of polyominoes α	364
Fullness of the structure A, %	99.61
Sidelobe level γ at $r = 1.750$, dB	-23.03

Example 6: bandwidth 10:1

Input parameters:

- SLL optimization: $r = 1.818$.

Figure 4.30 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 4.24.

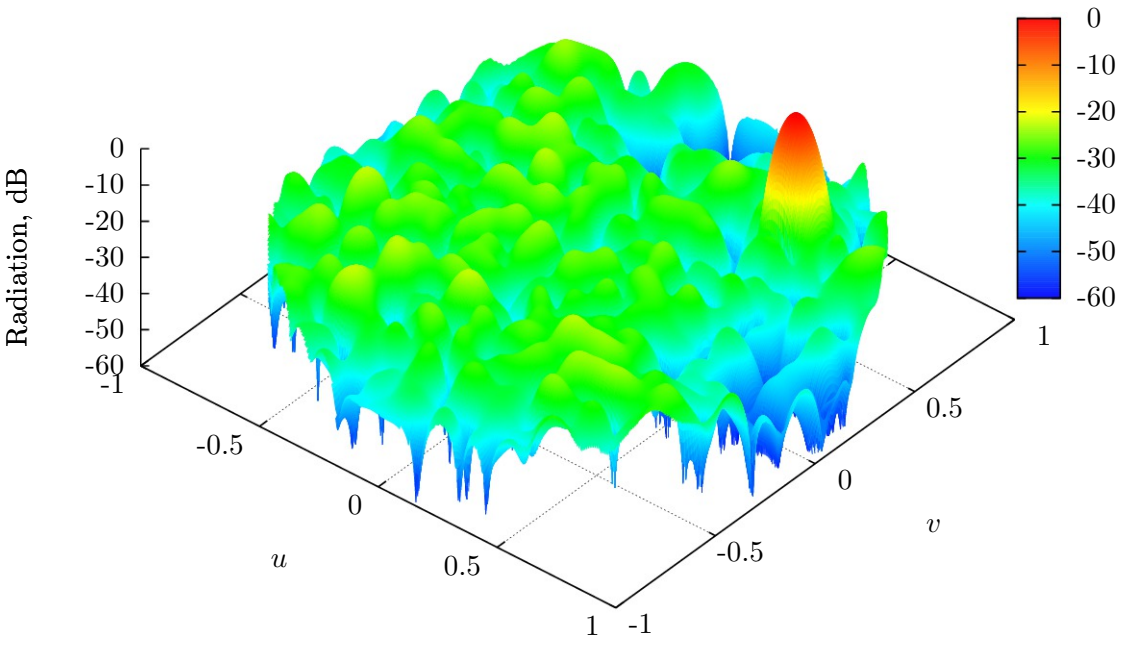


Figure 4.30: Radiation pattern in the sixth example at $r = 1.818$

Table 4.24: Output data of the sixth example

Parameter	Value
Number of polyominoes α	363
Fullness of the structure A, %	99.90
Sidelobe level γ at $r = 1.818$, dB	-22.24

Example 7: bandwidth 15:1

Input parameters:

- SLL optimization: $r = 1.875$.

Figure 4.31 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 4.25.

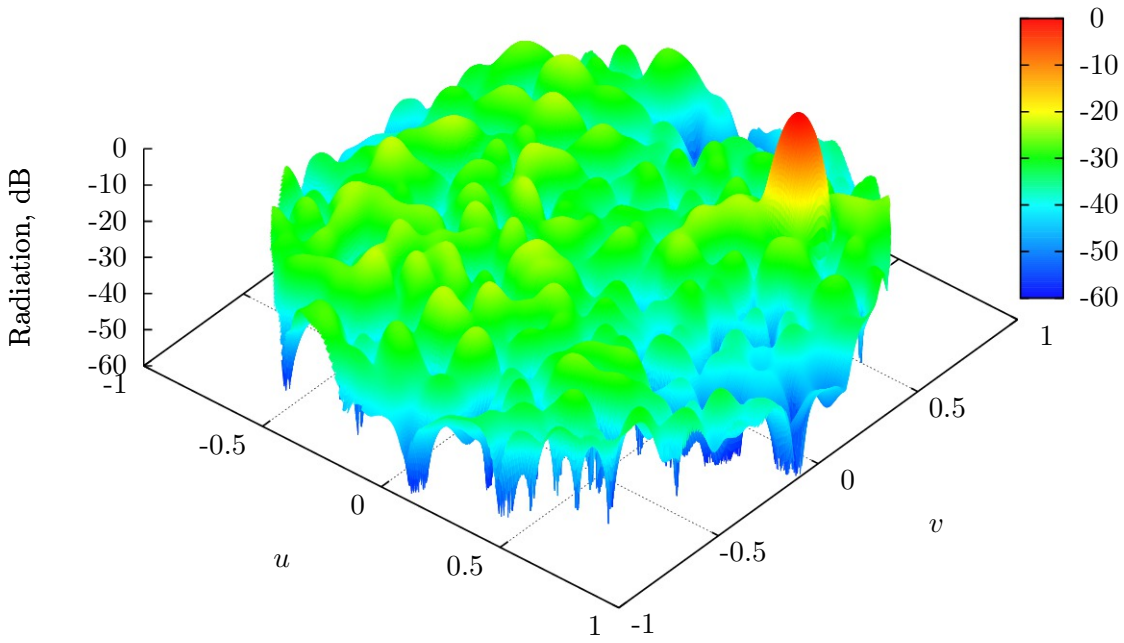


Figure 4.31: Radiation pattern in the sixth example at $r = 1.875$

Table 4.25: Output data of the sixth example

Parameter	Value
Number of polyominoes α	366
Fullness of the structure A, %	100.00
Sidelobe level γ at $r = 1.875$, dB	-22.27

Example 8: bandwidth 20:1

Input parameters:

- SLL optimization: $r = 1.905$.

Figure 4.32 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 4.26.

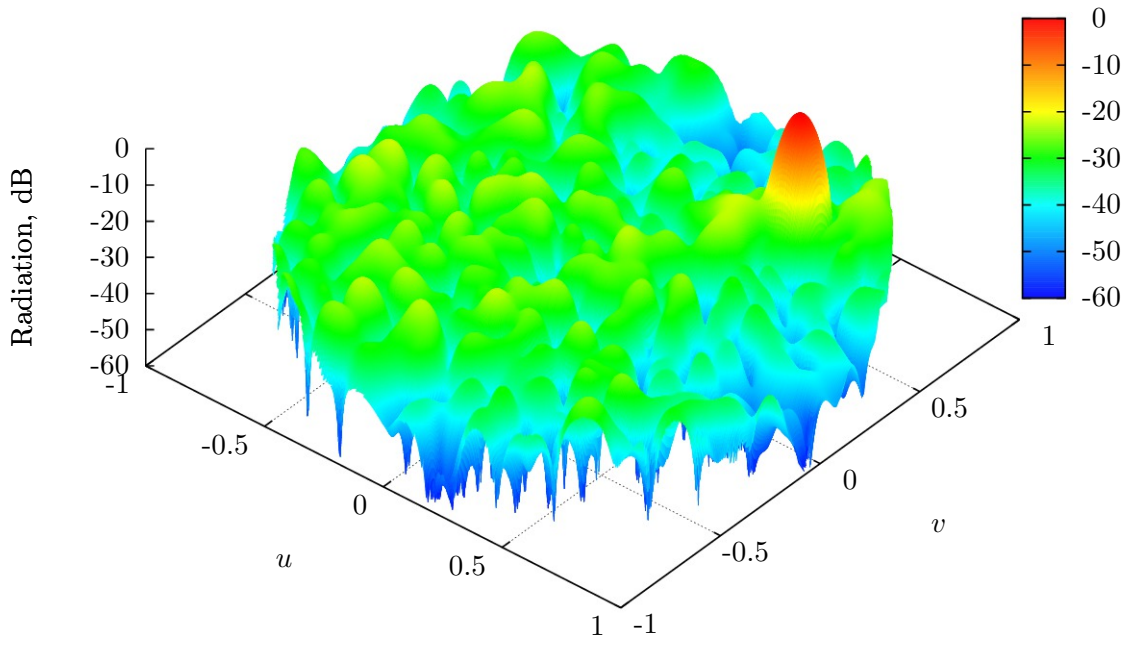


Figure 4.32: Radiation pattern in the sixth example at $r = 1.905$

Table 4.26: Output data of the sixth example

Parameter	Value
Number of polyominoes α	362
Fullness of the structure A, %	99.71
Sidelobe level γ at $r = 1.905$, dB	-22.01

Next, figure 4.33 presents the graph of sidelobe levels for 32×32 structures, averaged over seven different seeds $s = 10, 20, \ldots, 70$. The graph demonstrates the upper limit of SLL to which it tends at wide bands.

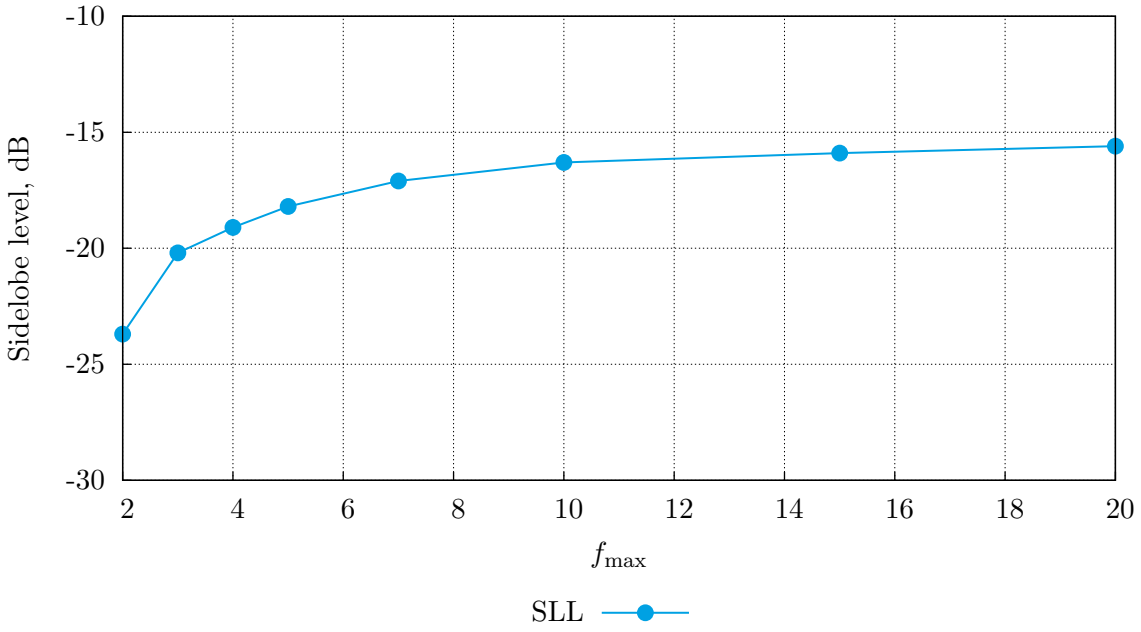


Figure 4.33: SLL at different bandwidths

4.6 Chapter 4 conclusions

1. Examples of radiation patterns and parameters of antenna arrays were listed, which were obtained by the Gwee—Lim algorithm. The analysis shows that sidelobe level at band $r = 1.818$ is too high for structures tiled with L-octomino.

2. Examples of radiation patterns and parameters of antenna arrays were listed, which were obtained by the snowball algorithm. The analysis shows that the snowball algorithm can synthesize structures with better sidelobe level suppression in the radiation pattern.

3. There was done the final comparative analysis of the Gwee—Lim algorithm and the "Snowball" algorithm. The analysis from the point of view of radiation pattern forming and level of sidelobes has shown that the developed algorithm is able to synthesize antenna arrays with characteristics, outperforming existing analogues, including ones by the Gwee—Lim algorithm.

4. Analysis of steering and bandwidth capacities of the resulting structures has been done. Steering analysis shows, that 32×32 structures are able to scan up to $\pm 30^\circ$ from broad sight at a bandwidth of 10:1 and up to $\pm 75^\circ$ at a bandwidth of 2:1.

5. The bandwidth analysis has shown that the upper limit for SLL in a 32×32 structures tiled with L-trominoes is -15 dB.

Chapter 5

Multi-beam features of phased antenna arrays

In this chapter we unveil and make preliminary analysis of another important feature of phased antenna arrays: multi-beam radiation patterns. It means that the array can have more than one main beam in its radiation pattern. Those beams can be controlled electronically and independently, making possible two spatially separated channels or two scanning lobes for radar.

Although it is possible to achieve this feature with structures tiled with one type of subarrays, we will use two shapes of subarrays simultaneously as a trade-off between number of subarrays and performance. In this case subarrays of one shape will act to form the beam in one direction.

Figure 5.1 shows such a structure. All L-octominoes are steered to angle $(45°; 45°)$ while all L-tetrominoes as steered to $(-135°; 45°)$. The corresponding radiation pattern is shown in figure 5.2.

This chapter includes analysis of sidelobe level for different angles between the lobes. The final part is dedicated to providing examples of power management among the beams. In the first case the control of lobe power is assigned by the genetic algorithm, and in the second these lobes are rigidly aligned.

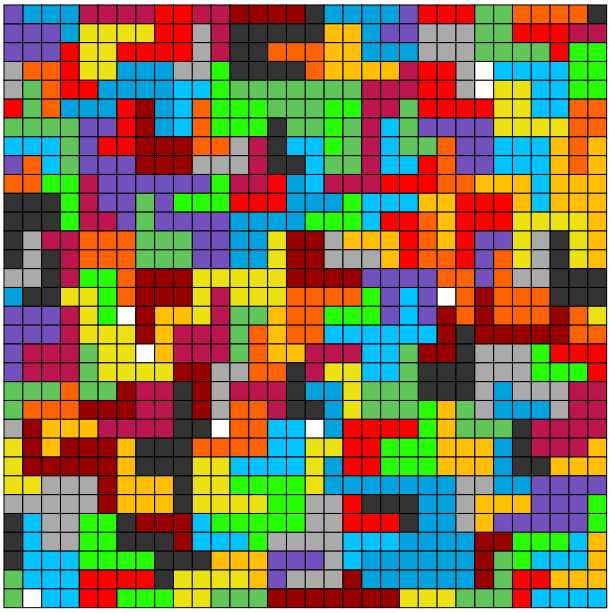


Figure 5.1: Structure of the array in the first example

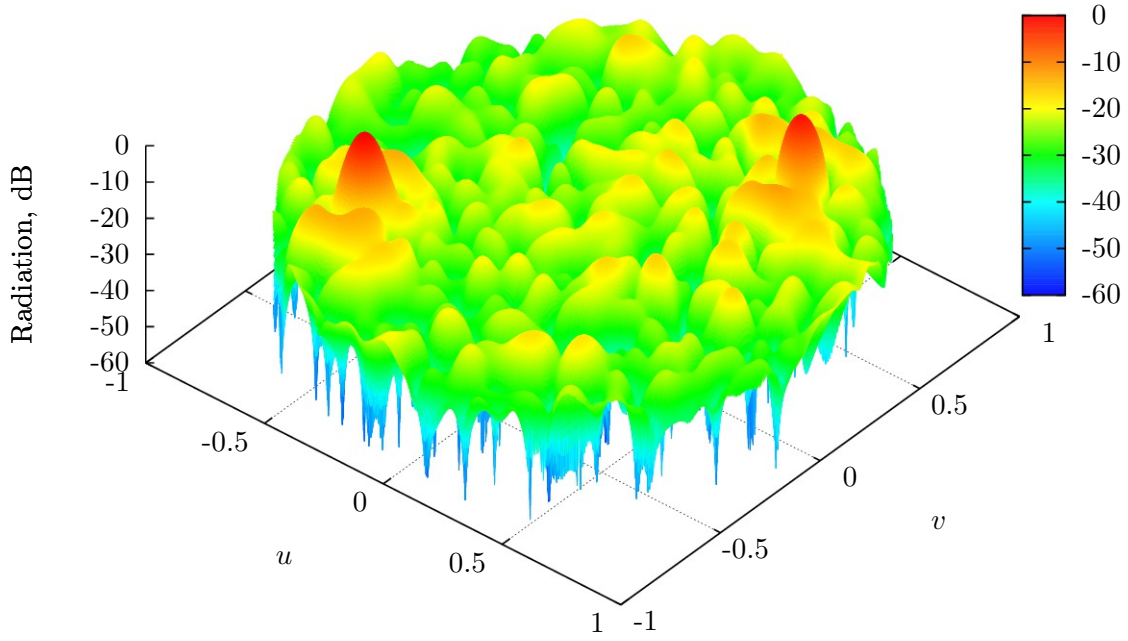


Figure 5.2: Radiation pattern for structure in figure 5.1 at $r = 1.3$

5.1 Analysis of SLL for different angles between two beams

This paragraph presents examples and analysis of radiation patterns of 32×32 arrays, obtained by Snowball, having two main beams pointed at different angles. In total there are five examples. The following settings were applied:

- polyomino type: L-octomino and L-tetromino;
- number of iterations: $K = 50$;
- population size: $P = 20$;
- seed: $s = 37$.

Calibrated values of p_c, p_m and p_{bm} were used. Figure 5.3 shows the graph of fitness function and its components in the seventh example.

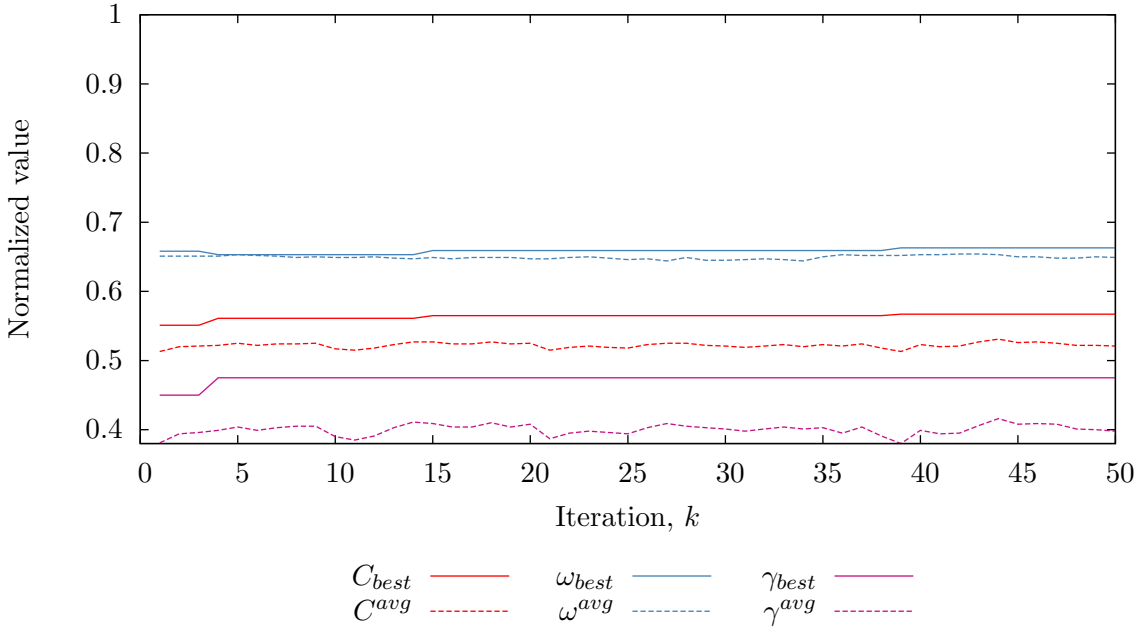


Figure 5.3: Fitness function in the first example

Example 1: angle 30°

Input parameters:

- beam 1 direction: $(0.25; 0.06)$;
- beam 2 direction: $(-0.25; -0.06)$.

Figure 5.4 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.1.

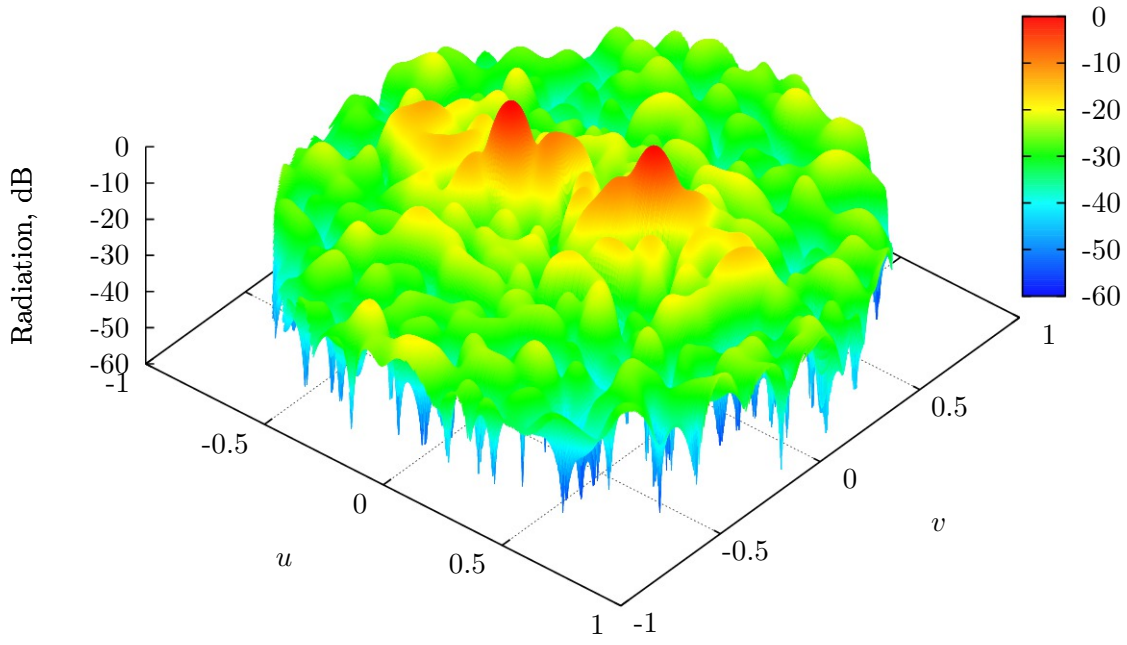


Figure 5.4: Radiation pattern in the first example at $r = 1.300$

Table 5.1: Output data of the first example

Parameter	Value
Number of polyominoes α	212
Fullness of the structure A, %	98.73
Sidelobe level γ at $r = 1.300$, dB	-19.05
Beam power difference at $r = 1.300$, dB	3.25

Example 2: angle 60°

Input parameters:

- beam 1 direction: $(0.43; 0.25)$;
- beam 2 direction: $(-0.43; -0.25)$.

Figure 5.5 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.2.

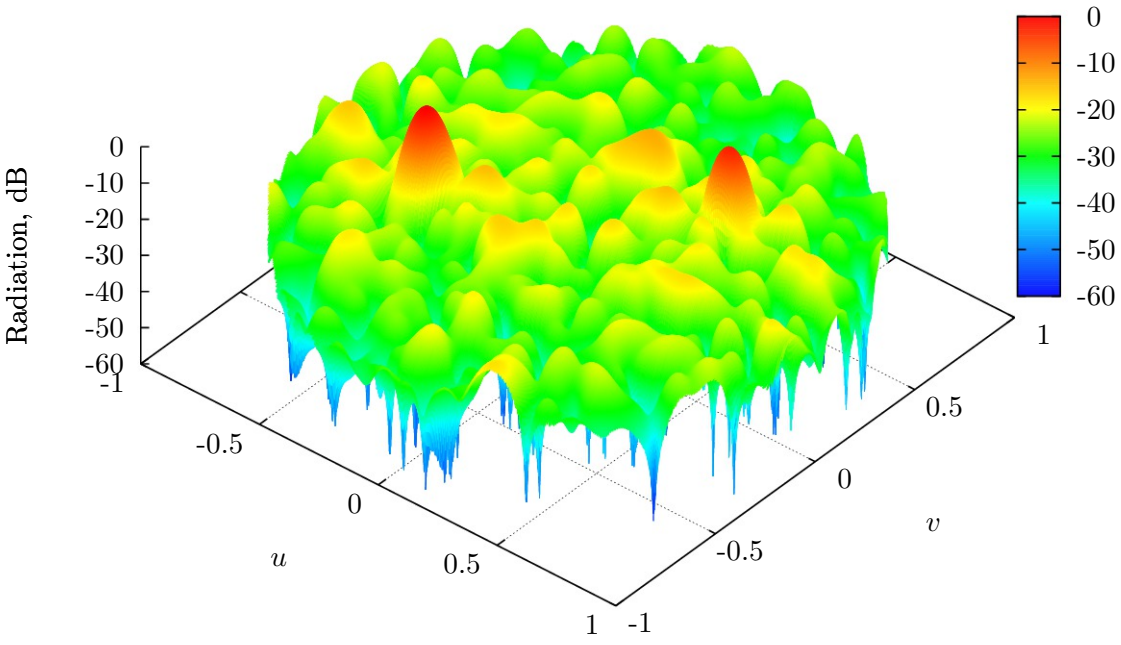


Figure 5.5: Radiation pattern in the second example at $r = 1.300$

Table 5.2: Output data of the second example

Parameter	Value
Number of polyominoes α	210
Fullness of the structure A, %	98.14
Sidelobe level γ at $r = 1.300$, dB	-17.56
Beam power difference at $r = 1.300$, dB	0.00

Example 3: angle 90°

Input parameters:

- beam 1 direction: $(0.5; 0.5)$;
- beam 2 direction: $(-0.5; -0.5)$.

Figure 5.6 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.3.

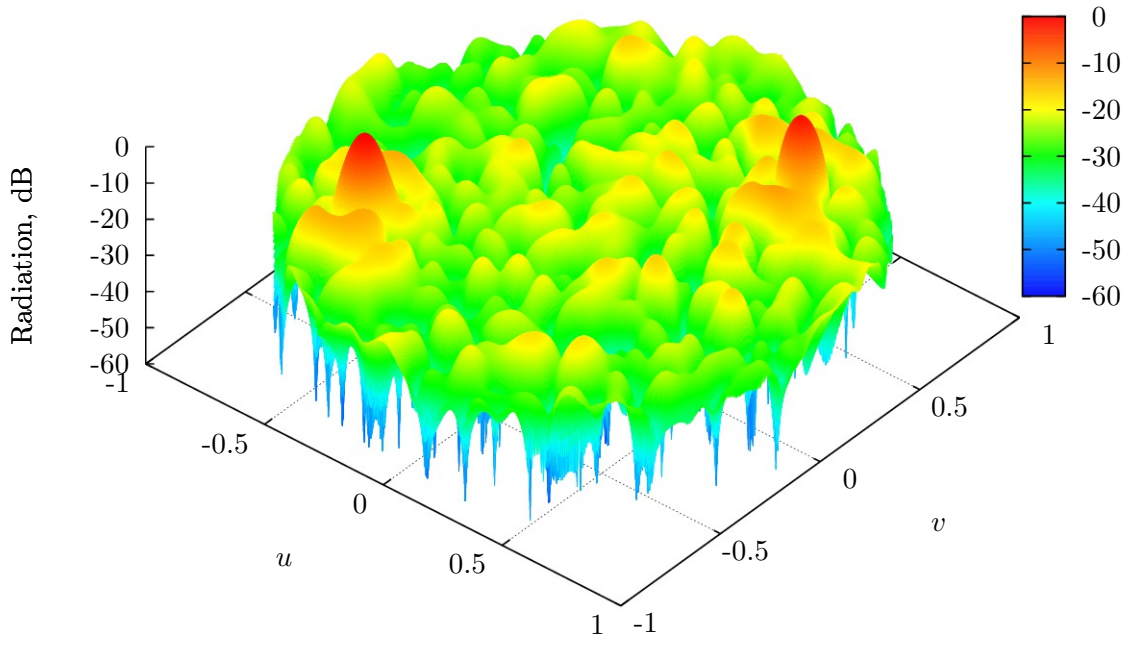


Figure 5.6: Radiation pattern in the third example at $r = 1.300$

Table 5.3: Output data of the third example

Parameter	Value
Number of polyominoes α	208
Fullness of the structure A, %	99.22
Sidelobe level γ at $r = 1.300$, dB	-16.22
Beam power difference at $r = 1.300$, dB	0.00

Example 4: angle 120°

Input parameters:

- beam 1 direction: $(0.433; 0.75)$;
- beam 2 direction: $(-0.433; -0.75)$.

Figure 5.7 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.4.

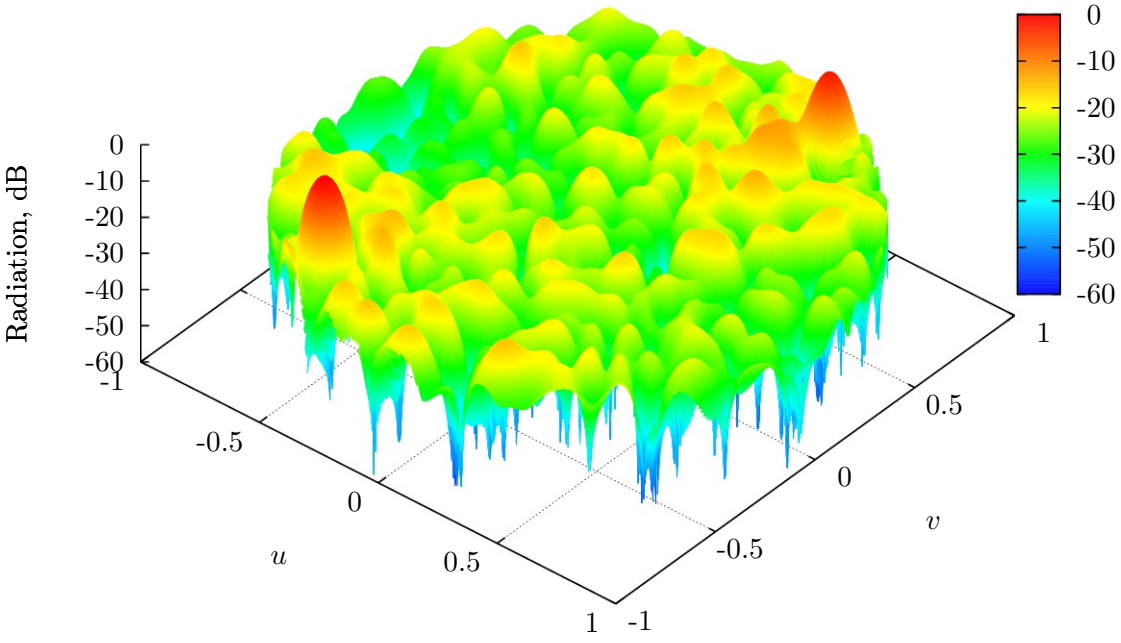


Figure 5.7: Radiation pattern in the fourth example at $r = 1.300$

Table 5.4: Output data of the fourth example

Parameter	Value
Number of polyominoes α	212
Fullness of the structure A, %	99.12
Sidelobe level γ at $r = 1.300$, dB	-15.38
Beam power difference at $r = 1.300$, dB	0.00

Example 5: angle 150°

Input parameters:

- beam 1 direction: $(0.251; 0.93)$;
- beam 2 direction: $(-0.251; -0.93)$.

Figure 5.8 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.5.

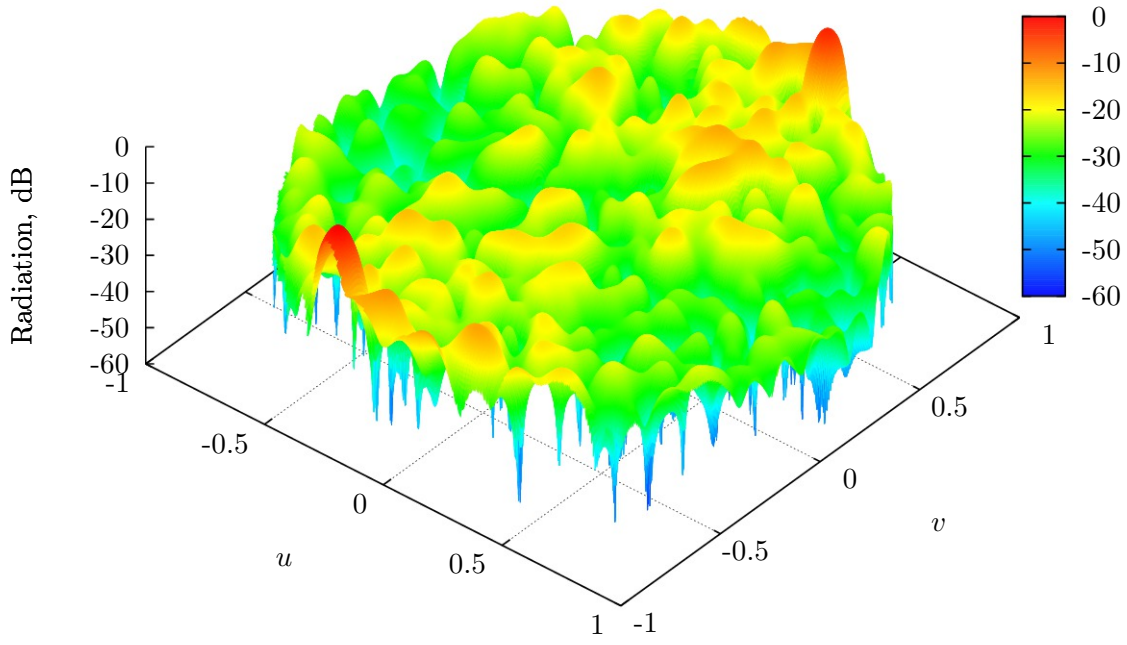


Figure 5.8: Radiation pattern in the fifth example at $r = 1.300$

Table 5.5: Output data of the fifth example

Parameter	Value
Number of polyominoes α	207
Fullness of the structure A, %	98.34
Sidelobe level γ at $r = 1.300$, dB	-15.03
Beam power difference at $r = 1.300$, dB	0.00

Figure 5.9 presents the graph of sidelobe levels and beam power differences for 32×32 structures, averaged over seven different seeds $s = 10, 20, \ldots, 70$. The graph demonstrates that power difference is mainly zero, while SLL rises too high for wide angles.

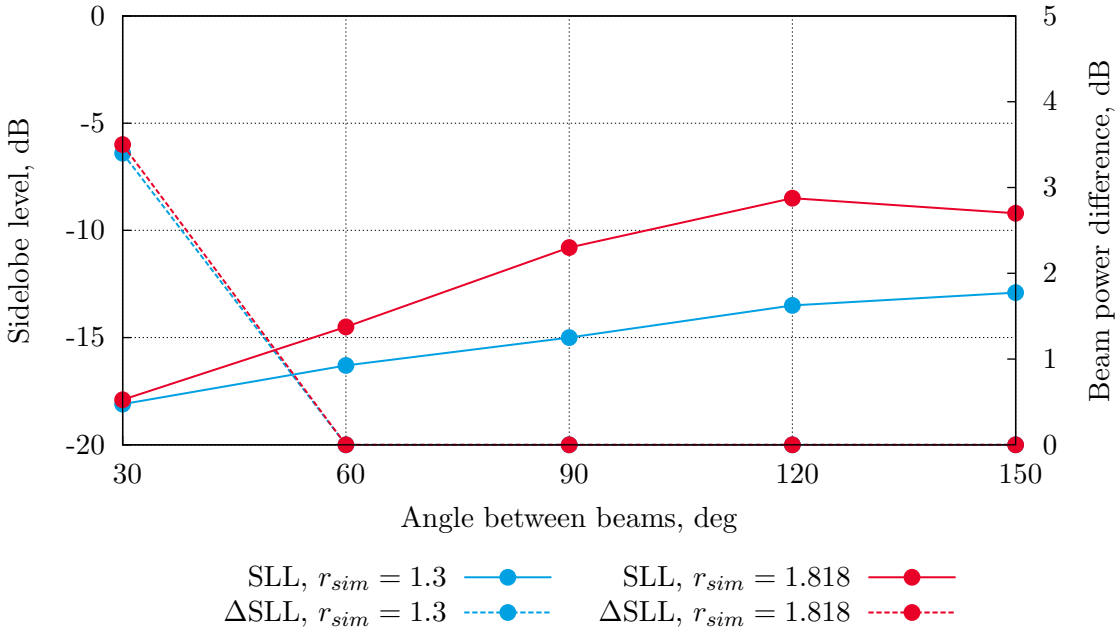


Figure 5.9: SLL and beam power difference for various angles

5.2 Application of the developed algorithm to two beams forming

As it is known, the amount of radiated power is proportional to the number of emitters. In the first set of six examples algorithm decides itself how many polyominoes of each shape should be there in the structure. So, it can be seen that the main lobe levels slightly differ.

In the second six examples it is hard-coded in the algorithm that the number of subarrays in shape of L-tetromino must be twice the number of subarrays in the shape of L-octomino. This is done for having equal areas

emitting in every direction. Accordingly, the lobes have the same power.

The following parameters were used:

- polyomino type (1 — 6): L-octomino and L-tromino;
- polyomino type (7 — 12): L-octomino and L-tetromino;
- number of iterations: $K = 50$;
- population size: $P = 10$;
- elitism: on;
- seed: $s = 37$;
- SLL optimization: none, $r = 1.3$, $r = 1.818$;
- steering: $(45°; 45°)$ and $(135°; 45°)$.

Figure 5.10 shows the graph of the fitness function in the fourth example.

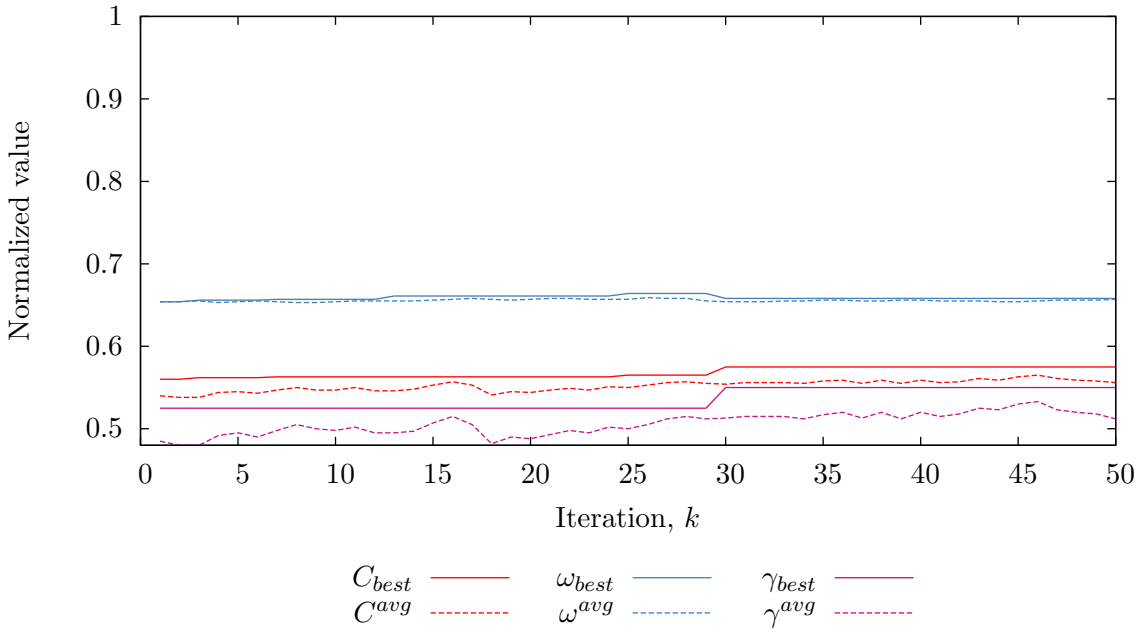


Figure 5.10: Fitness function in the fourth example

Example 1: Structure 32 × 32, L-octomino and L-tromino

Input parameters:

- structure size: $M = N = 32$;
- SLL optimization: no.

Figure 5.11 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.6.

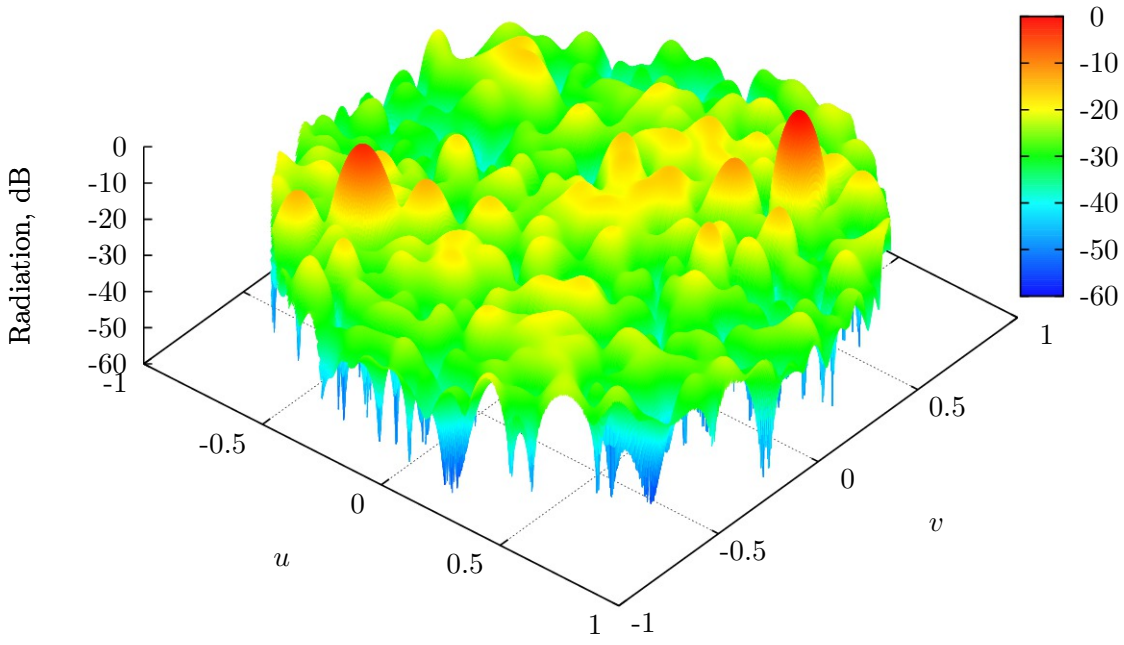


Figure 5.11: Radiation pattern in the first example at $r = 1.3$

Table 5.6: Output data of the first example

Parameter	Value
Number of polyominoes α	225
Fullness of the structure A, %	99.02
Sidelobe level γ at $r = 1.300$, dB	-12.63
Sidelobe level γ at $r = 1.818$, dB	-9.84

Example 2: Structure 64 × 64, L-octomino and L-tromino

Input parameters:

- structure size: $M = N = 64$;
- SLL optimization: no.

Figure 5.12 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.7.

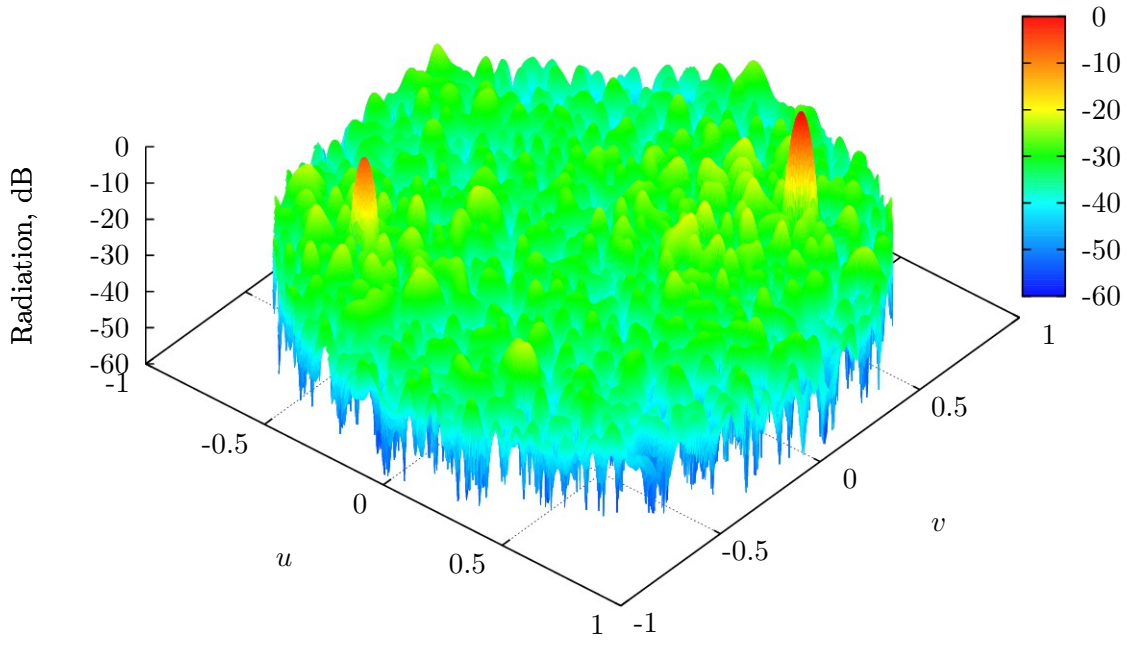


Figure 5.12: Radiation pattern in the second example at $r = 1.3$

Table 5.7: Output data of the second example

Parameter	Value
Number of polyominoes α	806
Fullness of the structure A, %	99.17
Sidelobe level γ at $r = 1.300$, dB	-20.79
Sidelobe level γ at $r = 1.818$, dB	-15.68

Example 3: Structure 32 × 32, L-octomino and L-tromino

Input parameters:

- structure size: $M = N = 32$;
- SLL optimization: $r = 1.3$.

Figure 5.13 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.8.

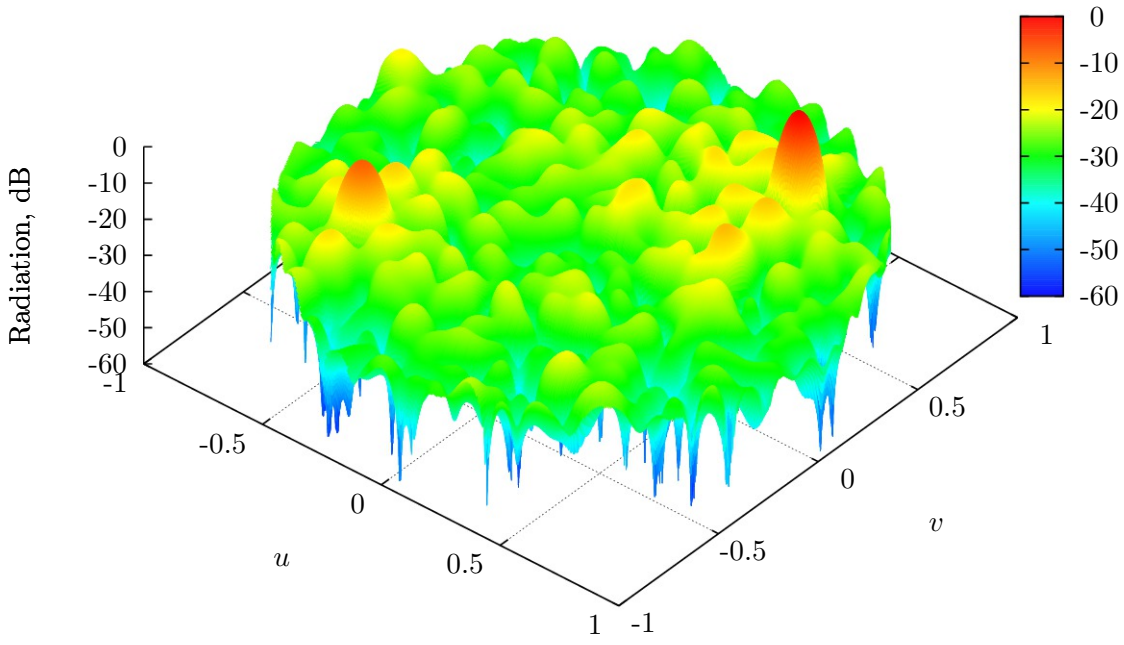


Figure 5.13: Radiation pattern in the third example at $r = 1.3$

Table 5.8: Output data of the third example

Parameter	Value
Number of polyominoes α	198
Fullness of the structure A, %	98.93
Sidelobe level γ at $r = 1.300$, dB	-18.02
Sidelobe level γ at $r = 1.818$, dB	-10.79

Example 4: Structure 64 × 64, L-octomino and L-tromino

Input parameters:

- structure size: $M = N = 64$;
- SLL optimization: $r = 1.3$.

Figure 5.14 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.9.

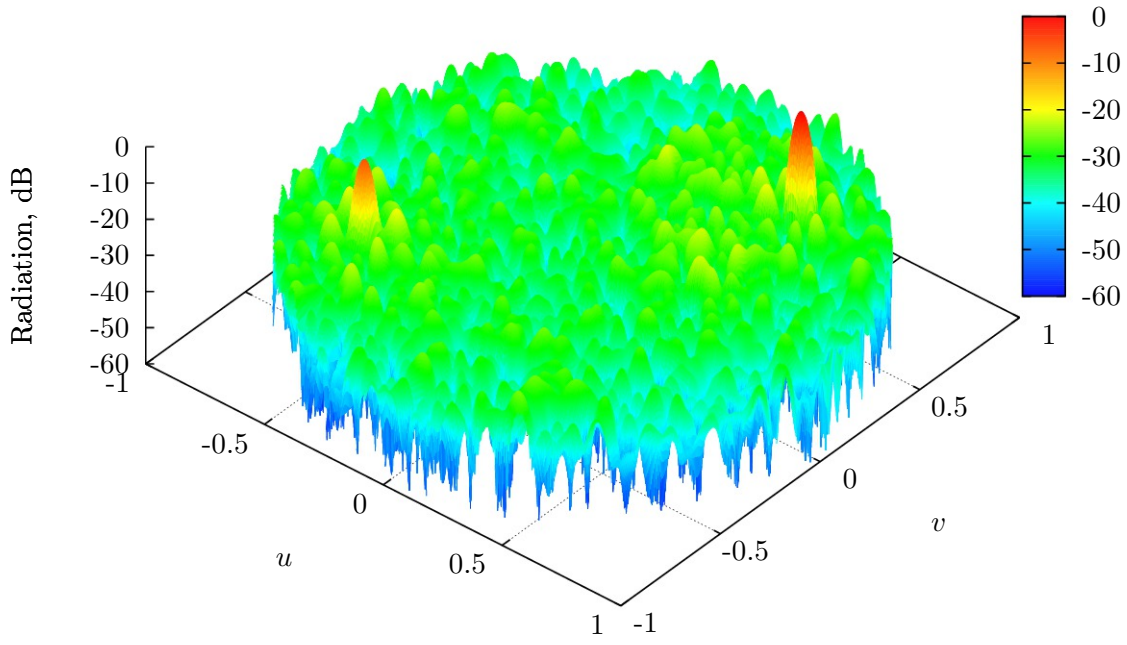


Figure 5.14: Radiation pattern in the fourth example at $r = 1.3$

Table 5.9: Output data of the fourth example

Parameter	Value
Number of polyominoes α	793
Fullness of the structure A, %	98.97
Sidelobe level γ at $r = 1.300$, dB	−21.06
Sidelobe level γ at $r = 1.818$, dB	−15.02

Example 5: Structure 32 × 32, L-octomino and L-tromino

Input parameters:

- structure size: $M = N = 32$;
- SLL optimization: $r = 1.818$.

Figure 5.15 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.10.

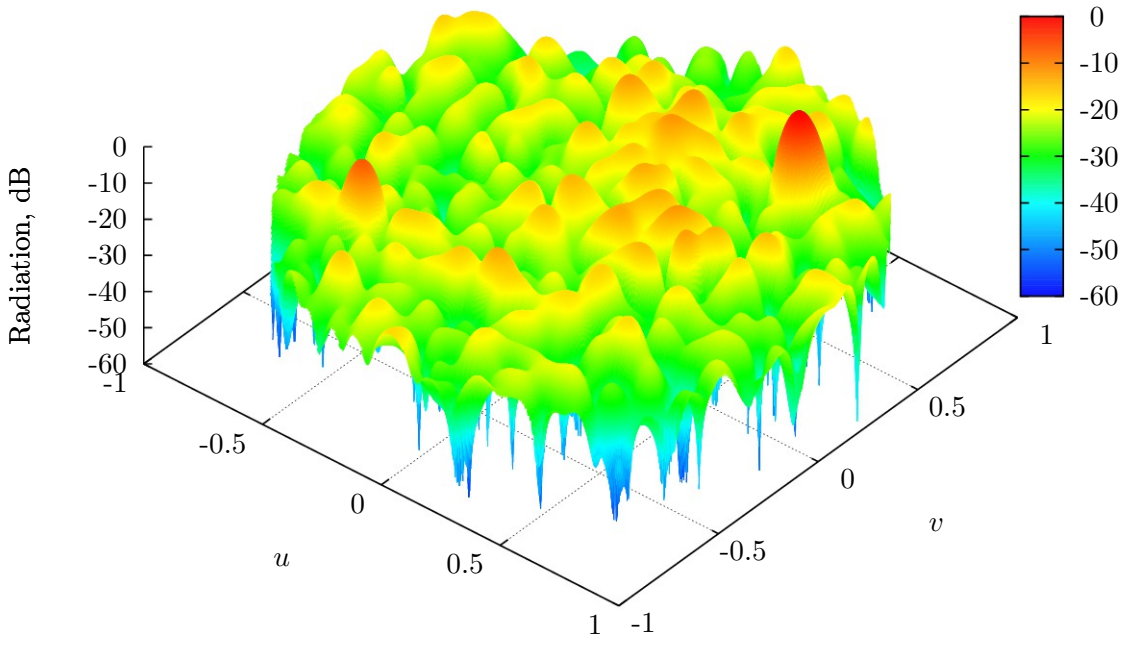


Figure 5.15: Radiation pattern in the fifth example at $r = 1.818$

Table 5.10: Output data of the fifth example

Parameter	Value
Number of polyominoes α	199
Fullness of the structure A, %	98.14
Sidelobe level γ at $r = 1.300$, dB	-17.78
Sidelobe level γ at $r = 1.818$, dB	-13.25

Example 6: Structure 64 × 64, L-octomino and L-tromino

Input parameters:

- structure size: $M = N = 64$;
- SLL optimization: $r = 1.818$.

Figure 5.16 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.11.

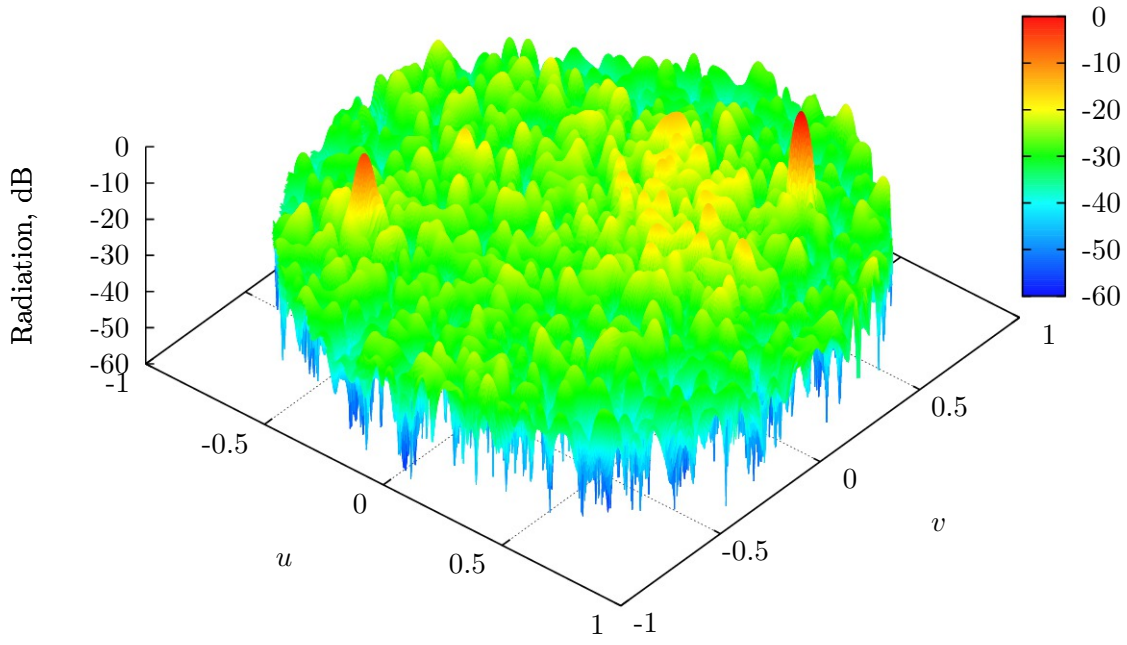


Figure 5.16: Radiation pattern in the sixth example at $r = 1.818$

Table 5.11: Output data of the sixth example

Parameter	Value
Number of polyominoes α	784
Fullness of the structure A, %	99.07
Sidelobe level γ at $r = 1.300$, dB	−20.48
Sidelobe level γ at $r = 1.818$, dB	−16.09

Example 7: Structure 32 × 32, L-octomino and L-tetromino

Input parameters:

- structure size: $M = N = 32$;
- SLL optimization: no.

Figure 5.17 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.12.

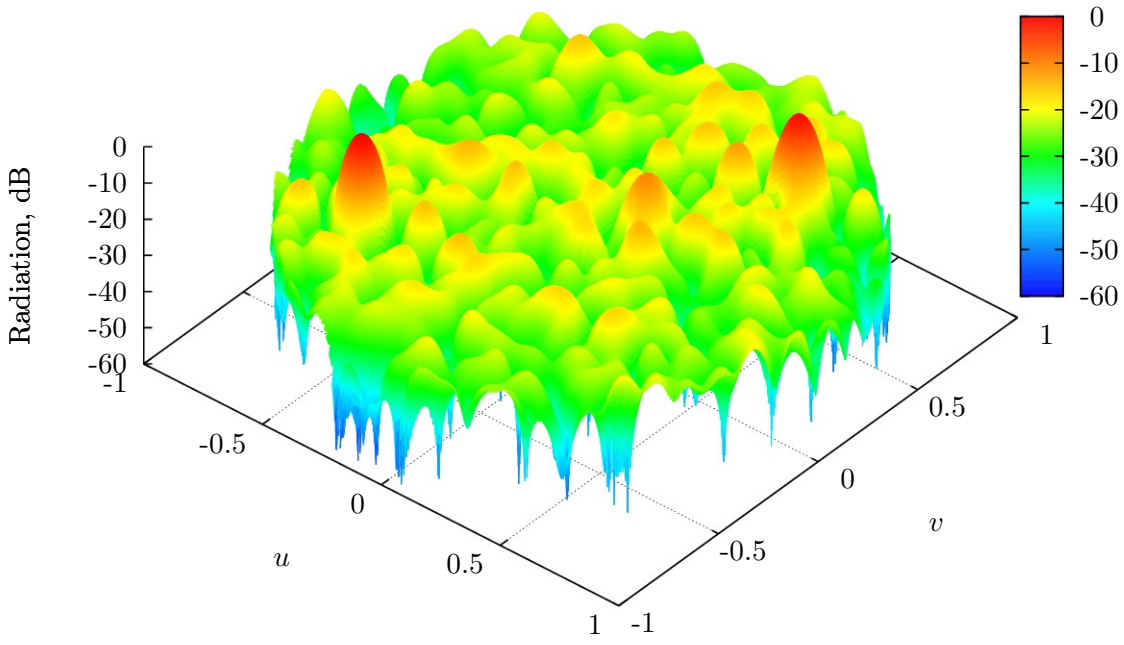


Figure 5.17: Radiation pattern in the seventh example at $r = 1.3$

Table 5.12: Output data of the seventh example

Parameter	Value
Number of polyominoes α	213
Fullness of the structure A, %	99.02
Sidelobe level γ at $r = 1.300$, dB	-10.78
Sidelobe level γ at $r = 1.818$, dB	-5.37

Example 8: Structure 64 × 64, L-octomino and L-tetromino

Input parameters:

- structure size: $M = N = 64$;
- SLL optimization: no.

Figure 5.18 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.13.

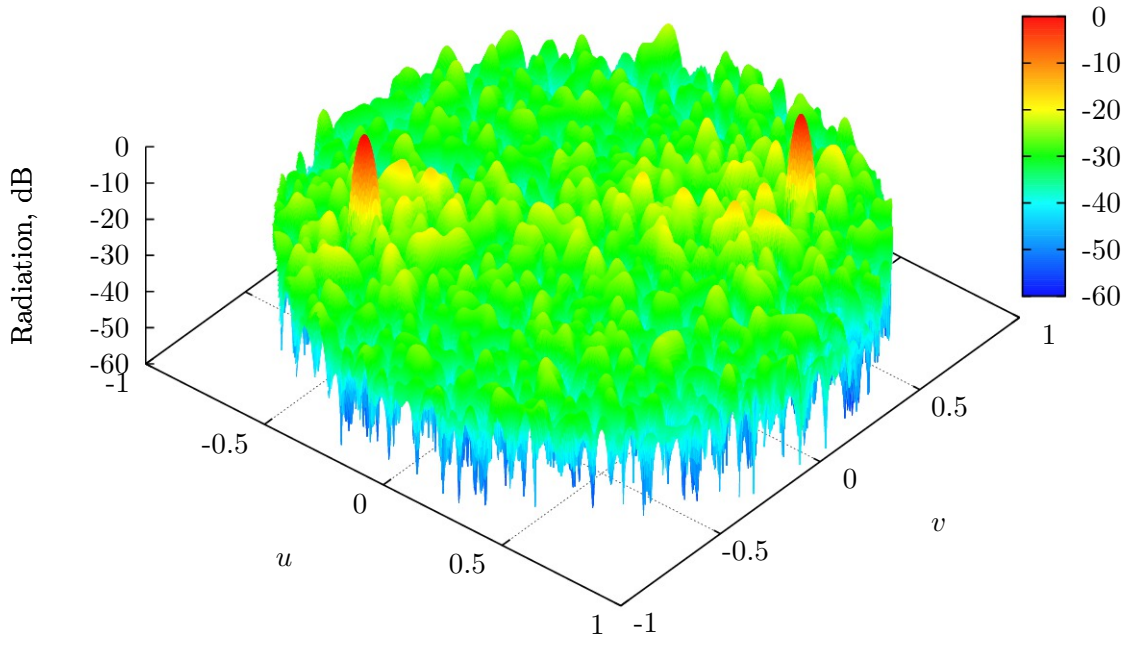


Figure 5.18: Radiation pattern in the eighth example at $r = 1.3$

Table 5.13: Output data of the eighth example

Parameter	Value
Number of polyominoes α	794
Fullness of the structure A, %	99.22
Sidelobe level γ at $r = 1.300$, dB	−18.38
Sidelobe level γ at $r = 1.818$, dB	−12.85

Example 9: Structure 32 × 32, L-octomino and L-tetromino

Input parameters:

- structure size: $M = N = 32$;
- SLL optimization: $r = 1.3$.

Figure 5.19 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.14.

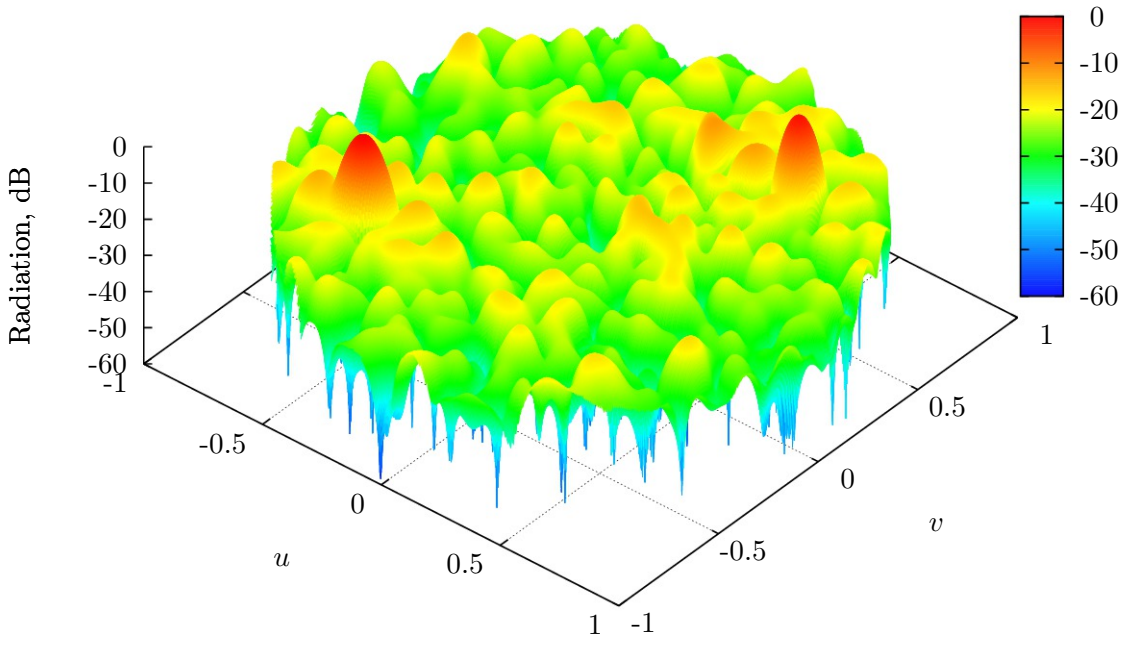


Figure 5.19: Radiation pattern in the ninth example at $r = 1.3$

Table 5.14: Output data of the ninth example

Parameter	Value
Number of polyominoes α	207
Fullness of the structure A, %	98.54
Sidelobe level γ at $r = 1.300$, dB	-15.26
Sidelobe level γ at $r = 1.818$, dB	-8.87

Example 10: Structure 64 × 64, L-octomino and L-tetromino

Input parameters:

- structure size: $M = N = 64$;
- SLL optimization: $r = 1.3$.

Figure 5.20 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.15.

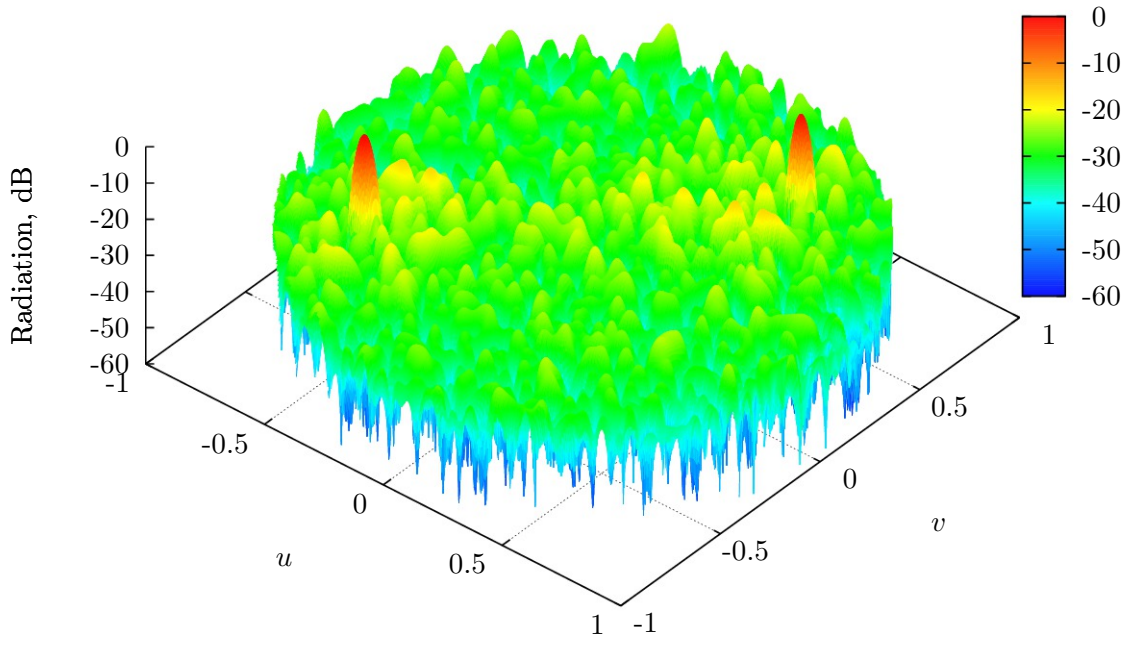


Figure 5.20: Radiation pattern in the tenth example at $r = 1.3$

Table 5.15: Output data of the tenth example

Parameter	Value
Number of polyominoes α	794
Fullness of the structure A, %	99.22
Sidelobe level γ at $r = 1.300$, dB	−18.38
Sidelobe level γ at $r = 1.818$, dB	−12.85

Example 11: Structure 32 × 32, L-octomino and L-tetromino

Input parameters:

- structure size: $M = N = 32$;
- SLL optimization: $r = 1.818$.

Figure 5.21 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.16.

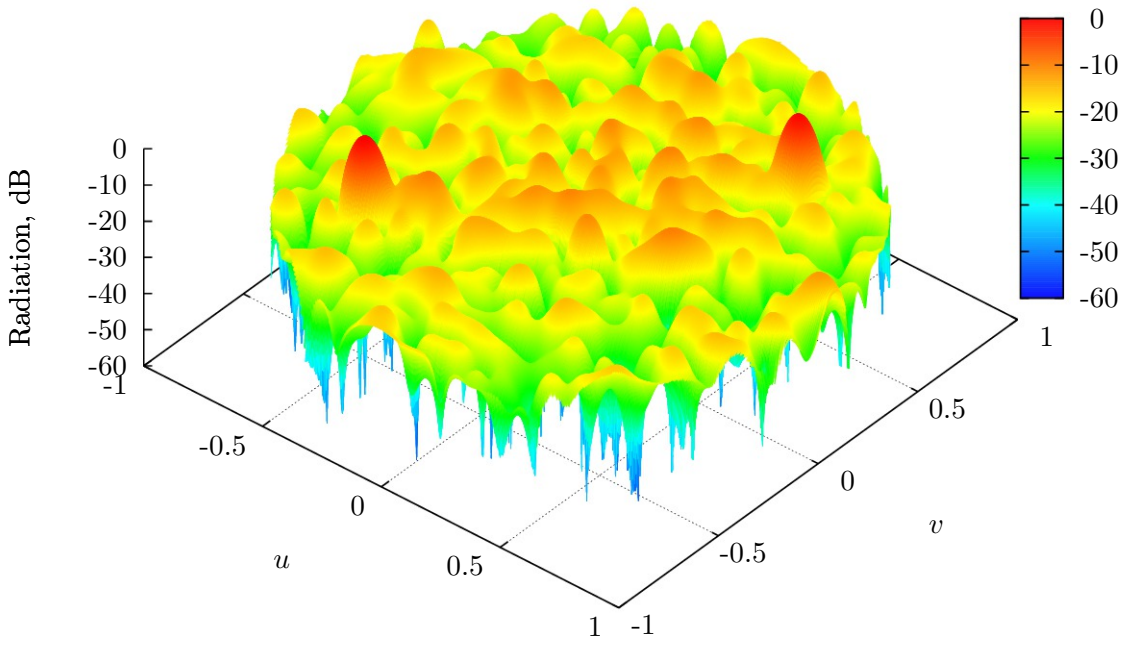


Figure 5.21: Radiation pattern in the eleventh example at $r = 1.818$

Table 5.16: Output data of the eleventh example

Parameter	Value
Number of polyominoes α	206
Fullness of the structure A, %	97.85
Sidelobe level γ at $r = 1.300$, dB	-13.37
Sidelobe level γ at $r = 1.818$, dB	-10.17

Example 12: Structure 64 × 64, L-octomino and L-tetromino

Input parameters:

- structure size: $M = N = 64$;
- SLL optimization: $r = 1.818$.

Figure 5.22 shows the radiation pattern obtained by SA with those parameters. Numerical results are provided in table 5.17.

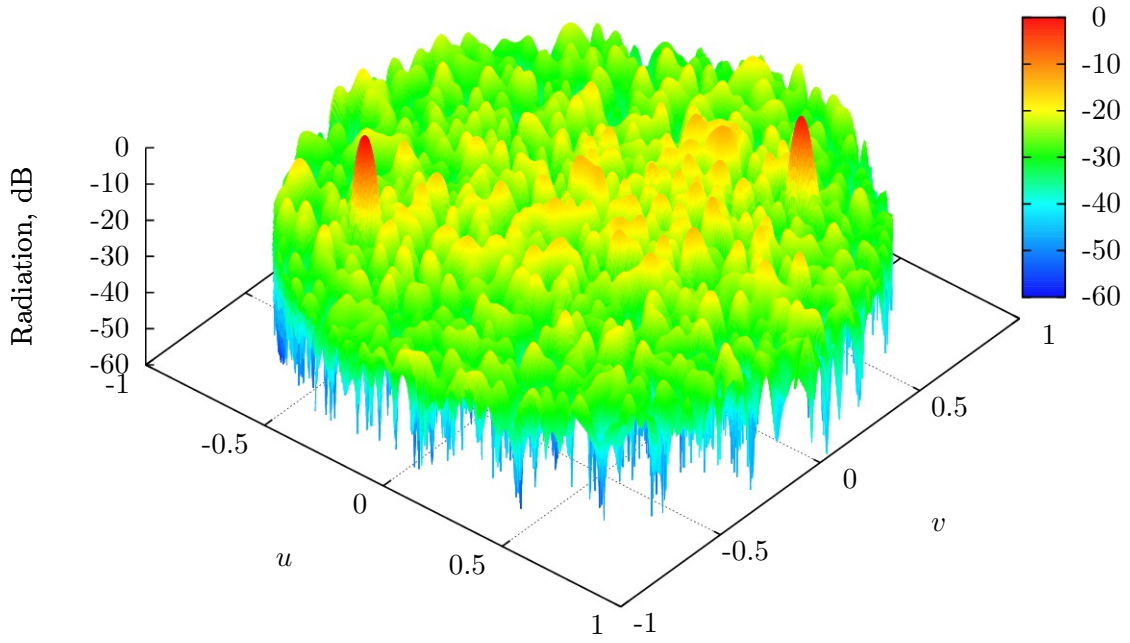


Figure 5.22: Radiation pattern in the twelfth example at $r = 1.818$

Table 5.17: Output data of the twelfth example

Parameter	Value
Number of polyominoes α	793
Fullness of the structure A, %	99.05
Sidelobe level γ at $r = 1.300$, dB	−18.65
Sidelobe level γ at $r = 1.818$, dB	−14.64

5.3 Chapter 5 conclusions

1. Examples of antenna arrays synthesis is shown, which are able to form two beams in radiation pattern for receiving and transmitting signals in two directions simultaneously.

2. Analysis of angle between the two main lobes was done. It showed, obviously, that SLL grows with the angle and reaches inappropriately high values.

3. Examples demonstrate ability of the algorithm to control the power radiated in each direction, considering structure size and polyomino type. Algorithm is able to make the beams equal in power. Otherwise, the difference due to SLL suppression may reach 8 dB.

Chapter 6

Conclusions

1. The mathematical model of the antenna array structure was developed, that describes geometrical and electrodynamic connections between the elements. The model is based on the matrix representation of polyomino orientations, differs by using the modified array factor and allows describing arrays tiled with polyomino-shaped subarrays.

2. The optimization method was developed, based on calculation of the integral of the autocorrelation function, different by using the Hilbert curve and allowing performing multicriteria optimization considering the irregularity of structures of polyominoes. Analysis of the method has shown that estimation of irregularity of a structure correlates with computed value of sidelobe level and can be used in the process of optimization of phased antenna arrays.

3. The "Snowball" algorithm for synthesis of structures of polyominoes was developed, based on the evolutionary principle, different by way of polyomino placement in the structure and producing structures with fullness up to 98–100%. Parameters of the algorithm have been calibrated.

4. The software was developed based on the proposed algorithm for solv-

ing the task of antenna array structure optimization. Software made it possible to conduct numerical simulations of structures of phased antenna arrays.

5. The results of simulations were shown, demonstrating efficiency of the proposed algorithm and structures of arrays obtained by it. Optimization led to suppression of sidelobe levels by more than 3 times in comparison with other researchers' results.

Bibliography

[1] G. Abdrakhmanova, R. Chirikov, and V. Bagmanov. Laguerre functions for pulse simulation in uwb communication systems. In *Proc. of 14th Int. Workshop "Problems of Technics and Technologies of Telecommunication"*, pages 87–91, Samara, Russia, 2013. PSUTI.

[2] G. I. Abdrakhmanova. Uwb antenna simulation based on optimization algorithms. *Modern problems of science and education*, (4), 2013.

[3] Buthainah Al-kazemi and Chilukuri K Mohan. Multi-phase generalization of the particle swarm optimization algorithm. In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 2, pages 1057–1062, 2002.

[4] Peter J. Angeline. Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. In *Evolutionary Programming VII*, pages 601–610. Springer, 1998.

[5] Peter J Angeline. Using selection to improve particle swarm optimization. In *Proceedings of IEEE International Conference on Evolutionary Computation*, volume 89, 1998.

[6] Constantine A. Balanis. *Antenna theory: analysis and design*. John Wiley & Sons, 2012.

[7] E. Bekele, R. Chirikov, M. Carlin, L. Manica, G. Oliveri, L. Poli, P. Rocca, and A. Massa. What's new on antenna synthesis @ eledia

research center. In *Atti XIX Riunione Nazionale di Elettromagnetismo (XIX RiNEm)*, pages 297–300, Rome, Italy, 2012. RiNEm.

[8] Daniel W. Boeringer and Douglas H. Werner. Particle swarm optimization versus genetic algorithms for phased array synthesis. *Antennas and Propagation, IEEE Transactions on*, 52(3):771–779, 2004.

[9] Andreas Bortfeldt. A genetic algorithm for the two-dimensional strip packing problem with rectangular pieces. *European Journal of Operational Research*, 172(3):814–837, 2006.

[10] Andreas Bortfeldt and Hermann Gehring. A hybrid genetic algorithm for the container loading problem. *European Journal of Operational Research*, 131(1):143–161, 2001.

[11] Robert C. Brigham, Richard M. Caron, Phyllis Z. Chinn, and Ralph P. Grimaldi. A tiling scheme for the fibonacci numbers. *Journal of Recreational Mathematics*, 28:10–16, 1997.

[12] A. V. Chekanin and V. A. Chekanin. Algorithms for effective rectangular packing problem solving. *Journal of computational mathematics and mathematical physics*, 53(10):1639–1648, 2013.

[13] I Chernyh. *Electrical devices modelling in MATLAB, SimPowerSystems and Simulink*. Litres, 2011.

[14] Phyllis Chinn, Ralph Grimaldi, and Silvia Heubach. Tiling with l's and squares. *Journal of Integer Sequences*, 10(2):3, 2007.

[15] R. Chirikov. Innovative tiling algorithm for the design of antenna arrays. In *Proc. of 15th Int. Workshop on Computer Science and Information Technology CSIT2013*, volume 3, pages 87–90, Vienna, Austria, 2013. USATU.

[16] R. Chirikov. Large phased antenna array design as an optimal packing problem. *Infocommunication Technologies: journal of PSUTI*, 12(3):77–82, 2013.

[17] R. Chirikov, G. Abdrakhmanova, and A. Abdrakhmanova. Fractal antennas design for uwb communication systems. In *Proc. of 7th All-Russian Winter School "Actual Problems of Science and Technics"*, pages 153–156, Ufa, Russia, 2012. USATU.

[18] R. Chirikov and P. Rocca. Genetic algorithm for advanced clustering in phased array design. In *Proc. of 13th Int. Workshop on Computer Science and Information Technology CSIT2011*, volume 2, pages 32–34, Garmisch-Partenkirchen, Germany, 2011. USATU.

[19] R. Chirikov and P. Rocca. Implementation of the gwee—lim tiling algorithm. In *Proc. of 14th Int. Workshop on Computer Science and Information Technology CSIT2012*, volume 3, pages 163–166, Hamburg, Germany, 2012. USATU.

[20] R. Chirikov, P. Rocca, and V. Bagmanov. Calibration of the genetic algorithm parameters for phased antenna array design. *Electrical and Data Processing Facilities and Systems: journal of UGUES*, 9(3):89–92, 2013.

[21] R. Chirikov, P. Rocca, V. Bagmanov, and G. Abdrakhmanova. Evolutionary algorithms for advanced clustering methods in phased array design. In *Proc. of 12th Int. Workshop "Problems of Technics and Technologies of Telecommunication"*, pages 263–264, Kazan, Russia, 2011. KSTU.

[22] R. Chirikov, P. Rocca, V. Bagmanov, and A. Sultanov. Phased antenna design algorithm for satellite communications. *Vestnik USATU: journal of USATU*, 17(4 (57)):159–166, 2013.

[23] R. Chirikov, P. Rocca, and E. Grakhova. Irregularity estimation of planar structures based on colour mixing. In *Proc. of Int. Conf. "Information Technologies for Intelligent Decision Making Support"*, pages 45–48, Ufa, Russia, 2012. USATU.

[24] R. Chirikov, P. Rocca, L. Manica, S. Santarelli, R. J. Mailloux, and A. Massa. Innovative ga-based strategy for polyomino tiling in phased array design. In *EuCAP-2013*, pages 2216–2219, Gothenburg, Sweden, 2013. EuCAP.

[25] R. Chirikov, P. Rocca, and A. Massa. Advanced clustering optimization in phased array design through genetic algorithms. In *Proc. of 13th Int. Workshop "Problems of Technics and Technologies of Telecommunication"*, pages 190–192, Ufa, Russia, 2012. USATU.

[26] Richard Courant and David Hilbert. Methods of mathematical physics. 1966.

[27] Y. Davidor. *Genetic Algorithms and Robotics: A heuristic strategy for optimization*. World Scientific, 1991.

[28] Lawrence Davis. Job shop scheduling with genetic algorithms. In *Proceedings of the 1st international conference on genetic algorithms*, pages 136–140. L. Erlbaum Associates Inc., 1985.

[29] Harald Dyckhoff. A typology of cutting and packing problems. *European Journal of Operational Research*, 44(2):145–159, 1990.

[30] Harald Dyckhoff and Ute Finke. Cutting and packing in production and distribution: A typology and bibliography. 1992.

[31] Russell C. Eberhart and Yuhui Shi. Comparison between genetic algorithms and particle swarm optimization. In *Evolutionary Programming VII*, pages 611–616. Springer, 1998.

[32] F. I. Emelchenkov. Onboard non-equidistant phased arrays with high density of phase shifters. *Antennas journal*, (11):102, 2005.

[33] Ailbhe Finnerty, Pavel Kucherbaev, Stefano Tranquillini, and Gregorio Convertino. Keep it simple: Reward and task design in crowdsourcing. In *Proceedings of the Biannual Conference of the Italian Chapter of SIGCHI*, page 14. ACM, 2013.

[34] Silvano Gambadoro. *Innovative Strategy for Element Clustering in Large Antenna Arrays*. PhD thesis, University of Trento, 2012.

[35] Hermann Gehring and A. Bortfeldt. A genetic algorithm for solving the container loading problem. *International Transactions in Operational Research*, 4(5-6):401–418, 1997.

[36] David Edward Goldberg. Computer-aided gas pipeline operation using genetic algorithms and rule learning. 1983.

[37] John J. Grefenstette and J. Michael Fitzpatrick. Genetic search with approximate function evaluation. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pages 112–120. L. Erlbaum Associates Inc., 1985.

[38] B.H. Gwee and M.H. Lim. Polyominoes tiling by a genetic algorithm. *Computational Optimization and Applications*, 6(3):273–291, 1996.

[39] David Hilbert, William Bragg Ewald, Wilfried Sieg, and Ulrich Majer. David hilbert's lectures on the foundations of mathematics and physics, 1891-1933: David hilbert's lectures on the foundations of arithmetic and logic 1917-1933/william ewald; wilfried sieg, ed. in collab. with ulrich majer and dirk schlimm. 2013.

[40] RJW Hodgson. Partical swarm optimization applied to the atomic cluster optimization problem. In *GECCO*, volume 2, pages 68–73, 2002.

[41] J.H. Holland. *Adaptation in natural and artificial systems*. University of Michigan press, 1975.

[42] L. V. Kantorovich and V. A. Zalgaller. *Optimal cutting of industrial materials*. Nauka, Sib. branch, 1971.

[43] Anirban Karmakar, Rowdra Ghatak, Utsab Banerjee, and DR Poddar. An uwb antenna using modified hilbert curve slot for dual band notch characteristics. *Journal of Electromagnetic Waves and Applications*, 27(13):1620–1631, 2013.

[44] Charles L. Karr. Design of an adaptive fuzzy logic controller using a genetic algorithm. In *ICGA*, pages 450–457, 1991.

[45] V. M. Kartak. Matrix-based searching algorihm for optimal solution of packing rectangles in a half-infitite band. *Information technologies*, 2:24–30, 2008.

[46] V. M. Kartak. Grouping method for continuous linear cutting problem solving. *Discrete analysis and operations investigation*, 16(3):47–62, 2009.

[47] James Kennedy. Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 3. IEEE, 1999.

[48] James Kennedy and Rui Mendes. Population structure and particle swarm performance. 2002.

[49] Dang Oh Kim, Che Young Kim, Jeung Keun Park, and Nam I Jo. Compact band notched ultra-wideband antenna using the hilbert-curve slot. *Microwave and Optical Technology Letters*, 53(11):2642–2648, 2011.

[50] Thiemo Krink, Jakob S VesterstrOm, and Jacques Riget. Particle swarm optimisation with spatial particle extension. In *Computational Intelligence, Proceedings of the World on Congress on*, volume 2, pages 1474–1479. IEEE, 2002.

[51] Y. V. Krivosheev. Sidelobe suppression in non-equidistant arrays made of equal rectangular subarrays. *III All-Russian conference "Radiolocation and communication" — IRE RAS*, pages 72–76, 2009.

[52] Yury V. Krivosheev and Alexandr V. Shishlov. Grating lobe suppression in phased arrays composed of identical or similar subarrays. In *Phased Array Systems and Technology (ARRAY), 2010 IEEE International Symposium on*, pages 724–730. IEEE, 2010.

[53] M.H. Lim, S. Rahardja, and B.H. Gwee. A ga paradigm for learning fuzzy rules. *Fuzzy Sets and Systems*, 82(2):177–186, 1996.

[54] Morten Lovbjerg and Thiemo Krink. Extending particle swarm optimisers with self-organized criticality. In *Computational Intelligence, Proceedings of the World on Congress on*, volume 2, pages 1588–1593. IEEE, 2002.

[55] Morten Lovbjerg, Thomas Kiel Rasmussen, and Thiemo Krink. Hybrid particle swarm optimiser with breeding and subpopulations. In *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2001, pages 469–476. Citeseer, 2001.

[56] R.J. Mailloux. Polyomino shaped subarrays for limited field of view and time delay control of planar arrays. Technical report, DTIC Document, 2006.

[57] R.J. Mailloux, S.G. Santarelli, and T.M. Roberts. Wideband arrays using irregular (polyomino) shaped subarrays. *electronics Letters*, 42:1019, 2006.

[58] R.J. Mailloux, S.G. Santarelli, and T.M. Roberts. New results using polyomino-tiled subarrays for time-delay control of wideband arrays. Technical report, AIR FORCE RESEARCH LAB HANSCOM AFB MA SENSORS DIRECTORATE, 2007.

[59] R.J. Mailloux, S.G. Santarelli, T.M. Roberts, and D. Luu. Irregular polyomino-shaped subarrays for space-based active arrays. *International Journal of Antennas and Propagation*, 2009, 2009.

[60] Robert J. Mailloux. *Phased array antenna handbook*. Artech House Boston, 2005.

[61] Robert J. Mailloux, Scott G. Santarelli, and Thomas M. Roberts. Array aperture design using irregular polyomino subarrays. In *Phased Array Systems and Technology (ARRAY), 2010 IEEE International Symposium on*, pages 740–744. IEEE, 2010.

[62] Rui Mendes, Paulo Cortez, Miguel Rocha, and José Neves. Particle swarms for feedforward neural network training. *learning*, 6(1), 2002.

[63] Vladimiro Miranda and Nuno Fonseca. Epso-best-of-two-worlds meta-heuristic applied to power system problems. In *Proc. of the IEEE Congress on Evolutionary Computation*, volume 2, pages 1080–1085, 2002.

[64] Andrea Francesco Morabito, Tommaso Isernia, Maria Grazia Labate, Michele Durso, and Ovidio Mario Bucci. Direct radiating arrays for satellite communications via aperiodic tilings. *Progress In Electromagnetics Research*, 93:107–124, 2009.

[65] Shigenori Naka, Takamu Genji, Toshiki Yura, and Yoshikazu Fukuyama. Practical distribution state estimation using hybrid particle swarm optimization. In *Power Engineering Society Winter Meeting, 2001. IEEE*, volume 2, pages 815–820. IEEE, 2001.

[66] Vincenzo Pierro, Vincenzo Galdi, Giuseppe Castaldi, Innocenzo M Pinto, and Leopold B Felsen. Radiation properties of planar antenna arrays based on certain categories of aperiodic tilings. *Antennas and Propagation, IEEE Transactions on*, 53(2):635–644, 2005.

[67] Lorenzo Poli, Paolo Rocca, Luca Manica, and Adnrea Massa. Pattern synthesis in time-modulated linear arrays through pulse shifting. *IET Microwaves, Antennas & Propagation*, 4(9):1157–1164, 2010.

[68] Lorenzo Poli, Paolo Rocca, Luca Manica, and Andrea Massa. Handling sideband radiations in time-modulated arrays through particle swarm optimization. *Antennas and Propagation, IEEE Transactions on*, 58(4):1408–1411, 2010.

[69] Charles Radin. The pinwheel tilings of the plane. *Annals of Mathematics*, pages 661–702, 1994.

[70] P. Rocca, R. Chirikov, and R. J. Mailloux. Polyomino subarraying through genetic algorithms. In *IEEE Int. Symp. on Antennas and Propagation*, pages 42–43, Chicago, USA, 2012. APS.

[71] Paolo Rocca, Lorenzo Poli, Giacomo Oliveri, and Andrea Massa. Synthesis of time-modulated planar arrays with controlled harmonic ra-

diations. *Journal of Electromagnetic Waves and Applications*, 24(5-6):827–838, 2010.

[72] E. V. Schepin. About fractal peano curves. *Papers of Mathematical institute RAS*, 247(0):294–303, 2004.

[73] Luk Schoofs and Bart Naudts. Swarm intelligence on the binary constraint satisfaction problem. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 2, pages 1444–1449. IEEE, 2002.

[74] Yuhui Shi and Russell C Eberhart. Fuzzy adaptive particle swarm optimization. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, volume 1, pages 101–106. IEEE, 2001.

[75] Yuhui Shi and Renato A. Krohling. Co-evolutionary particle swarm optimization to solve min-max problems. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 2, pages 1682–1687. IEEE, 2002.

[76] Thomas G Spence and Douglas H Werner. Design of broadband planar arrays based on the optimization of aperiodic tilings. *Antennas and Propagation, IEEE Transactions on*, 56(1):76–86, 2008.

[77] A. Kh. Sultanov, I. V. Kuznetsov, and V. V. Blokhin. *Signal and structural methods of throughput enhancement of communication systems.* 2006.

[78] R. Tang. Survey of time-delay beam steering techniques. *Phased array antennas*, pages 254–260, 1972.

[79] Raymond Tang and Richard W Burns. Array technology. In *IEEE Proceedings*, volume 80, pages 173–182, 1992.

[80] Johannes Terno, Guntram Scheithauer, Uta Sommerweiß, and Jan Riehme. An efficient approach for the multi-pallet loading problem. *European Journal of Operational Research*, 123(2):372–381, 2000.

[81] A. A. Tolkachev, E. N. Egorov, and A. V. Shishlov. Some trends in radiolocation and communication systems. *Radiotechnics*, (4):5–11, 2006.

[82] G. Toso, C. Mangenot, and AG Roederer. Sparse and thinned arrays for multiple beam satellite applications. In *Antennas and Propagation, 2007. EuCAP 2007. The Second European Conference on*, pages 1–4. IET, 2007.

[83] Frans Van den Bergh and Andries P. Engelbrecht. Effects of swarm size on cooperative particle swarm optimisers. 2001.

[84] Frans Van Den Bergh and Andries P. Engelbrecht. Training product unit networks using cooperative particle swarm optimisers. In *Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on*, volume 1, pages 126–131. IEEE, 2001.

[85] V. I. Vasiliev and B. G. Ilyasov. *Intellectual control systems using fuzzy logic: Course book*. USATU, 1997.

[86] E. S. Ventsel. *Operation investigation: tasks, principles, methodology*. Nauka, 1988.

[87] M. A. Verkhoturov, G. N. Verkhoturova, and R. R. Yagudin. Controlling 3d objects placement in assembly systems. *Vestnik USATU*, 16(8 (53)):45–51, 2013.

[88] Maria Carolina Viganó, Giovanni Toso, Gerard Caille, Cyril Mangenot, and Ioan E Lager. Sunflower array antenna with adjustable

density taper. *International Journal of Antennas and Propagation*, 2009, 2009.

[89] Chengjian Wei, Zhenya He, Yifeng Zhang, and Wenjiang Pei. Swarm directions embedded in fast evolutionary programming. *networks*, 9:11, 2002.

[90] Daniel S. Weile and Eric Michielssen. Genetic algorithm optimization applied to electromagnetics: A review. *Antennas and Propagation, IEEE Transactions on*, 45(3):343–353, 1997.

[91] Xiao-Feng Xie, Wen-Jun Zhang, and Zhi-Lian Yang. A dissipative particle swarm optimization. *arXiv preprint cs/0505065*, 2005.