Peiman Amini Behbahani

Paper: **DeGRaM: Using chess-like settings as a design space for generative art and design**

Topic:
Art

Author:
Peiman A. Behbahani
Architect & Researcher

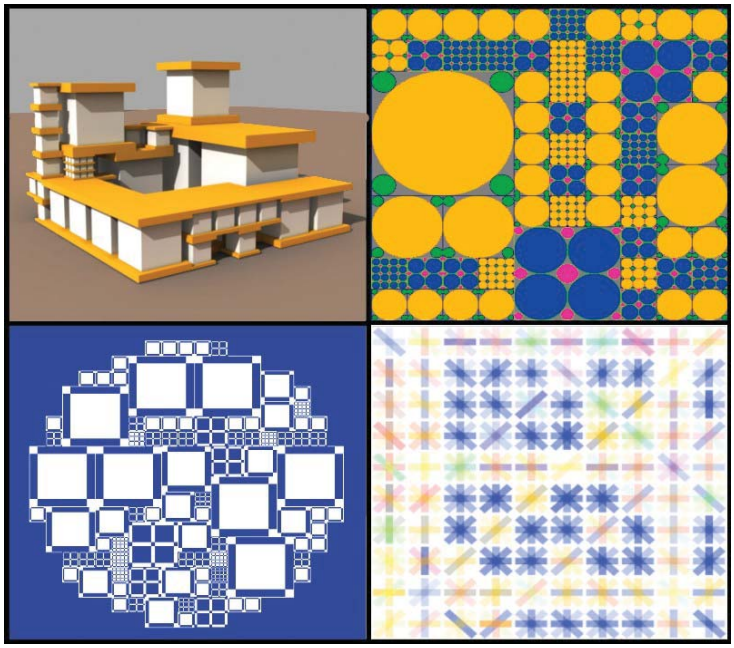
References:
[1] Sean Hanna, *"Where Creativity Comes From: the social spaces of embodied minds"*, Proceedings of HI'05, Computational and Cognitive Models of Creative Design. University of Sydney. pp. 45-70., 2005
[2] Margaret Boden, Mark d'Inverno, Jon McCormack, "*Computational Creativity : An Interdisciplinary Approach",* Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany

Abstract:

Chess is a complex game with an almost infinite number of piece arrangements and moves. Comparing pieces and their behaviours to automata and rules, respectively, each chess settings can generate its own unique design. It is also possible to extend the chess (both the grid and pieces) to explore different types of generative systems.

For this purpose, a software called DeGRaM (**De**sign by **G**rid **R**elations **a**nd **M**otions) is developed to calculate piece moves and interactions into maps and matrices for modelling them via different kinds of geometrical patterns in both 2D and 3D spaces. In this paper, the software's procedure, concepts and outputs are explained.

This study examines various alterations of chess, ranging from changing the board definition to piece's properties in order to generate initial matrices and how they may influence the software's outputs. To this end, it explores a number of concepts which are involved in producing these matrices. In addition, it applies different patterns and shape grammars suiting different design expectations. Overall, various compositions of piece definition and placements, grid configurations and geometrical patterns are inspected in this research, to explore potentials of this design space.



Samples generated by DeGRaM in 3D and 2D settings

Contact:
amini.peiman@gmail.com

Keywords:
chess, design space, generative design

DeGRaM: Using chess-like settings as a design space for generative art and design

Peiman Amini

MSc. Architecture

amini.peiman@gmail.com

Abstract

Complexity is considered a realm for creativity. However, it can be achieved via interaction of simple rules. Chess, and gridded board game in general, are complex situations resulted by simple pieces, rules and motions. The relative infiniteness of chess situations, while each of them being unique, makes it a proper space for generative design. For this purpose, a software called DeGRaM (**De**sign by **G**rid **R**elations **a**nd **M**otions) is developed to calculate piece moves and interactions into maps and matrices for modelling them via different kinds of geometrical patterns in both 2D and 3D spaces. In this paper, the software's procedure, concepts and outputs are explained. In addition, a number of visualisation pattern are introduced in both 2D and 3D outputs.

1. Introduction

Complexity has been considered a playground for creativity, especially when the creative procedure belongs to generative category. However, complexity is sometimes achieved by applying simple criteria. Two early examples of this argument are flocking behaviour of birds by Reynolds and early rules of cellular automata by Wolfram [1]. In a more detailed definition, a complex system is a system composed of interconnected parts that as a whole exhibit one or more properties (behavior among the possible properties) not obvious from the properties of the individual parts [2].

Games are other examples of simple rules which lead to complex situations, called *game complexity*. Chess is one of the most famous and widely played games in this regard. Since hundred years ago, the complexity of chess problems and positions was discussed. Shannon [3] estimated 10^{120} variation of chess situations from the initial position. A brief calculation reveals that there are around 70000 ways to place only two (opponent) pieces on the standard chess board (ignoring the impossibility of king-to-king neighbourhood and pawn on the first row).

$$N = \binom{6}{1} . \binom{6}{1} . \binom{64}{2} = 72576$$

2. Chess

The standard chess is consisted of an 8x8 grid of black and white squares on which 32 pieces of six kinds are placed, equally for each of two sides. Each kind of piece has its own value, moving and capturing style. Ignoring the exclusive rules (like *en passant*, checking or castling), the motion and interaction can be represented via simple 2D vectors (illustrated in **Table 1**).

Piece	Max Scale	Moves							
		1	2	3	4	5	6	7	8
Pawn	1	0,1	1,1	-1,1					
Bishop	n	1,1	-1,1	1,-1	-1,-1				
Knight	1	1,2	-1,2	1,-2	-1,-2	2,1	2,-1	-2,1	-2,-1
Rook	n	1,0	0,1	-1,0	0,-1				
Queen	n	1,1	-1,1	1,-1	-1,-1	1,0	0,1	-1,0	0,-1
King	1	1,1	-1,1	1,-1	-1,-1	1,0	0,1	-1,0	0,-1

***Table 1**. vectors and scales of pieces' moves in standard chess*

There are different sources for assessing values of chess pieces. A common evaluating [4] for pieces in which they are assigned values based on their moving and capturing capacities (except for the *king* as it is the ultimate goal of chess) is displayed in **Table 2.** The value of *king* in this table is based on its moving and capturing capabilities discarding its importance.

Piece	Value
Pawn	1
Bishop	3
Knight	3
Rook	5
Queen	9
King	4

***Table 2.** Values of pieces*

Moves in chess are primarily based on how much the possible destination of a piece is exposed to *threat* and gains *support* regarding to the value (or the role) of the piece. However, the best move is generally opted in a more holistic and planned view of the position. In this case, the player considers a map of threat-support-value for possible moves. The map may also be expanded for next step of moves. It will differ for each placement of pieces on the board offering an almost infinite number of maps.

Although the aforementioned maps are generated for individual cells ((in this paper the word “cell” is used instead of traditional “square” as the latter is only applicable for Cartesian 2D boards), we can define zones in chess. Zones are created when a number of adjacent cells have a similar characteristic regarding the chess concepts. For instance, if four adjacent cells are directly accessible for pieces of one side, it can be considered a *threat* zone for the other side.

The ultimate goal in the standard chess is to defeat the opponent by checkmating the king. However, there are different views on how one side may win or a draw is resulted. Capturing the opponent pieces to achieve either qualitative and/or quantitative superiority is a common practice in chess.

3. Chess variants and visualisation

The four main aspects of chess - pieces, moves, board and goal - have been subjects of alterations in order to achieve new, more challenging or more interesting types of chess.

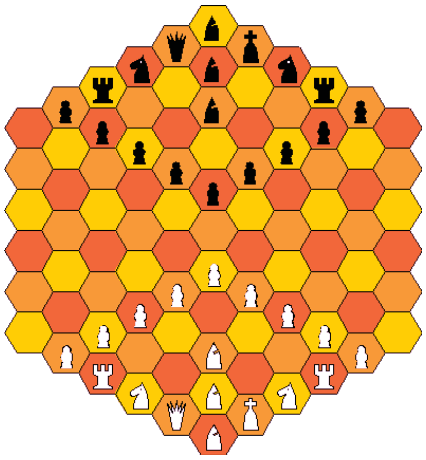


***Fig. 1**: Glinski's hexagonal chess [5]*

An early alteration of board can be found in Glinski's hexagonal (beehive) grid in early 1930s (**Fig. 1**). ChessVariants website [5] enumerates tens of *recognised* chess variants along with hundreds of less considered ones. Besides, there a number of patent registrations of chess. They range from changing the board into an MxN grid like Wildebeest or Tamerlane with 11x10 board with two “camel” pieces, changing or adding pieces like the Anti-king piece or the weird Raumschach complicated 3D five storey chess.

There has been a long time since chess is depicted in visual art. However, almost all objects have focused on static representation of pieces on a dark-light gridded board. In this case, the artistic effects are applied to the figure or pose of various pieces as well as dramatising the board (**Fig. 2**).

***Fig. 2**: A sample chess art [6]*

In this regard, DeGRaM (**De**sign by **G**rid **R**elation **a**nd **M**otion) is developed to capture a dynamic picture of chess, in particular, and other grid and piece games, in general. Therefore, it implements several concepts like moves (including their direction, steps and depth) and relation (such as threat, support or freedom) rather than the direct depicting of pieces and gridded texture.

5. DeGRaM-Chess

DeGRaM is a Windows based software developed in Free Pascal Compiler (fpc) via Lazarus IDE. Its goal is to generate complex visuals by using the simple rules and pieces of grid-and-piece games. Chess as a game with simple and familiar rules was selected as the primary base for DeGRaM pieces and grids.

The software is consisted of three parts that can perform separately. These parts include defining input that contains the grid and pieces rules and arrangement, calculative processing that is the main part of DeGRaM generating matrices and maps of moves and actions, and, finally, the visualisation section that contains geometrical and graphical functions for realising the maps into visible graphics. While the second part has to be done exclusively by DeGRaM, the other parts can be replaced by import or export extensions.

5.1. Defining

The software's input includes a set of piece and grid definition which can be entered either via its own interface or by being imported as text or comma delimited files. A defining file consists of two sections. The first sections includes properties of pieces such as, namely, name, caption, side, moving vectors with their possible depth (how far a piece is allowed to move) and capturing capability (side is only applicable to pieces like pawns, moves of which differ in different sides). **Table 3** shows the definition of bishop. The second section lists the cells by their coordinates and the occupying piece if one is filled. This method allows defining custom shapes of the board by excluding cells in an mxn grid. For example “1,3 B” indicates a bishop (the capital letter shows its side) in A3.

Property	Value(s)				
name	“Bishop”				
captions	“b”	“B”			
value	3				
moves	i	j	k	Depth*	capture or Move
#0	1	1	-	0	Both
#1	1	-1	-	0	Both
#2	-1	-1	-	0	Both
#3	-1	1	-	0	Both
*zero for depth means limitlessness of the move					

Table 3. *Definition of an ordinary bishop*

5.2. Matrix Mapping

This part is the main calculation section of the software. It mostly acts as a cellular automata processing in which cells in the grid behave like automata. Accordingly, for each cell a respective value is assigned for different parameters (threat, support, etc.). Another function of this section is to store paths and movements of pieces. This information is also saved to cells for caps of the path.

Level	Description	Condition
Absolute support	No threat but has support	$Fst(n) = false \wedge \sum_{i=1}^{n} Di = 0$
No confronts	No threat, no support	$Fst(n) = false \wedge \sum_{i=1}^{n} Ai = 0$
Support	Both, but support is more	$Fst(n) = false \wedge \sum_{i=1}^{n} Ai > 0$
Equilibrium	Both and as same	$Fst(n) = false \wedge \sum_{i=1}^{n} Ai = \sum_{i=1}^{n} Di$
Threat	Both, but threat is more	$Fst(n) = true \wedge \sum_{i=1}^{n} Di > 0$
Absolute threat	No support but has threat	$Fst(n) = true \wedge \sum_{i=1}^{n} Di = 0$

Table 4. *description of support-threat measure*

A measure of six levels is developed to assess threat-support duality for each cell (**Table 4**). This measure is based on the presence and interaction of support and threat. To explain, it is considered that in a situation where a number of pieces are in direct access of each others, they start capturing each other. The capturing is

stopped as the one of the opponents has no other involved pieces in the scene. Therefore we have the formula below, where *Fst(0)* is FALSE, *n* is total number of captures, Ai and Di stand for the respective values of attacker's and defender's pieces in the captureing *i.* Therefore, *Fst(i)* is a boolean function which returns TRUE if the cell is in threat.

$$Fst(i) = Fst(i-1) \vee \left(\sum_{i=1}^{n} Ai < \sum_{i=1}^{n} Di\right)$$

If *Fst(n)* becomes true (which means at a capturing sequence the attacker's total loss was lower than the defender's, so he/she stops capturing) this is considered a threat. **Table 4.** displays all possible threat-support measure levels.

The concept of freedom is indirectly significant in piece safety. As a matter of fact, it suggests the number of moves applicable by a piece in its current cell towards the threat free destinations. This is important when a piece is under attack and is seeking for an escape way.

5.3. Visualisation

The third part interprets the matrices into various visuals. Several parameters such as the motion concepts, board settings and zoning are involved in the graphical representation based on which calculation method (cellular or path) is applied. In general, there have been three parameters of presentation:

5.3.1. Dimension

Due to prevalence of 2D board alterations of chess, this software is mostly focused on 2D outputs. However, a 3D geometrical pattern as also programmed to generate design resembling building or furniture outlines.

5.3.2. Matrix

As is mentioned, two matrices - cell's features and moving paths - are generated by DeGRaM. In a presentation method one or both can be shown. However, only one of them is regarded as the main parameter while the other is displayed as texture or colour.

5.3.3. Zoning

Zoning indicated unifying or averaging the values of adjacent cells (single or combined). Functionally, it may be useful to illustrate the dangerous and safe parts of the board. However, its main benefit is aesthetical by variegating the monotonous cells juxtaposition. The current version of DeGRaM uses two scale and percentage parameters to shape the zones. The scale is a number *n* that indicates the maximum *nxn* (or *nxnxn*) block of cells which can be combined as a zone. The value *n* can not be greater than half of the smallest dimension of the grid. Percentage evaluates the proportion of the *n* block that must have the same map value to be considered a zone. It should between 51% and 100%.

5.4. Limits

There are a number of limitations to the efficiency of this software since it is in the initial stages of research. The most important of them is the time consuming process of thousands of recursive motions. This issues reveals itself more in 3D grids. Memory management is also a barrier when the number of steps to predict is high. In addition, as this program primarily aims to generate output on the basis of *simple* rules, the pieces' behaviour is reduced to mere moving vectors. Therefore, it ignores many chess rules like castling, checkmate, pawn's double movement, etc.

6. DeGRaM Output

6.1. 2D Grid

DeGRaM has initially aimed 2D Cartesian grids - *boards.* Boards are defined either as complete *mxn* grids or with excluded or missing cells. As is mentioned, pieces are defined by their value, side and moving vectors while being placed on the board in a custom way. Before the calculation starts, the number of steps should entered. This number is often selected as one or two because larger values means more processing time and memory usage. Besides, any number larger than two will not take a great difference in complexity with the two-step calculation. The method of visualisation is input after the calculation with the scale and percentage of zoning. So far, three methods for 2D visualisation are developed in the program. They are named by their appearance: wind-rose, asterisk, articulation.

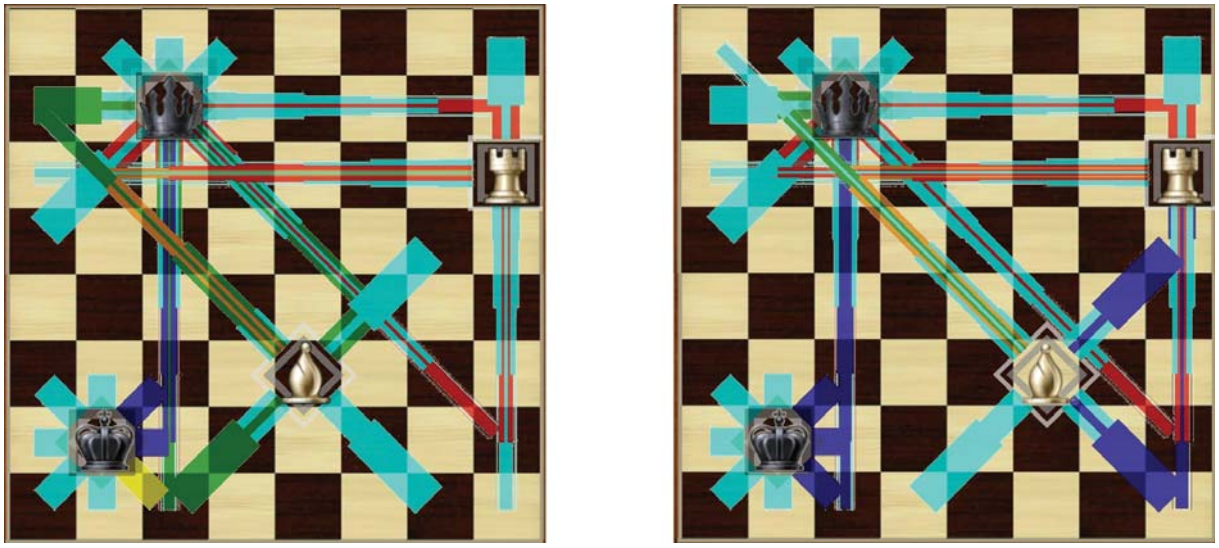


Fig 3. *One-step wind roses by replacing the white bishop from e5 to e6*

6.1.1. Wind-rose

It has been named due to its resemblance to climatic wind roses. It is consisted of bars aligned according to the direction which can also illustrate other properties such as depth of motion and threat-support equilibrium. These data are illustrated through

the colour, angle and length and width of the bars. Pieces also are displayed as simple geometrical shapes which is somehow taken from their moving pattern (e.g. squares for rook, 45 degree rotated squares for bishops, eight winged stars for queen, etc). Although these shape patterns are only applicable for standard pieces. **Fig. 3** displays the one-step visualisation for two piece arrangements. The difference between them is in the location of the white bishop. In **Fig. 4**, the change of the output by moving a piece is illustrated in 8x8 grid with two-step visualisation. **Fig. 5** shows the same piece arrangement while the square *d4* is excluded. While the colours indicate the level of threat-support, their opacity represents the step of move (more opaque means earlier steps).

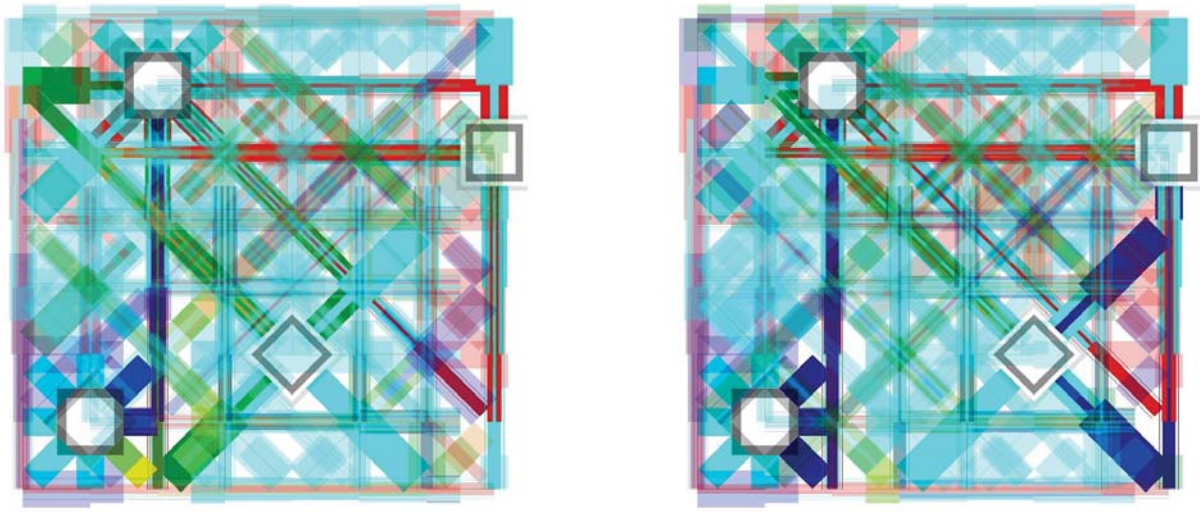


Fig 4. *Two-step wind roses by replacing the white bishop from e5 to e6*

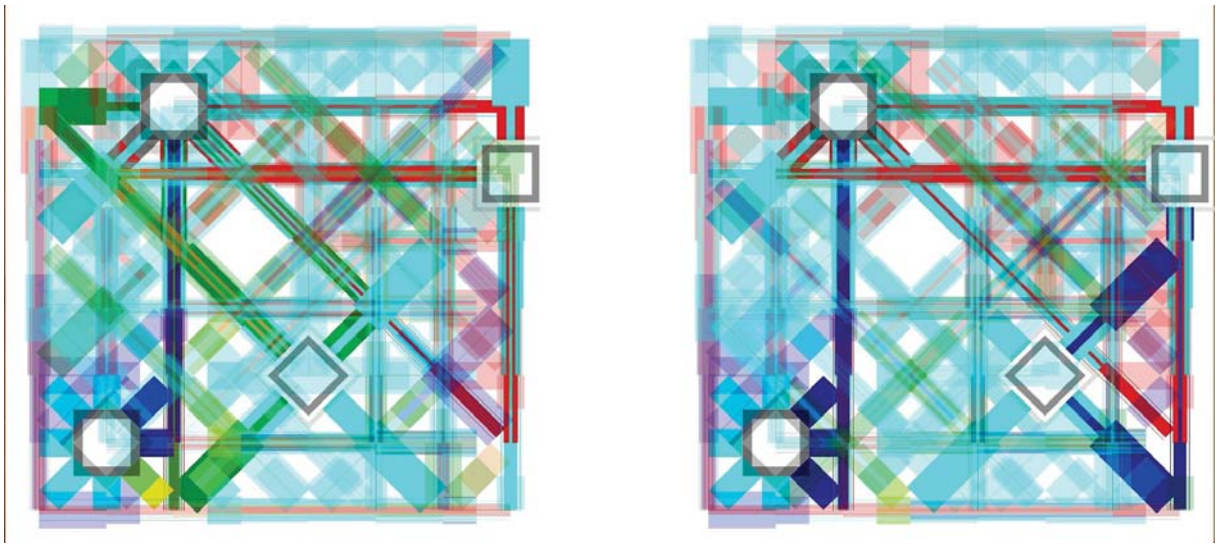


Fig 5. *Two-step wind roses (same as Fig 4.) while the d4 square is excluded*

6.1.2. Asterisk

While in the *wind-rose* the direct path of piece move is illustrated, in the *asterisk* method, only its effect on individual cells is displayed. In this case, the length and

width of bars (asterisk wings) are the same, determined by the width of the cell. On the other hand, colours' function and assignment are as same as the wind-rose method. **Fig. 6** show the asterisk visuals for their respective wind rose equivalents (Fig 4).

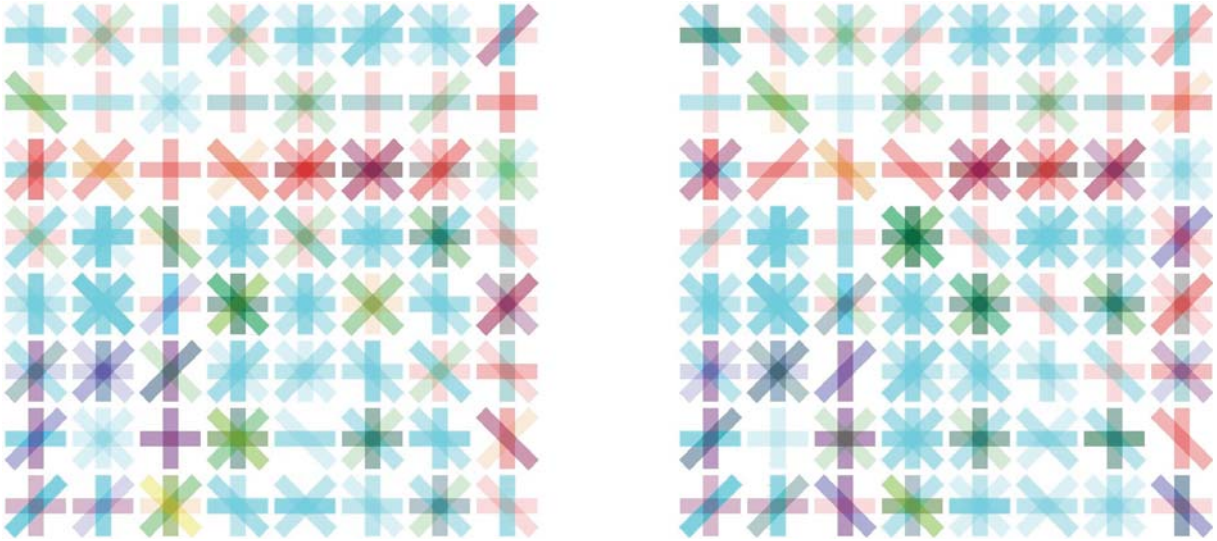


Fig 6. Two-step asterisks (same arrangement as Fig 4.)

6.1.3. Articulation

This type of visualisation focuses only on the value assigned to cells. However, the value is affected by presence of the motion, allowing tracking of the moves. For instance, it will consider threat-support values for either one piece, one side or an average of all sides for each cell. The severity of threats or supports is represented as articulation of cells. Colours are applied to identify sides. So far, two articulation methods are developed: squares and circles. **Fig. 7** shows the square and circles visuals for their respective wind rose equivalents (Fig. 4).

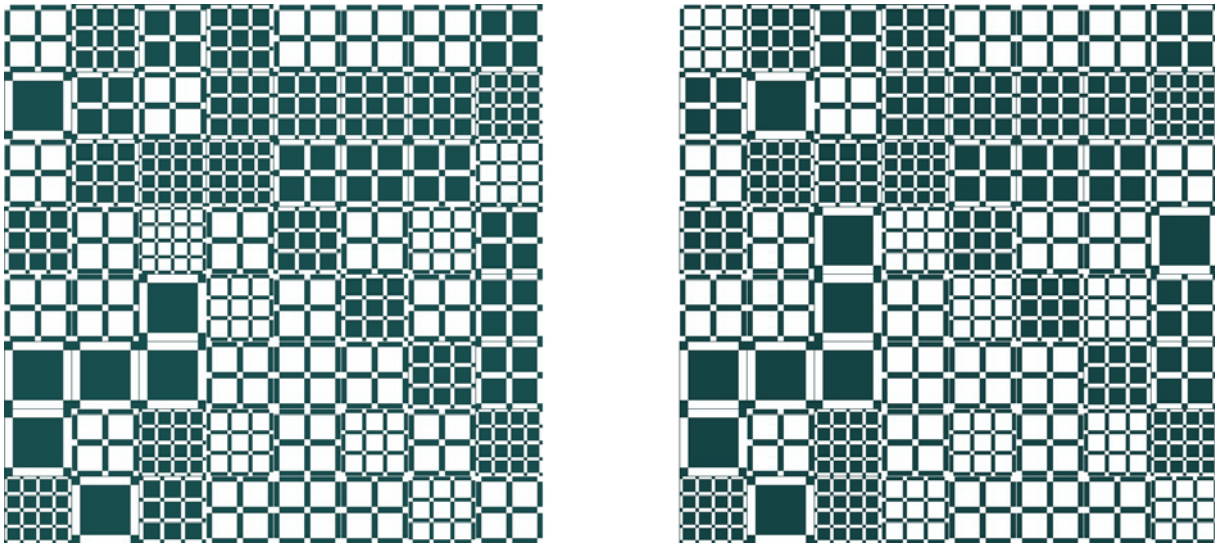


Fig 7. *Two-step square articulation (same arrangement as Fig 4.)*

Since this presentation lacks the direct display of moves, it is more difficult to establish a relation between the output and the original piece arrangement, especially in two-step calculation. The articulation also facilitates zoning as the visualisation is completely modular as well as the board itself. The output-arrangement relation is more undetectable by zoning as several paths are dissolved into zones. **Fig. 8** illustrates zoning for one of the previous settings both in one step and two step configurations.

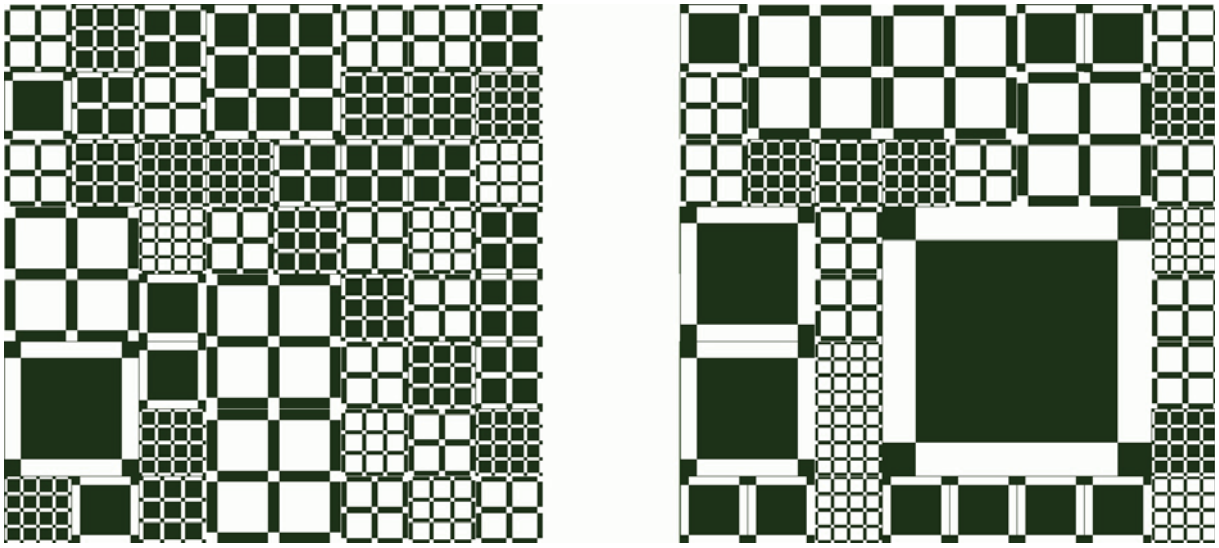


Fig 8. *left:Two-step square zoned articulation (same arrangement as Fig 4.left), right: One-step square zoned articulation for the same settings*

6.2. 3D Grid

3D grids differ from their 2D counterparts in various ways. First, as the 2D pieces are not suitable for this settings, the piece definition is completely different to the standard chess. For instance, the moves for 2D piece does not normally exceed eight vectors, while for a 3D piece, it may be as many as 26 vectors. Second, the visualisation differs as many of the cells are hidden behind other cells. Computationally, 3D grids takes substantially more time and memory.

Grid definition follows the same rules at in 2D type. An initial *a*x*b*x*c* grid is defined by listing its cells coordinates and excluding unwanted cells. Although 2D pieces are possible in 3D grids, they will not gain a satisfactory result as they do not interact in Z-different cells. It is also possible to rotate 2D moves by X or Y axes to see more interactions. However, original 3D pieces are inspired by 2D pieces. For instance, a 3D rook is similar to a 2D one with extra up and down vertical vectors. Detecting the output-arrangement relation is more difficult in 3D grids even in one step results as motion paths are hidden behind cells nearer to the point of view. This is severd in zoned presentation.

Since the DeGRaM was originally developed for 2D boards, there has not been as much diversity for the 3D outputs. The only presentation method is labelled *cubes* which is the 3D counterpart of square articulation in 2D. Like 2D squares, the cubes articulation also may be undergo ne zoning. *Empty zones* are a feature which is

given to 3D grids. An empty zone is a zone where most of its cells are neither threatened nor supported by any sides. Please, note that rendering is done by Vray® rendering engine in Autodesk 3DSMax® software.

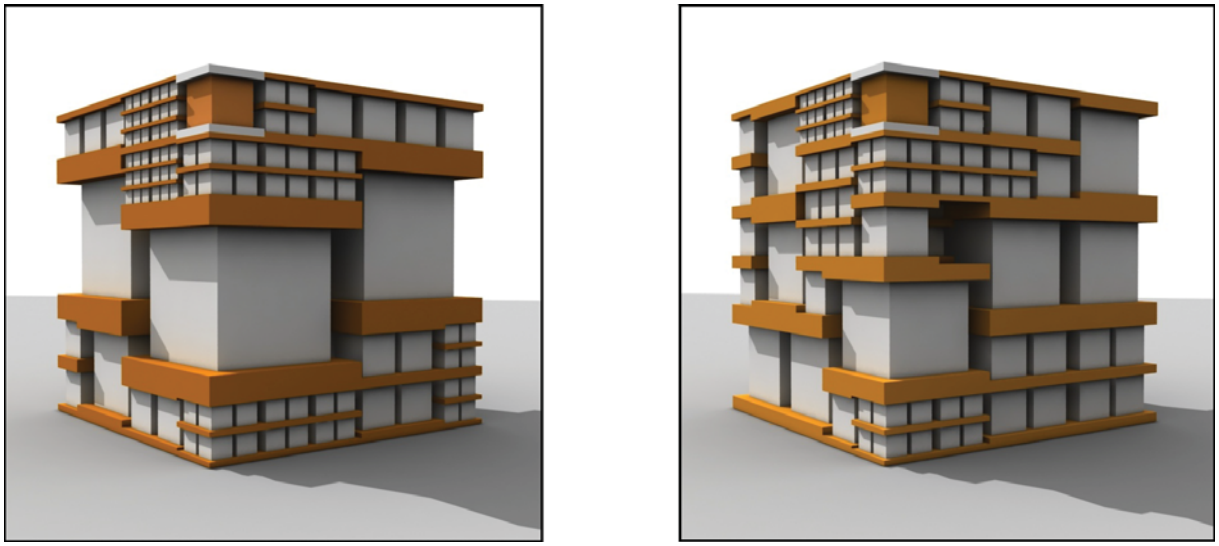


Fig 9. *left:Two-step 3D articulation with 3x zoning (70%). right: Two-step articulation with 2x zoning (70%) for the same settings (R:1,2,1; q:5,1,4; K:2,5,6) in 6x6x6 grid.*

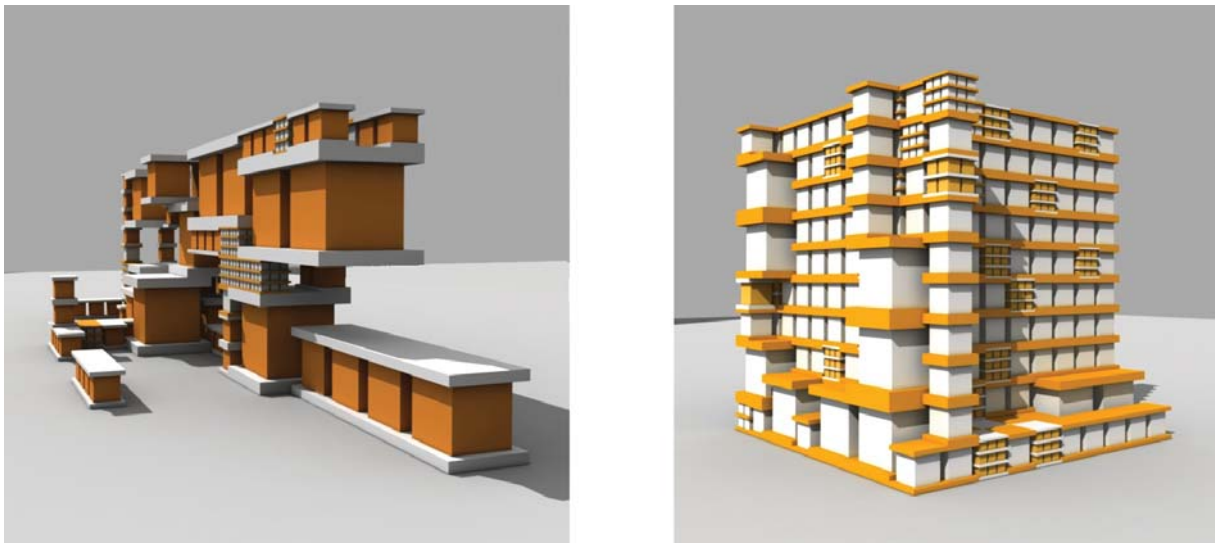


Fig 10. *two different 3D zoned outputs of DeGRaM for larger grids.*

7. Discussion

DeGRaM is a software aimed to illustrate motion and relations in gridded games, especially similar to chess. It uses various concepts in chess such as threatening and supporting to calculate the consequences of the moves for every piece, in order to generate unique maps for each grid configuration. Thereafter, the mapping results are visualised by geometrical patterns.

DeGRaM is still in its early stages of development. It is depended on Cartesian coordinates and simple fixed piece moves. In addition, the software does not regard chess as a win-loss game. Therefore, its calculation and piece motion are based on mere simple one or two move prediction. Besides, the king's role in the game is reduced to a simple piece without being *checked.*

Finally, the software's outputs only have aesthetical purposes. It can be assumed that some of outputs can be planned for specific goals. For example, the 3D outputs may well be used for exploring design space on building architecture.

References

[1] Joslyn, C. and Rocha, L. (2000). Towards semiotic agent-based models of socio-technical organizations, Proc. AI, Simulation and Planning in High Autonomy Systems (AIS 2000) Conference, Tucson, Arizona, pp. 70-79.

[2] Hanna S. (2005) Where Creativity Comes From: the social spaces of embodied minds, *Proceedings of HI'05, Computational and Cognitive Models of Creative Design*. University of Sydney. pp. 45-70. ISBN: 1-86487-789-8

[3] Shannon, C. (1950). Programming a Computer for Playing Chess. *Philosophical Magazine 41* (314)

[4] Chess.com; URL: http://www.chess.com/article/view/chess-piece-value

[5] ChessVariants.com

[6] URL : http://cutcaster.com/photo/800937986