3D MAPPING OF ISLAMIC GEOMETRIC MOTIFS

Z.SAYED

PHD

2017

3D MAPPING OF ISLAMIC GEOMETRIC MOTIFS

Zahra SAYED

Submitted for the Degree of

Doctor of Philosophy

Faculty of Engineering & Informatics

University of Bradford

2017

بسم الله الرحمن الرحيم

In the Name of Allah the Most Gracious, the Most Merciful

Dedicated To:
My Beloved Parents

Abstract

Zahra Sayed

3D Mapping of Islamic Geometric Motifs

Keywords: Islamic geometry, shape grammar, 3D Mapping, motifs, parameterized, shell mapping, point set registration.

In this thesis a novel approach in generating 3D IGP is applied using shape grammar, an effective pattern generation method. The particular emphasis here is to generate the motifs (repeat unit) in 3D using parameterization, which can then be manipulated within 3D space to construct architectural structures. Three unique distinctive shape grammar algorithms were developed in 3D; Parameterized Shape Grammar (PSG), Auto-Parameterized Shape Grammar (APSG) and Volumetric Shell Shape Grammar (VSSG).

Firstly, the PSG generates the motifs in 3D. It allows one to use a single changeable regular 3D polygon, and forms a motif by given grammar rules including, Euclidean transformations and Boolean operations. Next, APSG was used to construct the architectural structures that manipulates the motif by automating the grammar rules. The APSG forms a wall, a column, a self-similarity star and a dome, the main features of Islamic architecture. However, applying Euclidean transformations to create non-Euclidean surfaces resulted in gaps and or overlaps which does not form a perfect tessellation. This is improved upon by the VSSM, which integrates two key methods, shell mapping and coherent point drift, to map an aesthetically accurate 3D IGM on a given surface.

This work has successfully presented methods for creating complex intricate 3D Islamic Geometric Motifs (IGM), and provided an efficient mapping technique to form visually appealing decorated structures.

Acknowledgements

First and foremost, I would like to praise God, the Almighty, for bestowing upon me the opportunity and capability to successfully complete this thesis. Deepest gratitude to my beloved parents and my family for the greatest gift; their everlasting love and support.

I am thankful to my entire supervisory team, all of whom played an integral role in the making of this thesis. Firstly, Prof. Hassan Ugail for his invaluable knowledge, support and guidance throughout the entire PhD; Dr. Jon Purdy, for his keen interest and stepping in as my external supervisor and finally to Dr. Ian Palmer, for being the selfless supervisor he is, in the initial stage of my PhD. A special note of gratitude to Dr. Carlton Reeve and Dr. David Connah for their droplets of wisdom.

An extended thanks to the Centre of Visual Computing (CVC) for partially funding my research. Also, to all the generous photographers who shared their beautiful photography enabling me to create a visually striking thesis.

I would also like to thank all my colleagues at the CVC for creating a positive and stimulating atmosphere in the many offices we shared throughout my PhD, a special note to Nosheen Hussain, Nur Baini Ismael, Ali Bukar and Ahmad Al-Dahoud. For the proofreading, critical feedback and advice. Thank you.

Finally to the Beloved Prophet Muhammad (May peace and blessings be upon him), for teaching me the importance of seeking knowledge.

List of Publications

Z. Sayed, H. Ugail, I. Palmer, J. Purdy, and C. Reeve, “Auto-Parameterized Shape Grammar for Constructing Islamic Geometric Motif-Based Structures,” Trans. Comput. Sci. XXVIII, vol. 9590, pp. 146–162, 2016.

Z. Sayed, H. Ugail, I. Palmer, J. Purdy and C. Reeve, "Parameterized Shape Grammar for Generating n-fold Islamic Geometric Motifs," *2015 International Conference on Cyberworlds (CW)*, Visby, 2015, pp. 79-85

Contents

Figures and Tables

Figures

List of Abbreviations

2D	2-Dimensional
3D	3-Dimensional
AD	Anno Domini era
APSG	Auto-Parameterized Shape Grammar
BTF	Bidirectional Texture Function
CG	Computer Graphics
CPD	Coherent Point Drift
CSG	Constructive Solid Geometry
EM	Expectation Maximization
GDM	Generalized Displacement Mapping
GMM	Gaussian Mixture Model
ICP	Iterative Closest Point
IGM	Islamic Geometric Motifs
IGMBS	Islamic Geometric Motif-based Structures
IGP	Islamic Geometric Patterns
IS	Initial Shape
LOD	Level of Detail
PDF	Probability Density Function
PIC	Polygons-In-Contact
PSG	Parameterized Shape Grammar
PSR	Point Set Registration
RPM	Robust Point Matching

SA	Shape Algebras
SG	Shape Grammar
VSSG	Volumetric Shell Shape Grammar

1. Introduction

"Research is creating new knowledge."[1]

Islamic art is a distinctive art form that is not restricted to religious work, but instead is formed from traditional Muslim cultures [1]. It features the art and architecture that was historically created in the land that was ruled by Muslims. Islamic art is usually classified by the dynasty reigning at the time the art work was produced. The formation of Islamic art emerged during the reign of the Ummayad dynasty (661-750 AD) and historically concluded with smaller yet powerful regional dynasties, the Safavids, the Ottamons and the Mughals [2]. A timeline of all the dynasties is shown in Appendix A. Due to the geographic spread and history, Islamic art is formed by a wide range of styles and influences. There are three main categories in Islamic art [3]–[5]:

1. Arabic Calligraphy
2. Arabesque (Floral Decoration)
3. Islamic Geometric Patterns (IGP)

Arabic Calligraphy is an art form of scripting Qur'anic verses on different mediums, from architectural walls to book covers and paper with different Arabic

[1] *Neil Armstrong.*

fonts. It is known as the 'Jewel' of Islamic Art [6]. The floral decoration that is usually decorated around the Arabic script is known as Arabesque. It represents rhythm and growth of life prominently by the use of spirals [7]. Islamic Geometric Patterns (IGP) are based on geometry and geometric shapes. They are a unique category in Islamic art functioning as a major element of adorning architectural surfaces.

Figure 1.1 shows examples of all three categories of Islamic art. The ‘incipit’ page (left) and the ‘carpet’ page (right) is from a part of the Sultan Faraj ibn Barquq’s thirty volume Qur’an [8]. An incipit page is a page followed immediately after the front cover of the Qur’anic manuscript that includes both calligraphy and decoration of either or both arabesque and IGP. The carpet page on the other hand is a page that is fully decorated and is known as a ‘carpet’ page as it resembles oriental carpets that are placed at the entrance of an architectural building [8]. The creativity of 2-Dimensional (2D) and 3-Dimensional (3D) expressions can also be seen clearly in Islamic art and architecture [3] see Figure 1.2.

Much research has been carried out in IGP in the computer graphics field, from forming classifications of IGP [9] to creating algorithms for constructing IGP [10], [11]. It is also been done commercially, using 3D software and through the use of fabrication methods [12]. Methods of constructing IGP will be discussed in chapter 2, discussing their advantages and disadvantages, along with the state

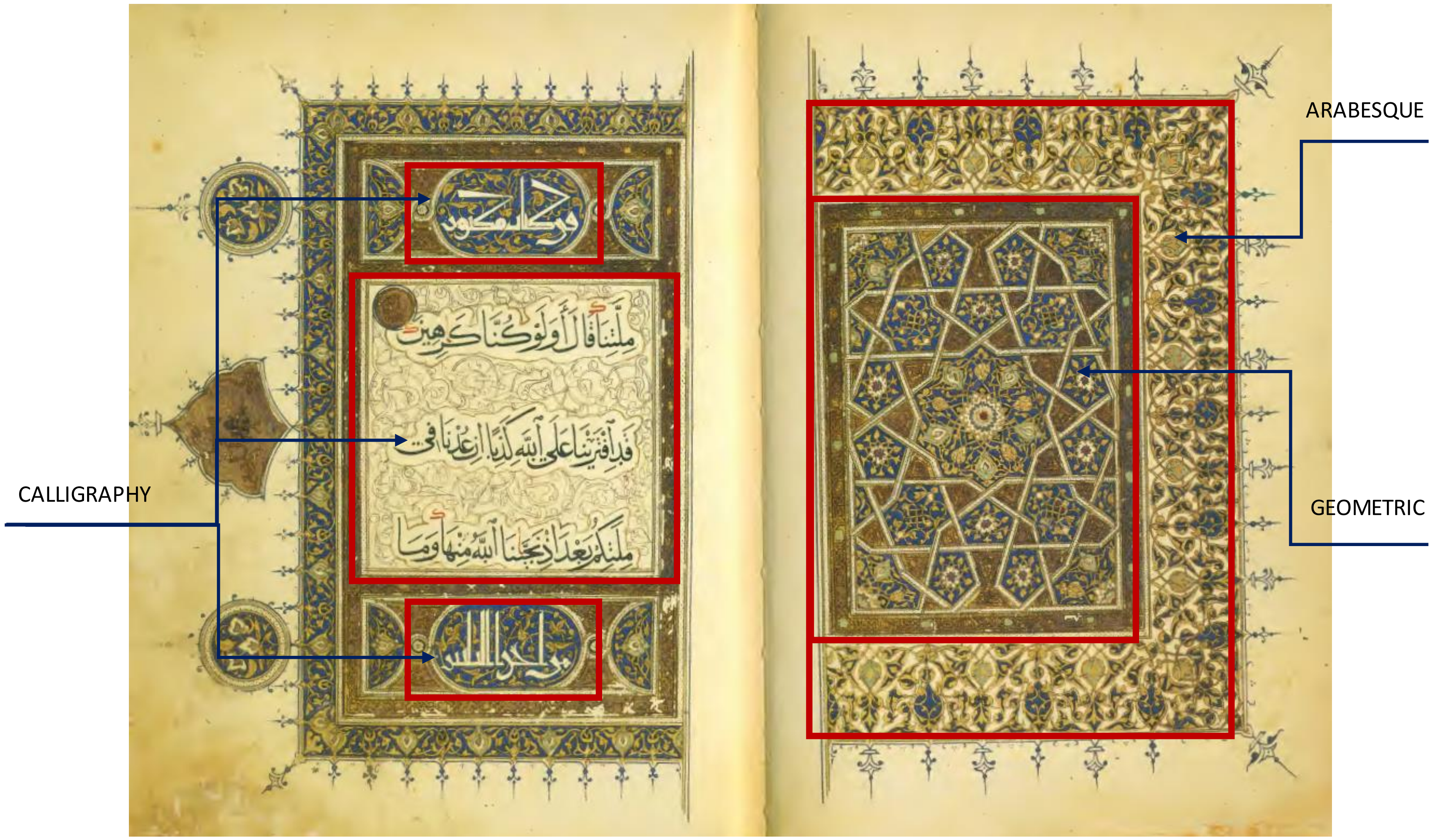


Figure 1.1 Example of the application of calligraphy, arabesque and IGP [8].

of Art. A historical background is briefly stated in the next section, for the reader to gain insight and understanding of IGP, including a detailed definition on how to define a pattern as ‘Islamic’.

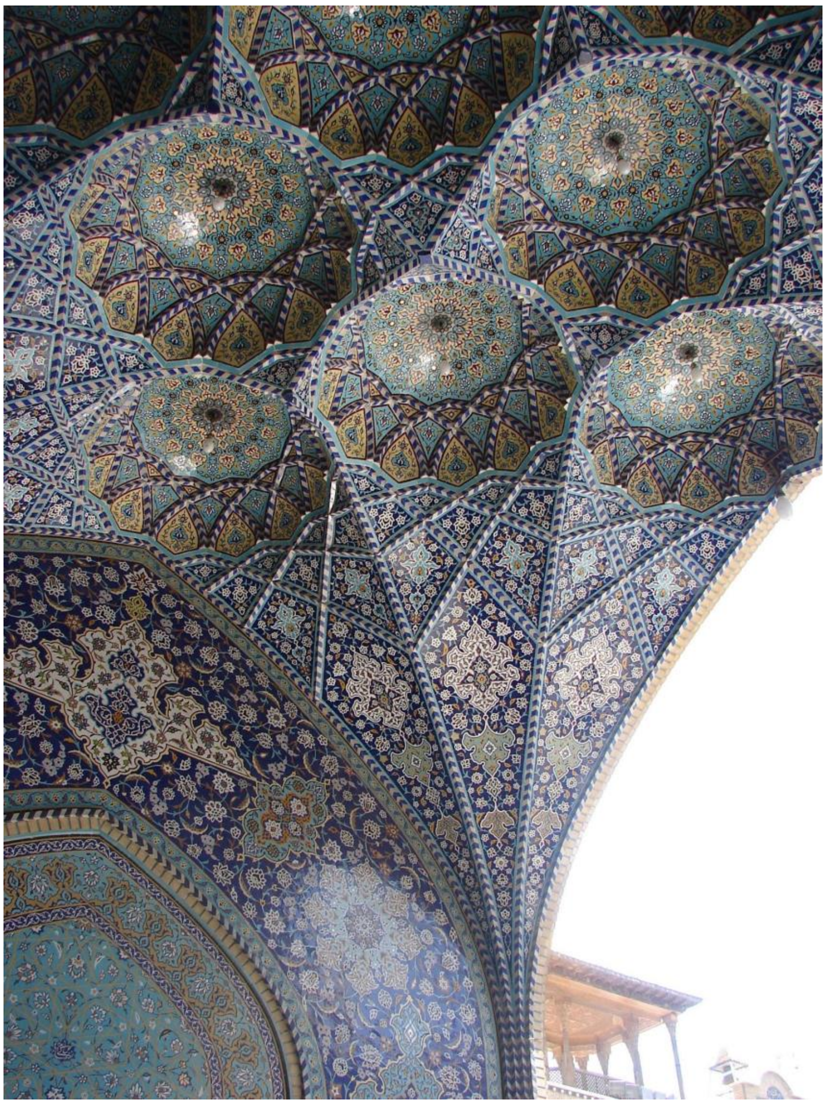


Figure 1.2 Exterior view of the entrance of Sayyeda Fatima al-Ma’suma mausoleum, Qom, Iran [Photography by Ali Reza Sarvdalir].

1.1 Historical Background

Geometric patterns are one of the major categories of Islamic Art. They adorn everything from architectural structures to decorated hand crafted calligraphy. There are many different types of IGP. Some are simple, like a repetition of regular polygons and some more complex. These patterns are mainly referred to as 'Medieval Islamic tiles' as they were produced by Medieval Islamic Artisans [13].

To define a pattern 'Islamic' is not as simple as there is no definitive answer. The criteria stated below by Abas [6] however argues how an IGP can be interpreted as 'Islamic' within reason.

"*Definition: An Islamic pattern is one which satisfies one or more of the following criteria:*

1. *The pattern is transcribed with Arabic Calligraphy from the Qur'an.*
2. *The pattern was invented between 900 A.D and 1500 A.D and was used to decorate architectural surfaces or other works of art for Muslims, in a culture where the majority of the population, or at least the ruling element, professed the faith of Islam.*
3. *The pattern is derived from one or more patterns which satisfy criterion 2 and is such that the characteristic shapes from the original (or originals) are recognizable."* [6]

The first criterion is the true fact that in Islam the word of God, which is the Holy Qur'an, is of great value, and any object that adorns its verses will be known to be Islamic. Secondly, Abas [6] claims IGP were first constructed between 900AD up until 1500 A.D, although other researchers have claimed the formation to be earlier [2]. However, this criterion explains that any pattern constructed during that period which was also adorned on an architectural surface, in a land where the majority of the population were Muslims of the faith of Islam, can be named Islamic. This however does not imply that the pattern formed was actually constructed by a Muslim. Artists from various religions, like Christians, Jews & Hindus, also lived in the Islamic world at that time and were also inventing geometrical patterns [6]. For example, the Jalis (screens) in the Mughal Era were crafted by Hindu craftsmen. Lastly, the third criterion, in this modern day, if a pattern is derived from an original IGP, which resembles it closely through its form of shape can also be named 'Islamic'. Hence, this above criteria can be used to distinguish if a pattern can be considered Islamic or not.

Although a definition is provided by Abas, it only defines IGPs in historical and geographical terms. Researchers have interpreted IGPs being based on Greek geometry, where the formation of regular polygons are constructed through Greek mathematical theory (explained in chapter 2). Generally the IGPs are formed of four basic shapes:

- Circle/interlaced circles
- Squares/ 4-sided polygons

- Regular pointed stars (derived from squares and triangles inscribed in a circle)
- Multi-sided polygons

The main feature of IGP is symmetry. Symmetry plays a vital role in IGP and how they are formed. It unifies the patterns by creating a sense of balance and unity. Complex IGP which are constructed by the overlapping of shapes create an *emergence*. Emergence is another key characteristic in IGP. It can be defined as emerging shapes, which are identified either during a pattern generation or when the final pattern is generated [5]. Self-Similarity is where the shape is scaled proportionally exhibiting the same form, this is also a characteristic employed in IGP.

Islamic art is depicted by having a non-figurative approach. It is a way of showing unity of God, replacing as many figurative images of God to a single perception of God, 'Light' [6]. Light is what embodies and guides a Muslim, and is a means to shine, as it is mentioned in the Holy Qur'an,

“God is the Light of the Heavens and the earth. (24:35)”

[14]

Geometry was seen to be of a unifying structure, an art form of the unseen. Jowers el al. [5] describes how visual statements of the Islamic religion are created by the use of geometry in Islamic Art. For example, the famous 8-pointed star, which is known as the Khatemi Sulemani (Seal of Solomon) [6], symbolizes the eight bearers of God's throne. As it states in the Holy Qur'an,

"And the angels will be on the sides thereof, and eight will upload the Throne of thy Lord that day, above them. (69:17)"

[14]

Tessellations within IGP is a technique that was adapted from weaving carpets and rugs. The most popular art form in the rise of Islam was carpet weaving [15]. Carpets were applied for many purposes, both for a nomad who was travelling the earth and as an adornment for the rich. They were used as floor coverings, prayer mats, tent decorations and canopies [16]. The carpet also has significant importance in the Qur'an, one as a metaphorical use of the earth and also a symbol of richness that a believer will be rewarded with, as it states in the Holy Qur'an,

"Reclining on green cushions and beautiful fine carpets." (55:76)

[14]

Carpet weaving is formed by interlacing and intertwining yarns of thread, which creates repetition within the woven carpet. By adopting this traditional method, interlacing and repetition was adapted in the Islamic art form, applying it onto different mediums.

1.1 Aim and Objectives

This research aims to develop an efficient 3D mapping technique for mapping 3D Islamic Geometric Motifs (IGM) onto any given surface. The novel technique applies Euclidean geometry onto Non-Euclidean surfaces, overcoming the problem of gap formations and overlaps. To achieve the aim, the following objectives need to be carried out:

- To review construction methods of Islamic Geometric Patterns (IGP), both traditional and computerized, concluding to one that can support the generation of 3D IGM.
- To form a Parameterized Shape Grammar (PSG) that creates *n*-fold 3D IGM with the use of a single changeable initial shape.
- To automate the PSG, which will automatically apply the generated 3D IGM to form architectural structures, by the use of algorithms.
- To investigate the state-of-art in volumetric surface mapping, and how it can be adapted within the PSG.
- To design, develop and implement the Volumetric Shell Shape Grammar (VSSM) algorithm that can map 3D IGM efficiently onto point cloud surfaces.
- Extend the VSSM algorithm to map 3D IGM onto given surfaces.
- To validate the proposed mapping technique and investigate the limitations within it thus outlining future directions of research.

1.2 Research Contributions

This research has resulted in a number of novel contributions extending the current state-of-art. These are presented in detail in Chapters 4-6.

1. **3D Islamic Geometric Motifs (3D IGM)** – the formation of 3D Islamic geometric motifs with the use of a single initial regular polygon through shape grammar.
2. **Islamic Geometric Motif Based Structures (IGMBS)** - Automating a parameterized shape grammar to create four Islamic architectural structural surfaces with a generated 3D motif.
3. **Volumetric Shell Shape Mapping (VSSM)** – Map the 3D motif on a point cloud of a parametric surface through integration of CPD and shell mapping.

1.3 Thesis Overview

The thesis is structured as follows:

Chapter 2: Islamic Geometric Patterns. This chapter reviews the construction methods of IGP, both traditional and computerized.

Chapter 3: Mapping Techniques. This chapter provides a discussion on the literature on 3D mapping techniques from the different types of texture maps, categories of scale of a geometry and volumetric texture storage and mapping techniques.

Chapter 4: 3D Islamic Geometric Motifs. The initial part of the research is described in this chapter, including the methodology and implementation of creating 3D IGM with parametric shape grammar. It initiates the chapter with theory on parametric shape grammar and algebra of shapes, describing how the initial shape is formed with the shape grammar definition. Once the methodology of the initial shape and rules are described, the method is implemented and the results are presented and analysed.

Chapter 5: Architectural Structures. The next step is to manipulate the motifs to form an architectural structure. A theoretical background on shape grammar implementation within the architectural field is discussed, with a detailed explanation of the theory of tessellations within IGP.

Chapter 6: Point Cloud Mapping. This chapter explains the final part of the research, where shape grammar is integrated within shell mapping, forming the 'Volumetric Shell Shape Grammar' (VSSM). The point set registration techniques are explained with a detailed description of the Coherent Point Drift (CPD). VSSM, a generalized mapping technique, adapts the CPD method, to form the mapping function.

Chapter 7: Conclusion. This is the concluding chapter in this thesis, containing the concluding remarks of the proposed algorithm, with its limitations and future work.

2. Islamic Geometric Patterns (IGP)

"God is beautiful and He loves beauty."[2]

The following chapter introduces the reader to the construction methods of IGP, traditional and computerized. The chapter initiates with a brief insight in the historical documentation preserved, following onto a technique for categorizing IGP. It is concluded with a summary of the methods described with an explanation of the optimal method that will be applied in this research, 'shape grammar'.

2.1 Historical Documentation

There are very few manuals that have documented proof behind constructing IGP. Construction methods were trade secrets among the craftsmen [6]. However, templates and aides-memoire for pattern generation were recorded on scrolls [17]. Theoretical mathematicians in the likes of Al-Sijzi, Anu-Nasr al-Farabi, Abu'l-Wafa' al-Buzjani, Al-Kashi, Umar al-Khayyami and Abu Bakr al-Khalid al-Tajir al-Rassadi, among many others, wrote manuals on construction techniques that could be applied by the artisan, who formed the IGP [18], [19]. The geometrical texts are described in the next section.

[2] *Prophet Muhammad.*

2.1.1 Theoretical Texts

“Kitāb fīmā yahtāju ilayhi al-sāni‘ min a‘māl al-handasa” (About that which the artisan needs to know of geometric constructions) [20]

A theoretical book written by Abu’l-Wafa’ al-Buzjani, is a ‘how to guide’ for artisans on using the three mathematical tools that were adopted by the Romans, the straightedge (ruler), qonia (set square) and the compass [20]. It is a compilation of Euclidean constructions that are the foundation of generating many of the geometric patterns. It includes [21]:

- Construction and a division of a right angle into equal parts.
- Circle geometry including the bisection of a circle, the division of the circumference of a circle and intersecting the circle with a tangent.
- Creating regular polygons and polygonal figures within the circle.

The circle is used in al-Buzjani’s treatise to generate all of the regular polygons in a plane.” [18]

“Fī tadākhul al-ashkāl al-mutashābiha aw almutawāfiqa” (On Interlocking Similar and Congruent Figures) [20]

This is an anonymous manuscript written between the 11th and 13th century. It is a 20 page long Persian manuscript containing 61 2D repeat unit patterns.

2.1.2 Pattern Scrolls

Alongside the theoretical texts there were architectural scrolls used in workshops for craftsmen and master builders.

Topkapi Scroll

The Topkapi scroll, preserved in the Topkapi Palace Meseum in Istanbul, is an undated architectural scroll [22]. However, it has been presumed to be produced in the 15th century. It is made up of several pages which have been stuck together at both ends. The scroll is approximately 33cm high and 30m long [17]. It does not contain any text, but each figure shows its construction lines. The scroll contains a mix of different types of Islamic Art, from Arabic calligraphy to 2D geometrical patterns. It also contains 2D architectural plans for muqarnas (stalictiles) and domes. The Topkapi scroll is the best preserved of its kind [2].

Tashkent Scroll

The Tashkent scroll is a fragmented scroll preserved in the Institute of Oriental Studies, at the Academy of Sciences of Tashkent. It was a scroll attributed to an Ukbek builder from Bukhara in the 16th century [21]. Like the Topkapi scroll, it also contains geometric patterns for the use of Banna'i (Persian for builder's technique) brick masonry [22].

Both scrolls contain patterns formed from square and triangular grids for brickwork and calligraphy, and polygonal and radial grids formed from concentric circles for 2D and 3D patterns. A detailed description about the theory of grids will be discussed in Chapter 5.

2.2 Categories of IGP

"Star patterns are a harmonious fusion of mathematics, art and spirituality, and expressions of symmetry, balance, and ingenuity."

[11]

Astronomy was a major subject studied by the Arabs. Stars were taken as an object of guidance as it guided one through the desert travels. Furthermore, they were important in guiding one to the direction of prayer that is offered five times a day by a Muslim, to this day. The Qur'an relates:

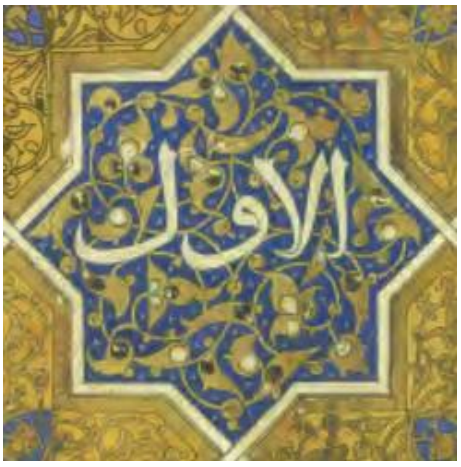


Figure 2.1 8-Pointed Star [8].

"And He it is Who hath set for you the stars that ye may guide your course by them amid the darkness of the land and the sea."(6:97)

[14]

One of the unique characteristics of IGP are the stars that centralize the pattern. Many IGP contain stars which create complexity in the pattern. The rare few without the stars are not as complex and are known as field patterns [10][23]. IGP usually contain 5-, 6-, 8-, 10- or 12-pointed stars, but it is not rare to see odd number point stars like 7- or 9-pointed stars or large number of points such as a 20-point star. However, all stars have the characteristic of being symmetrical as symmetry is an important factor in IGP [10].

IGP can be categorized into *n*-fold categories where *n* stands for the lowest number of rotational symmetry within the patterns. As every regular polygonal shape is derived from a circle, dividing a circle equally into *n* parts forms the *n*-fold. There are three main categories Figure 2.2 [2], [24]:

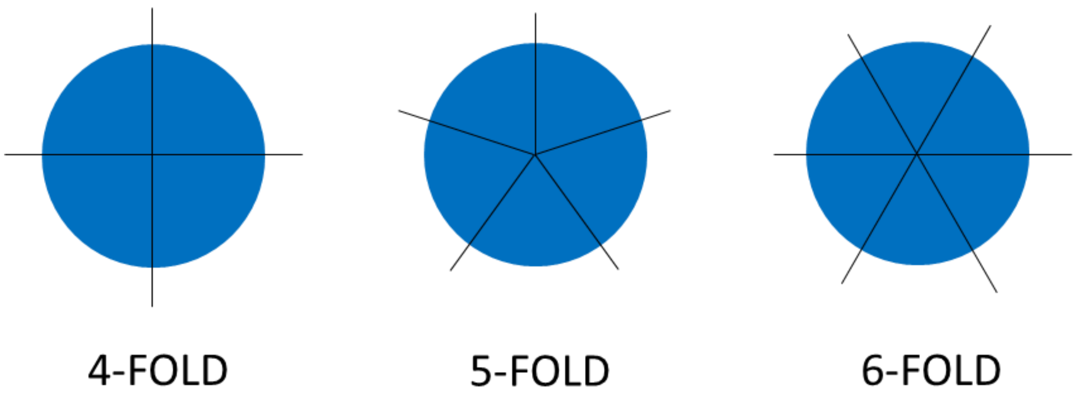


Figure 2.2 Main categories of IGP.

- 4-fold (derived from a square)
- 5-fold (derived from a pentagon)
- 6-fold (derived from a hexagon)

2.2.1 4-fold

The 4-fold patterns have the lowest number of symmetry, 4. These patterns are usually formed from two regular polygons, the square and the octagon, which form common star shapes (Figure 2.3). The 8-pointed stars widely represent the 4-fold category [24]. They are found in all parts of the Islamic world, regardless of the era. The earliest appearance of the 4-fold patterns were formed by the Ghaznavid and Qurakhanid artists, dated to the 11th century [2]. Figure 2.3 shows examples of different types of 4-fold patterns from one monument, Masjid-i-Jami ceiling in Iran.

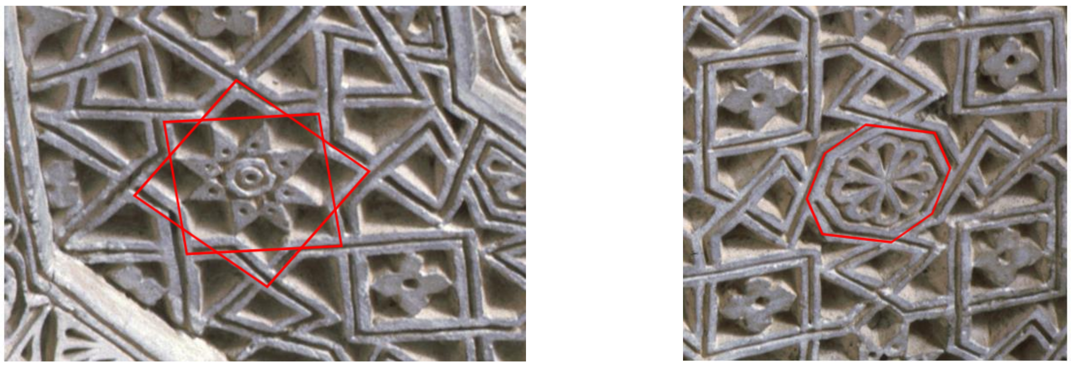


Figure 2.3 The common types of shapes formed in a 4-fold pattern [25].

2.2.2 6-fold

The majority of IGP are 6-fold patterns with a symmetry of 6. The triangle, square, hexagon and dodecagon are regular polygons forming the 6-fold patterns. The common shapes formed with this category are the hexagon or the 6-pointed star. Applying a small change within an IGP construction produces a different and unique IGP. With this in mind, the craftsmen were able to build upon their creativity of 6-fold patterns, taking the example of the variety of 6-pointed stars that can be formed within the 6-fold patterns (Figure 2.4). The common type of a 6-pointed star is the overlapping of two regular triangles (Figure 2.4 (a)). This is an IGP from Al-Salil Tala'I Mosque in Egypt. When the circle is divided into 12 equal parts, one can form 6-pointed stars of the other two types in Figure 2.4 (b-c). Figure 2.4 (b) is from a panel found in Sultan Han, Turkey and (c) from the mausoleum of Sultan al-Ashraf in Egypt.

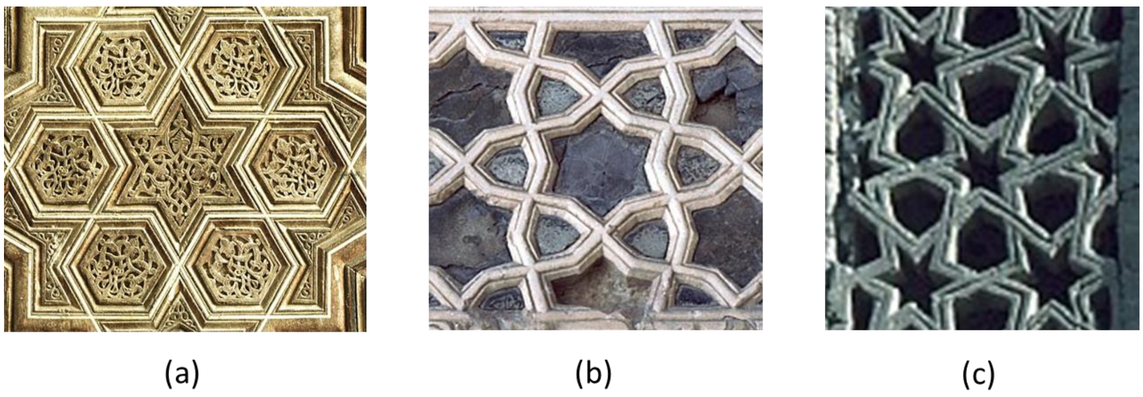


Figure 2.4 The different types of 6-pointed stars [25].

2.2.3 5-fold

All the patterns with a 5-fold symmetry form 10-pointed stars that are created from ether pentagons or decagons. They are a unique category within IGP as they are difficult to replicate periodically just by their individual shape. Overlapping two pentagons can create a 10-pointed star, an original Islamic motif. They are however applied on diverse ornamental mediums throughout history. Extending the points of a 10-pointed star creates kites which forms the rays, creating diverse 5-fold patterns. The earliest example of a 5-fold pattern is the Ghurid Soffit of the Taq-i-Bust arch, found in Bust, Afghanistan (1149 AD). Figure 2.5 shows an example of a 5-fold pattern, found in Imam Mosque in Iran.

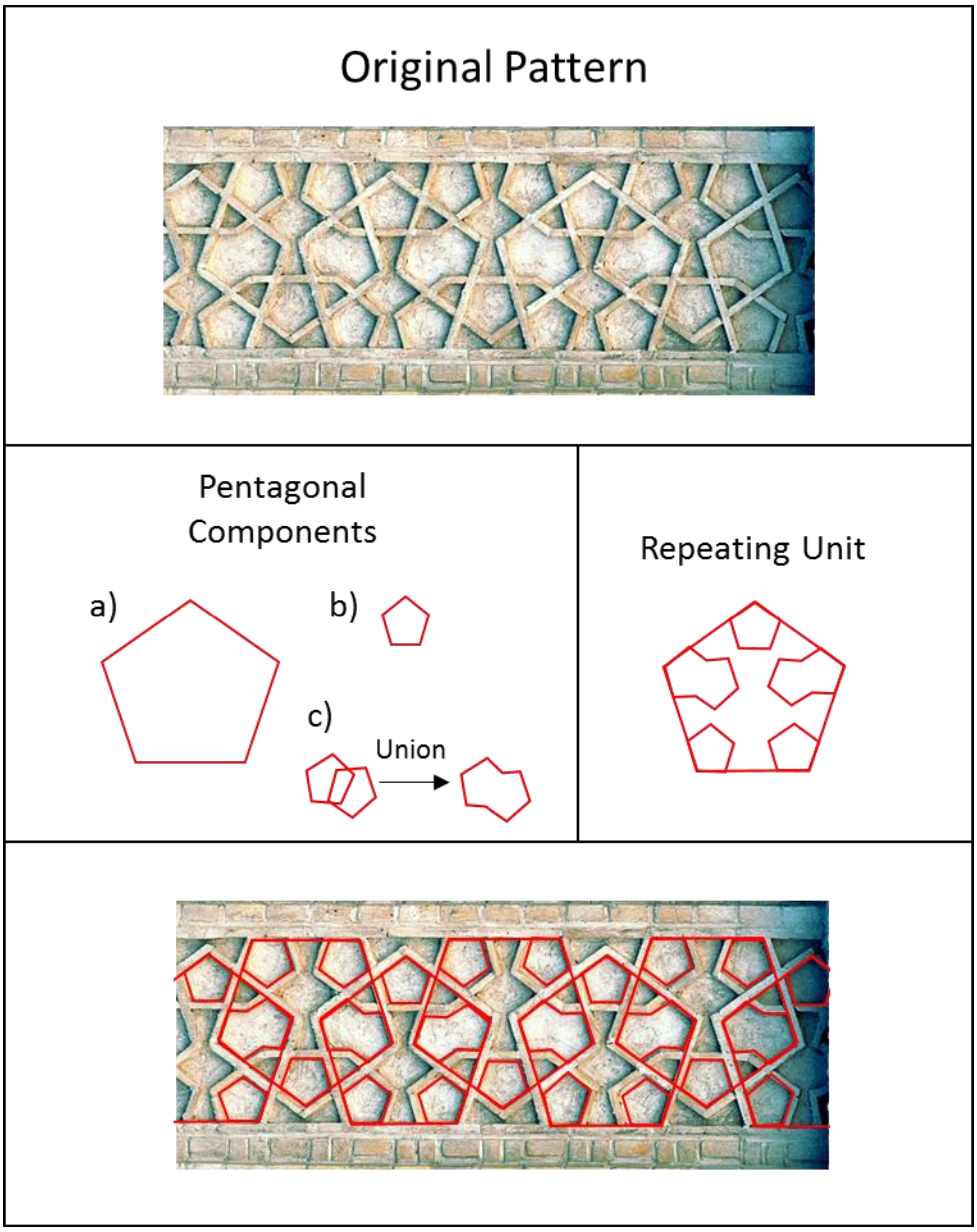


Figure 2.5 5-fold pattern, Imam Mosque in Iran [25].

With the diverse range and categories within IGP, various traditional methods were adopted to construct IGP, these methods are described in the next section.

2.3 Traditional Construction Methods of IGP

The intricacy and complexity of IGP show how mathematicians and craftsmen of that time were very skilled and learned people. The skill to construct an IGP is still however unknown, due to the secrecy of the skill being passed onto the student of the learned only [17]. Although historical documents exist, theoretical books and pattern scrolls, they do not provide a complete 'how-to guide' to form IGP. Many scholars recently have however discovered methods for constructing IGP from mere evaluation of them [11]. Despite not knowing the exact method, the curiosities of the construction methods are still researched upon today. The literature research reviewed the following traditional methods, in no particular order:

- Strapwork method
- Hasba method
- Modular Systems
- Polygons-in-Contact

2.3.1 Strapwork Method

Until recently, research in constructing IGP was proposed by the theory of using the 'Strapwork' method. The Strapwork approach uses two simple tools, a straightedge (ruler) and a compass. IGP are mainly made out of shapes such as triangles, squares, circles and polygons [13]. By using these two tools, one can create these shapes which are then transformed into stars and overlapping lattices [13]. Figure 2.6-Figure 2.8 shows how an IGP is constructed using the Strapwork method in three stages.

The Strapwork method usually forms a repeat unit that can be repeated in a periodic fashion.

- Initiated with a circle, drawn by a compass, creates the central part of the pattern.
- The circle is divided equally by straight lines, using the straightedge.
- The lines are intersected by arcs/circles to develop the pattern, through the artist’s knowledge, forming the basic regular polygons.

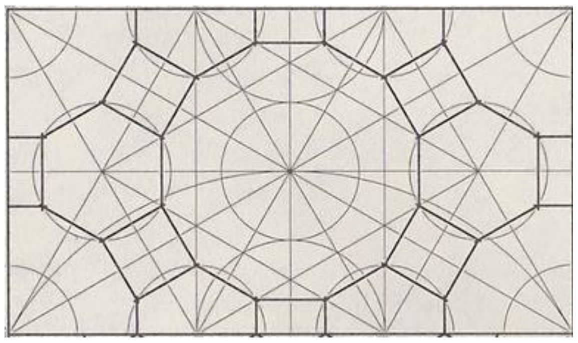


Figure 2.6 Stage 1 of Strapwork method [7].

- The pattern is then formed.

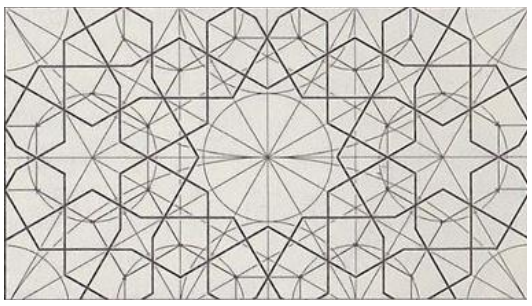


Figure 2.7 Stage 2 of Strapwork method [7].

- The underlying gridlines are removed and the resulting tile can be repeated over an infinite plane or even decorated.

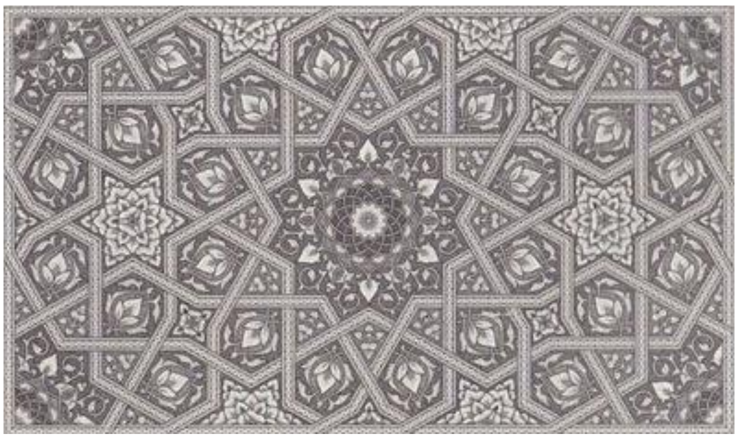


Figure 2.8 Stage 3 of Strapwork method [7].

Although the use of a straightedge and a compass seems relatively simple, the process can become cumbersome and errors can accumulate, for instance angular distortions [26]. Even repairing a damaged section with the Strapwork method can create errors.

2.3.2 Hasba Method

Hasba, meaning ‘measure’, is a method adopted by craftsmen who were situated in the Moroccan region [27]. It is a method used for applying geometric patterns by carving and painting on wood. The Hasba method is known as the measured method as it calculates four important components when creating an underlying grid to form the pattern, Hasba (measure), qassma (empirical unit division), laqtib (ribbon) and zaqaq (alley) [27]. The Hasba (h) defines and initiates the pattern formation, h either being an integer or rational number greater than or equal to 8. The qassma is the unit division calculated from the length of the side of the frame where $L = hq$. The laqtib is a feature with important properties. The width, or thickness, of the laqtib is equal to q; it is created from the gap between the shapes that are formed from the zaqaq area. The laqtib are however only acceptable if they have a continuous journey throughout, with the intersection of two ribbons with an up and under formation, like weaving a thread (see Figure 2.9).

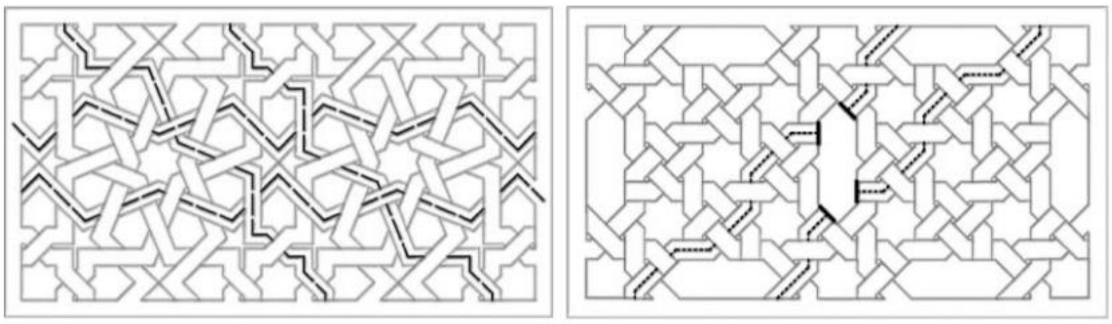


Figure 2.9 The (left) correct and (right) incorrect formations of laqtib [27].

The Zaqaq is the area where shapes are formed to create the pattern; the pattern has a width equal to $4q$. To form the underlying grid one can follow the method described below (see Figure 2.10).

Underlying grid algorithm [27]*:*

- Define Hasba, h.
- Define Qasma, q, empirical unit division.
- Draw a general frame with side length, $L = hq$.
- Draw 8 pairs of concentric circles with diameter $4q$ and $2q$ at the corners and the middles of the sides of the frame.

- Draw parallel lines that cross the centres of the circles and tangents to all the circles.

The pattern is then formed by drawing the repeating template (Figure 2.11 left) and then later applying the point group symmetry, $4m$, (Figure 2.11 right).

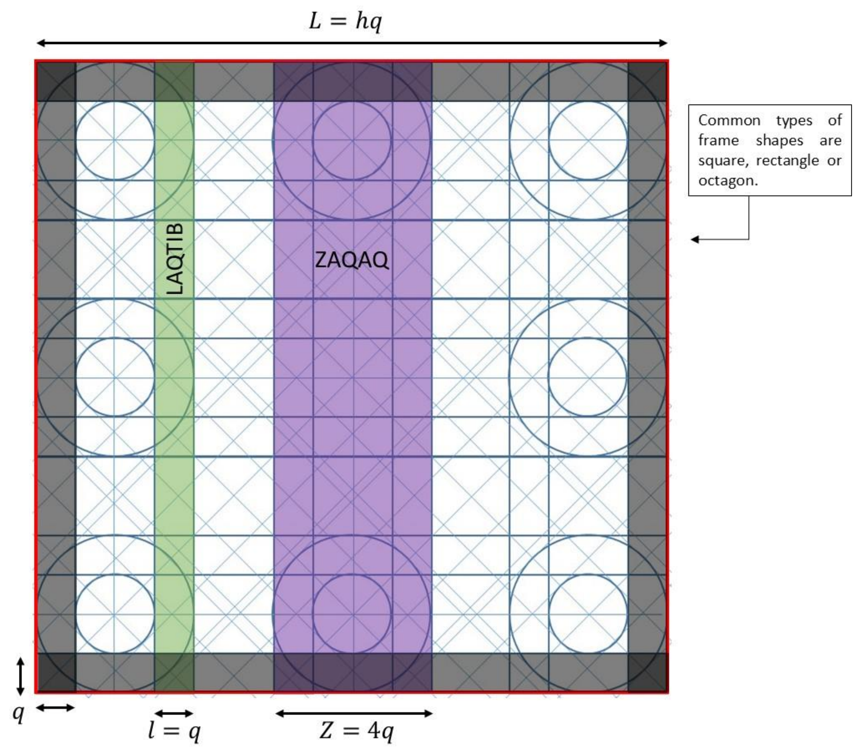

$L = hq$
Common types of frame shapes are square, rectangle or octagon.
LAQTIB
ZAQAQ
q
$l = q$
$Z = 4q$

Figure 2.10 Hasba Method, underlying grid formation (Reproduced [27]).

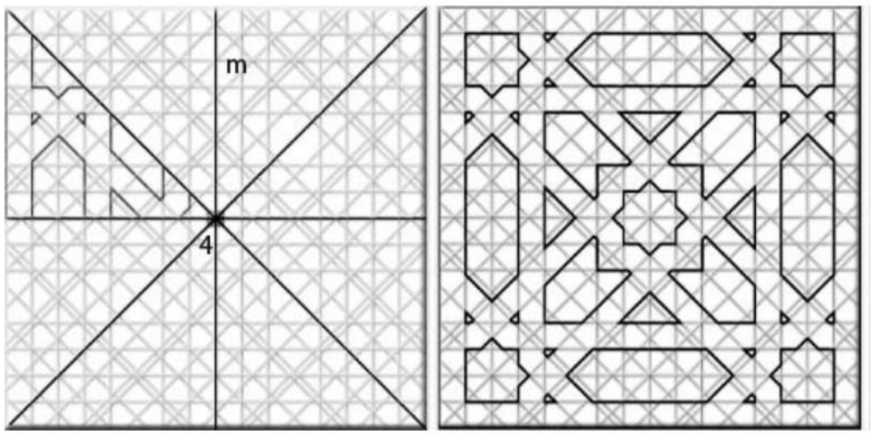


Figure 2.11 (Left) Repeating template (right) complete pattern [27].

With the same underlying grid, which was formed from $h = 16$, a variety of patterns can be formed (Figure 2.12).

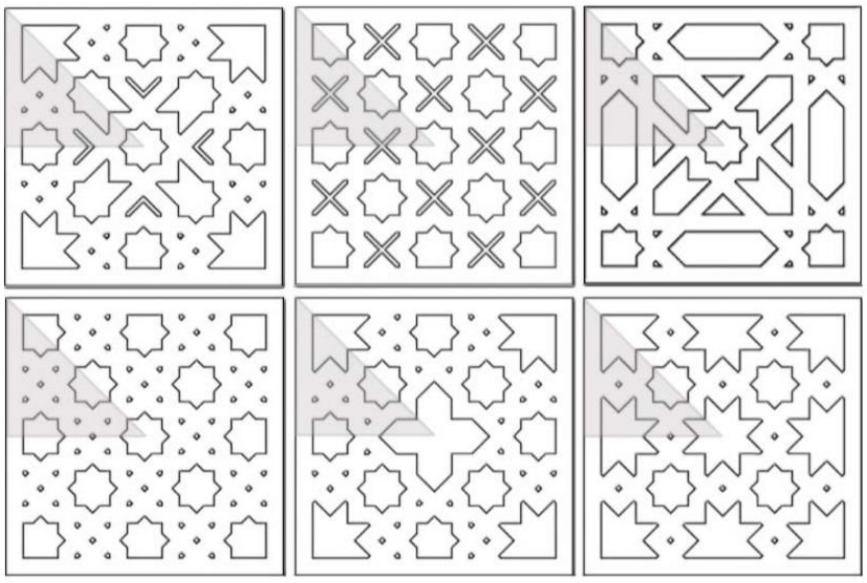


Figure 2.12 Variation of patterns formed from same underying grid [27].

Although this method is strategically calculated and the repeating template helps in the pattern formation, it is a long process. Identifying shapes within the grid, like the Strapwork method, is difficult for someone who is new in creating IGP.

2.3.3 Modular Systems

A modular design is a pattern made up of smaller sets of modules or shapes [23]. Bricks are one of the simplest forms of modular systems; they are arranged periodically, with just one module, a brick [23]. Modular systems are known to be one of the famous traditional methods of creating IGP in the western part of the Islamic world [23]. They are usually placed to create a square or rectangular motif, creating a tessellation when repeated periodically. There are many benefits in using this method; firstly, the ease of producing a large number of motifs and secondly, the simplicity through the use of a set of modules. This simplicity also has an aesthetic appeal to the viewer [23].

Zellij Tiles

One of the main modular design systems in Islamic Art is the ‘Zillij’ tile work. It is one of the distinctive forms of Islamic art in Spain and Morocco [5]. The tile work is created by small bright coloured hand-cut tiles known as ‘furnah’ [5], bringing it into the mosaic category. They are cut from large square pieces of enamelled terracotta [10]. The number of furnah is unknown to this date, but there are approximately 360 furnah in common use today [5]. Every Zellij pattern is made up of sets of furnah (Figure 2.13 left) and are transformed accordingly with simple Euclidean transformations [5].

The Zellij tile work has no distinctive generative procedure, due to secrecy of this art form. However, one approach is described by Abas [6] by means of the geometric study of Zellij patterns [5]. The craftsmen of Zellij patterns, known as Mallerns, are known to make use of a piece of graph paper to design their patterns (Figure 2.13 right), using the grids as a guide [10]. It is a technique of packing space, hence leaving no gaps or holes, similar to completing a jigsaw puzzle [10].

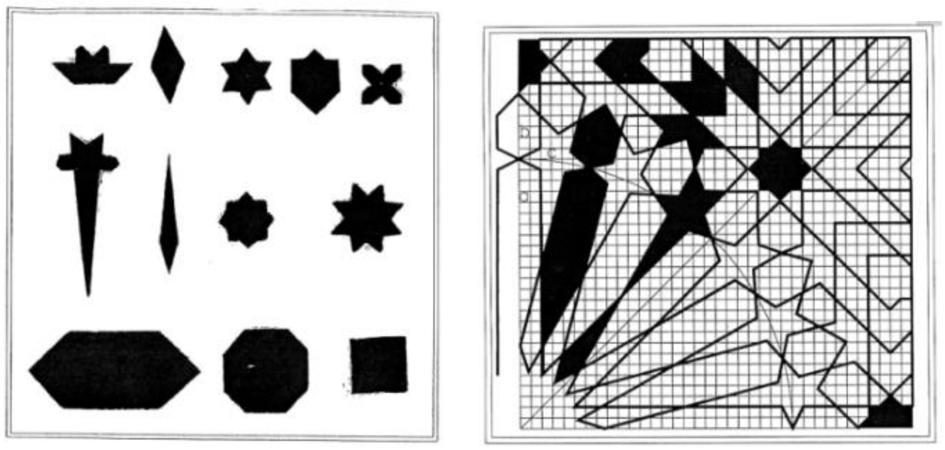


Figure 2.13 (Left) A Set of furnah. (right) Initial construction of the Zillij design [6].

Girih Tiles

Girih patterns, interwoven IGP, are complex geometric patterns. In 2007, Peter Lu and Paul Steinhardt proposed the use of five polygonal tiles to construct the Girih patterns [26]. They named the tiles, ‘Girih’ tiles; Girih meaning interlock in the Persian language [28]. The five polygonal tiles are a rhombus, a bowtie, a bobbin, a pentagon and a decagon (Figure 2.14) [26].

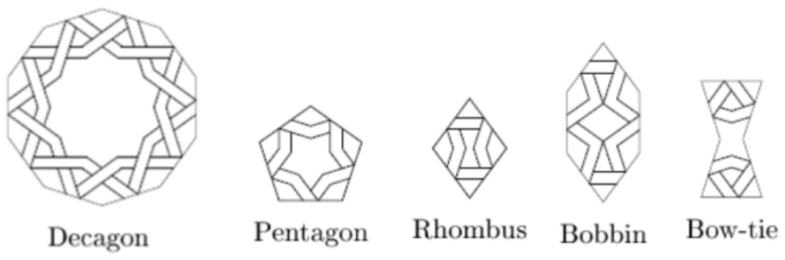


Figure 2.14 Girih Tiles with decorated lines [26].

The Girih tiles are special in the way that all the edges of the five different polygonal tiles have the same length [26]. Furthermore, they have decorating lines printed on them, creating a continuous pattern when joined together [13]. For the pattern to flow continuously, the mid-point of every edge is intersected with the decorating lines at 72º and 108º [26]. The pentagon has five-fold symmetry and the decagon ten-fold symmetry, leaving the bowtie, bobbin and the rhombus all with a two-fold symmetry. Hence, this tiling is not periodic due to the 5-fold and 10-fold symmetries by the pentagon and decagon. Due to being non-periodic, the tiling is connected through decorating lines, causing the IGP to flow continuously without the need of being periodic. Girih tiles can form a large variety of decagonal motifs, which do not appear naturally through the Strapwork method [26].

The use of Girih tiles creates minimal errors and speeds up the process of creating a complex tiled plane [26]. Although the Girih tiles are only used as a template purpose, the transformation from tile to design is very rapid. Figure 2.15 shows how the Girih tiles are transformed into the complex IGP [13].

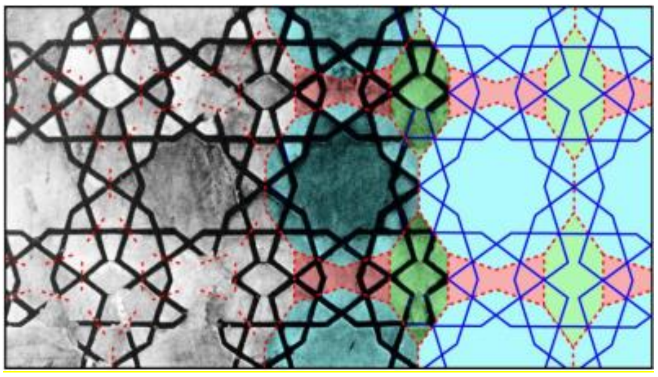


Figure 2.15 The Girih tiles (right) transformed into design [26].

2.3.4 Polygons-in-Contact

Polygons-In-Contact (PIC) is another method for constructing IGP. It was first initiated in the West by E.H.Hankin [11], [29], who discovered a polygonal grid under one of the original IGP on a momunent in India [29].

This method uses a network of polygons, which acts like a base to generate the IGP; the polygons are connected by their edges to form a polygonal network. Regular tiling and semi-regular tiling usually make up the polygonal networks. The IGP are then generated within the network base.

The polygonal networks are simple in the way that the central stars can be constructed easily. The stars that are constructed inside the polygons produce a base for extending the IGP [11] to form both simple and complex patterns. After the completion of the IGP, the underlying polygonal network is removed to present the final pattern [30]. The PIC method produces a variety of patterns but it is not an approach that can be applied universally throughout every IGP that is historically documented [29]. The PIC approach limits one to create only Islamic Geometric ‘Star’ Patterns. It is however the only method that has documented proof and is used widely amongst many artisans in the Islamic world to this day [30].

The IGP that are generated using the PIC method can be categorized into four categories as devised by Bonner [30]:

1. Accute
2. Obtuse
3. Middle
4. Two-Point

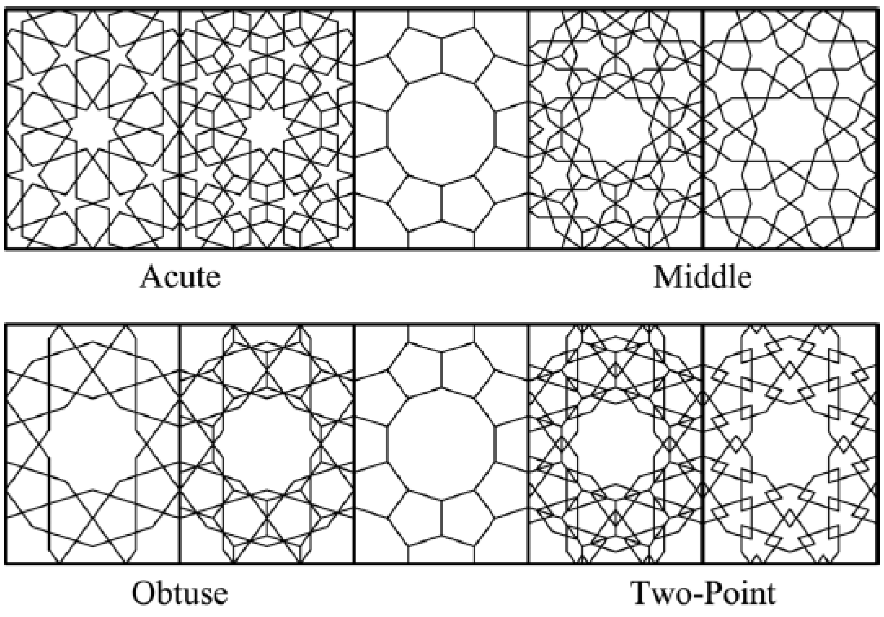


Figure 2.16 Star Patterns generated from polygonal networks [30].

The naming convention is determined by varying the angle of contact on the mid-point of each edge of the polygon on the polygonal network. An example of how a single polygonal network can produce various IGP is shown above, Figure 2.16. The accute pattern has a crossing angle of 36º, the obtuse with 108º and the middle being 72º, as it is in between 36º and 108º. The two-point pattern is derived from using two points, a given distance from the mid-point on each polygonal edge rather than just the one mid-point.

Both the strapwork and hasba methods form IGP by the initiation of straight lines and or circles and are built systematically by finidng the point where the line intersects with the other line and or circle. The pattern is formed using these two elements of shapes. As there are a vast amount of IGP with formed of various polygonal shapes to form a generalized algorithm would be complex as the system would need to calculate the correct position of where the line and or circle needs to be placed with correct intersections. The PIC method is simpler than the strapwork and the hasba method as the IGP is formed from a grid of polygons. However this method also applies the use of intersections with lines. The final method, Modular systems, is a method that generates the pattern with a set of shapes. As the objective of this research is to form IGPs with a single polygonal shape, modular systems is the closest.

2.4 Computerized Construction Methods of IGP

Computer generating traditional geometric patterns were initiated by Lalvani [31] who presented a method to form different kinds of traditional patterns by the use of coding. The method formed the central fundamental unit of the given pattern by applying the rules of symmetry, which was multiplied to form the pattern.

Aljamali and Banissi [32] applied the Lalvani approach of generating computerized IGP, but defined the central unit of the pattern by controlling the radius of the shape and the angle of rotation. The method forms the pattern by sub-motif grid based patterns, where a part of the central motif is analyzed and reproduced by symmetry rules.

Kaplan [11] derived a computerized algorithm by integrating the PIC method to form automated 2D IGP. The algorithm determined the mid-points of all the edges of the polygons on the given polgonal network, placing a 'X' at each point. This 'X' was then grown to create 'arms' which were extended to the point where the the pair of 'arms' were in contact with another set of 'arms' (Figure 2.18, left). This then formed the pattern. The pattern was visible after the removal of the polygonal network, as shown in Figure 2.18, right).

One advantage of this algorithm is the user can change the 'contact angle', (the angle of the 'X' at the midpoint), (Figure 2.17 (a)) therefore determining the possibility of creating a variety of IGP. It used the approach of Bonner's conventional IGP categorization, creating different patterns from a single polygonal network, for the pattern to be known as Islamic. Figure 2.19 shows how the mid-point on the Archimedian (semi-regular) tiling is found, and how the contact angles are derived, to form the acute, middle and obtuse patterns. The two-point is formed by calculating a distance, given by the user, from the mid-point on the polygonal edge, (Figure 2.17(b)), showing two contact points, creating two 'X' points on the polygonal edge.

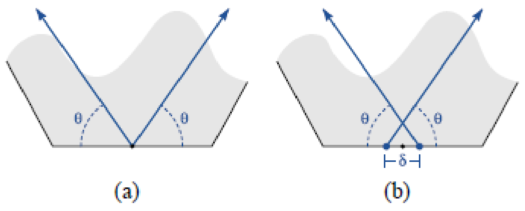


Figure 2.17 Contact points [11].

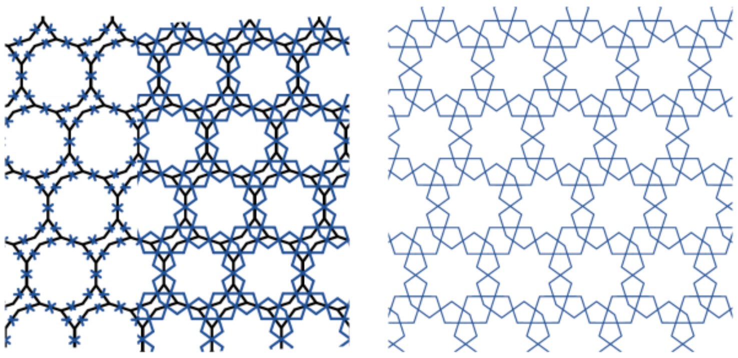


Figure 2.18 (Left) Mid-points on polygons (right) corresponding pattern [11].

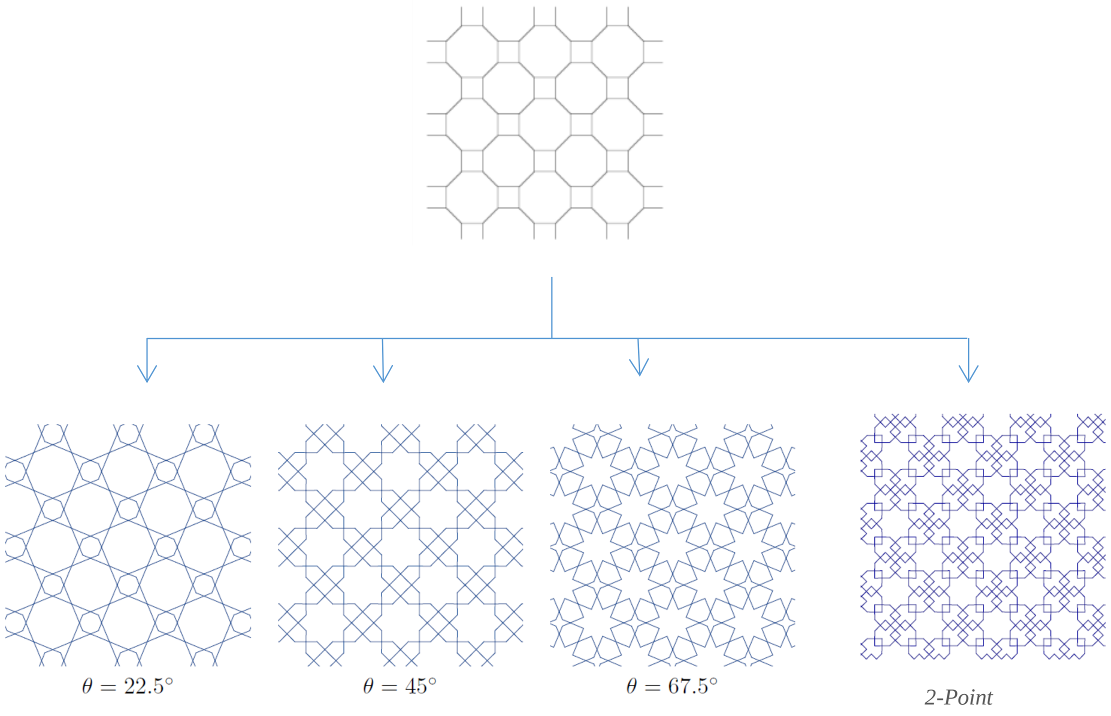


Figure 2.19 (Top) Underlying grid. (Bottom) IGP generated by varying contact angles [11].

2.4.1 Shape Grammars

Shape grammars were initialized by Stiny & Gips [33]. It is a method based on shape rules [4]. This method is used to create both original and new designs. Stiny devised shape grammars as mathematics, describing the final design formation as a calculation of shapes and rules, hence calculating them in algebras of shapes [3].

A shape grammar is defined by the following 4-tuple: SG = (V_T, V_M, R, I) [33] where:

1. V_T is a finite set of shapes. Its elements are known as terminals.
2. V_M is also a finite set of shapes but the elements here are non-terminal or markers such that $V_T^* \cap V_M = \emptyset$. V_T * is a set of elements formed by the arrangement of either an element or elements of V_T.
3. *R* is a finite set of ordered pairs *(u,v)* where *u* is a shape made of an element of V_T * and V_M and *v* is a shape made of either:

    a) An element of V_T * from *u*

    b) A combination of an element of V_T * from *u* and an element of V_M

    c) A combination of an element of V_T * and V_M with an additional element of V_T * from *u*.

    The elements of *R* are known as shape rules.

4. *I* is the initial shape which is made up of elements of V_T * and V_M.

An initial seed shape forms the final design [5] and can be anything from floral to straightedge. The rules however are the key to forming the end design. Varying the rules on the same initial shape forms various different designs. The amount of times the rules are used varies the complexity of the final design, which causes *emergence* in designs [3].

Emergence is a key characteristic in IGP. It can be defined as emerging shapes, which are identified either during a pattern generation or when the final pattern is generated [5]. These processes are known as ‘Emergence as process,’ and

‘Emergence as product’. These processes can be further divided into types of emergence; Gross [34] discovered three main types of emergence:

1. Intersection of two or more shapes to create an unintentional shape with the intersection marks.
2. Alternative configurations where a part of the shape is viewed in a different way to what it was initially perceived as.
3. Figure ground reversal, which creates a shape from the edges of the surrounding shapes.

Every traditional construction method explained above has an emergence type. Starting with the Strapwork method, produced by a straightedge and a compass, the emergence is shown during the process of creating the pattern whilst the straight lines intersect with the circular arcs. Hence, the most prominent type would be type 1 [5], intersection of shapes as shown in Figure 2.20. This also occurs in the Hasba method as the lines either interact with each other or with the circle.

Modular design systems involving Zilij tiles, Girih tiles and other modular systems, create an ‘Emergence as product’ structure [5]. As the designer is limited to only a set of tiles, type 3, figure-ground reversal, is the most suitable as edge shapes are created when joining the tiles together (see Figure 2.21).

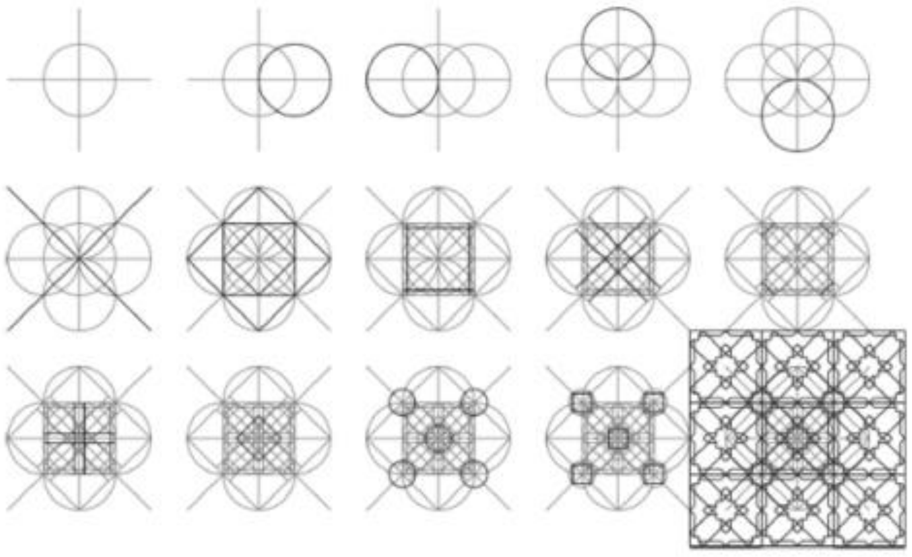


Figure 2.20 Compass & ruler construction of an Islamic geometric pattern [5].

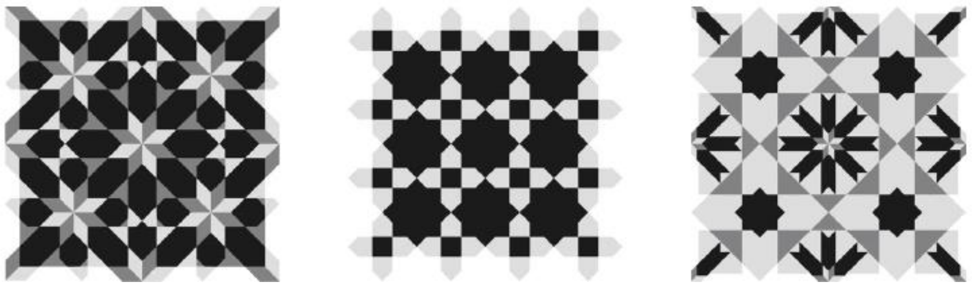


Figure 2.21 Examples of patterns generated by Modular systems [5].

Emergence is also generated in the process of the PIC method. Each polygonal edge of the polygon on the polygonal grid is intersected in the middle with a mark 'X'. Hence type 1, intersection of lines, is the most relevant.

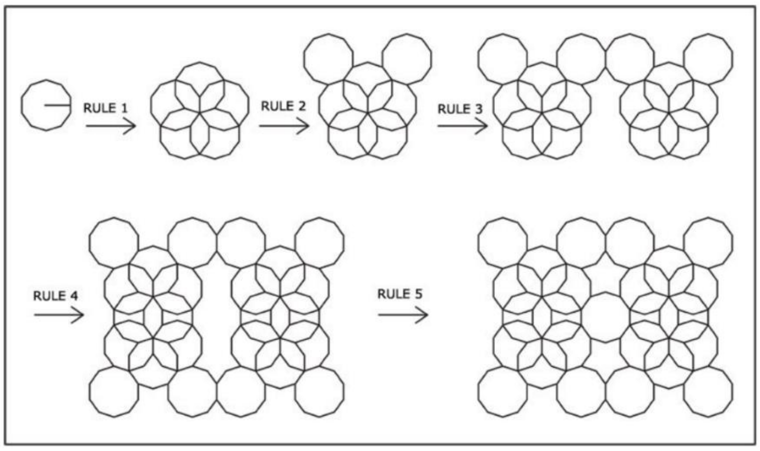
Lastly, the shape grammar method can be defined as a "..two stage process" [34]. First stage would be that of the construction of the motif or star and secondly, for it to be repeated under symmetry transformations. The star pattern here is an 'Emergence as product', but it would come under type 2, alternative configurations, because of the unlimited ways it can viewed.

Shape grammar allows transformation rules consisting of Euclidean transformations (translation, rotation and reflection), scaling proportionally, emergence rules and Boolean operations. Emergence rules are applied to the designs generated from previous shape grammar rules [35]. The use of the rules depends on the designer and how they create the grammar and arrange the rules. However, the rules designed are the only rules that can be applied throughout the design formation. Shape grammar acts as an algorithmic system; it creates its algorithm automatically with the rules designed by the designer [5].

Shape Grammar Implementation in Pattern Generation

Shape grammar has been applied in various areas, such as architecture, historical ornaments, product design and many works in pattern generation [4]. Ismail et al. [35] implemented shape grammar in generating Songket designs using a 'Bunga Cabit' (a traditional Malay Songket motif), using both Euclidean and emergence rules. Repeating the rules many times in a different order thus creating various patterns.

Additionally, shape grammar has been implemented in designing IGP. Ulu & Senar [4] produced IGP from using a decagon as an initial shape. However, the generated pattern produced a base grid for IGP (Figure 2.22 top). The final pattern was formed using Lu & Steinhardt's Girih tiles and Prof. Metin Arik and Mustafa Sancak's covering tiles, (Figure 2.22 bottom). The derived pattern is in Appendix B with the decagon design template covered with the three tile coverings.



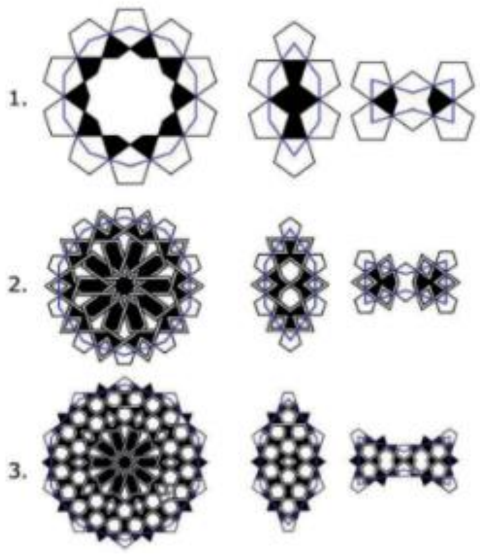


Figure 2.22 (Top) The generated IGP design template with a decagon. (bottom) Three groups of covering tiles of the decagon, tie and the bowtie [3].

Jowers et al. [5] described the three construction methods for IGP, one being shape grammar. By analysing eight different IGP corpuses (motifs), they came to the conclusion of using a triangular structure as an initial shape, as all the corpuses had an eight-fold rotational symmetry. The rules designed produced the inner content of the motif first (Rules 1-8), and then took the use of emergent rules (Rules 9-10) (Figure 2.24), for replicating under symmetry transformations. The grammar was formed of the following rules as shown in Figure 2.23.

Rule 1 – Addition of the grid.

Rules 2-6 – Line elements of the motif by recognition and replacement of grid lines.

Rules 7-8 – Removal of grid lines and their associated end points.

Rules 9-10 – Symmetry transformations where the ‘F’ represents a generated motif.

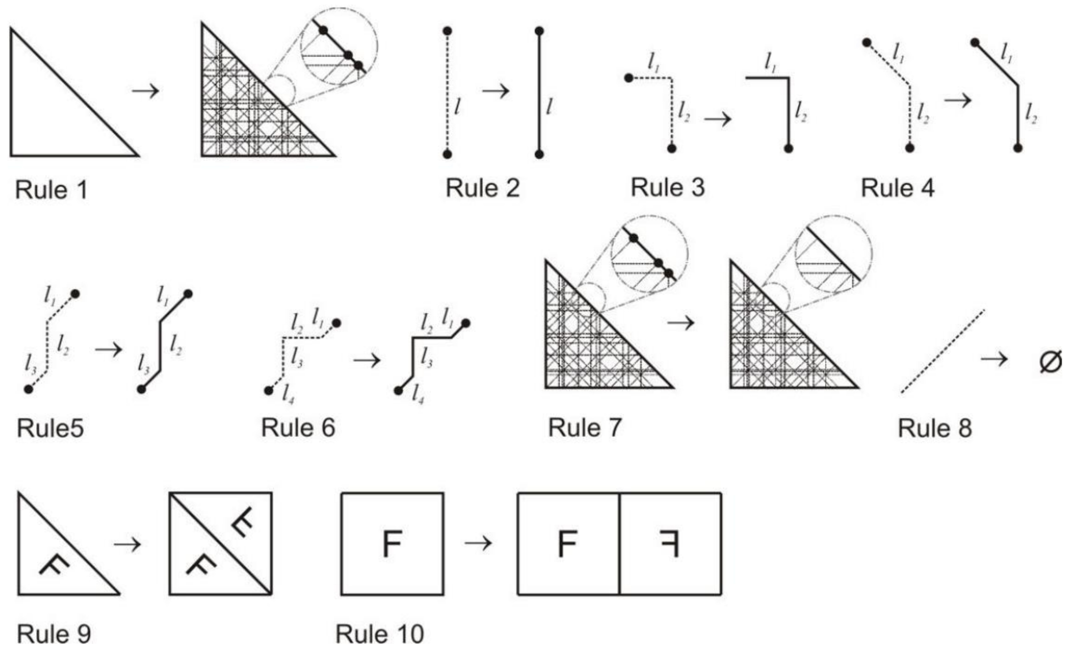


Figure 2.23 Grammar rules.

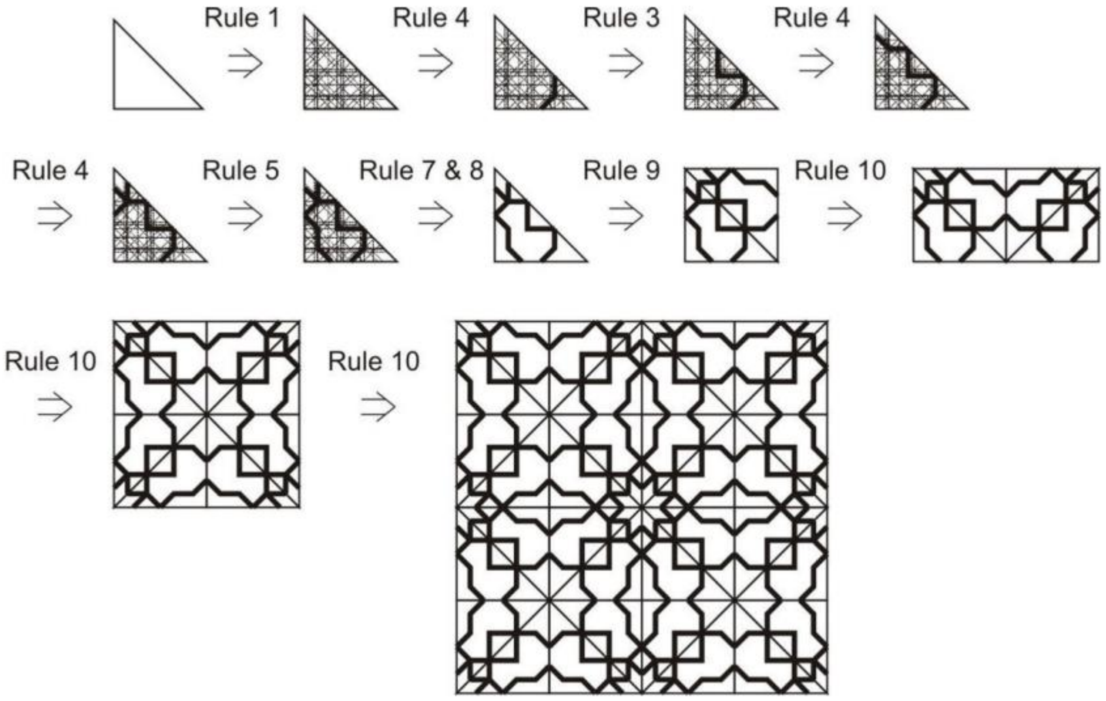


Figure 2.24 Shape Grammar construction of an Islamic geometric pattern [4].

However these rules are not generalised to create any type of IGP as analysing another set of corpuses would form a different set of rules within the formed grammar.

Shape grammar and architecture

Shape grammar has also been implemented in the fields including architecture widely [36]. The table of implementation of shape grammar can be seen in Appendix C [37]. The main contributions of shape grammar in architecture are analyzed below.

The Palladian Grammar [38]

The Palladian grammar is formed from a parametric shape grammar which was designed by Stiny. It creates the 2D ground architectural plans of Palladio's villas that are defined by the Palladian style. The grammar is formed of two stages, the initial stage of creating the basic ground plan and latter of the details within the plan. The complete definition of the Palladian grammar is not governed within the grammar, hence a partial plan of the villa is generated.

The Grammar of Paradise [39]

Stiny has formed the grammar of paradise which produces Mughal garden plans, examples are applied from four corpuses that include Indo-Pak char-bagh (four gardens). The char-bagh are formed from the geometry of a garden, site parti, canals, pathways and borders. These 4 elements of the garden are formed individually with the use of a parametric shape grammar. The char-bagh is formed of a 2D architectural ground plan.

Frank Lloyd Wright's Prairie Houses [40]

The Frank Lloyd grammar is formed of a parametric shape grammar that has an initial shape of a 3D shape but the transformations are in 2D. Again this grammar is formed of two main stages, the basic outline of the house, which is split into five different features that are separately formed with the fireplace being the focal point of the grammar. The latter stage creates the ornamentation of the basic outline which includes porches, terraces and balconies.

Grammar of Queen Anne houses [41]

The grammar of Queen Anne Houses once again does not define the entirety of the design. Only a few parts of the Queen Anne house were applied as the corpus of the grammar formation. Splitting the grammar into two stages, the interior and exterior, this grammar actually is formed in two different dimensions. The interior is a 2D plan, forming a 2D shape grammar and the exterior is extrusion of the 2D shapes within the plan which creates the 3D buildings. This is the only grammar which is formed in two different dimensions; the other applications all form in one dimension, either 2D or 3D.

Vitruvian Shape Grammar [42]

This grammar is based on the writings of the Roman architect Vitruvius. The rules of the grammar form the procedural modelling of the Vitruvian temple. Different elements within this grammar create the complete temple including the base, cellar, stairs and columns. Although the grammar forms a complete 3D model, its transformation is with a 2 dimensional axis.

From all the examples described above it can be seen that none of the grammars are actually applied as a mapping technique, and each of the architectural plans or buildings are formed in 2D space, with the use of the translation transformation as the most common rule adapted.

2.5 Summary

As it has been shown in this chapter, IGP are a diverse group of patterns that are categorized into different groups, 4-, 5- and 6-fold patterns. From all the traditional methods described none can construct all the types of IGP that are archived up to date [24], [25]. The computerized method that was formed by Kaplan [11] only creates star patterns, which singles out the small category of field patterns. Although shape grammar has been applied to create original and new IGP [4], [43], [5], the methods either form a tiling base, where a set of modular tiles can be placed or the rules are not generalized.

Hence none of the pattern generated works described above can create an IGP which has generalized shape grammar rules. As the initial aim of this research is primarily to form 3D IGM, shape grammar is the optimal method. Shape grammar is formed of shapes and not symbols hence it allows one to visually create the patterns step by step. The addition of depth, the third dimension, which was lacking in the previous construction methods can be implemented within shape grammar.

3. Mapping Techniques

“Colours speak all languages.”[3]

3.1 Introduction

A mapping function that applies a texture onto the surface of a 3D object or surface is known as texture mapping [44]. The texture map can be either 1D, 2D or 3D and are represented by an array or mathematical function [45]. For example a rock strata represents a 1D map, waves and surface bumps are 2D and clouds, wood or marble represent 3D texture maps. Texture space is known as the coordinate system that represents texture data, usually spanning from a range between [0,1] [44]. The coordinates (u, v, w), denotes the texture coordinates in texture space. To convert from object space coordinates (x, y, z) to texture coordinates the tangent space of the given surface is calculated and normalized. For a 2D parametrized surface, $\vec{S}(u, v)$, S: $\mathbb{R}^2 \rightarrow \mathbb{R}^3$, [46],

$$\vec{T}(u,v) = \frac{\partial S}{\partial u}\ ,$$

$$\vec{B}(u,v) = \frac{\partial S}{\partial v}\ , \tag{3.1}$$

$$\widehat{N}(u,v) = \frac{\vec{T}(u,v) \times \vec{B}(u,v)}{\left\|\vec{T}(u,v) \times \vec{B}(u,v)\right\|}\ .$$

[3] *Joseph Addison.*

The tangent, $\vec{T}(u,v)$, bitangent, $\vec{B}(u,v)$, and the normal vectors, $\hat{N}(u,v)$, form the tangent space [47]. The conversion matrix is obtained from normalising the tangent and bitangent [46],

$$\boldsymbol{TBN}(u,v) = \begin{matrix} \vec{T}(u,v) \\ \vec{B}(u,v) \\ \hat{N}(u,v) \end{matrix} . \tag{3.2}$$

The following chapter describes the definition of a mesostructure, the detail that is formed on a surface. Also the different types of textures that form mesostructure and the techniques of storing them through texture maps, procedural methods and actual 3D geometry. Volumetric mapping methods are then described in detail with a survey of the implementation of the shell mapping method.

3.2 Size of Geometry

Geometric scale can be categorized into three categories [48]:

- Macrostructure
- Mesostructure
- Microstructure

The largest geometric object is known as a macrostructure. It is usually represented by 3D primitives for example a cube, cone, sphere etc. or surface patches [49]. Macrostructures are the geometric type that apply fine scale detail on their surface. The fine scale detail are known as texture maps, 2D or 3D, that are stored in texture space, and these represent mesostructures [46]. Mesostructures are high frequency detail that can be distinguished clearly when seen at a close distance. The size of a mesostructure is larger than a pixel. When the geometric detail is less than 1mm in size, this type of geometry is a microstructure. Microstructures are invisible to the human eye, hence their implicit representation is through light scattering properties formed by 'Bi-directional Reflectance Distribution Reflectance' (BRDFs) [50]. Figure 3.1 shows examples of the three categories of geometric scale in terms of realistic rendering within computer graphics, with approximate scale ranges. The geometrical size applied in this research will be based upon mesostructures as they form the surface detail.

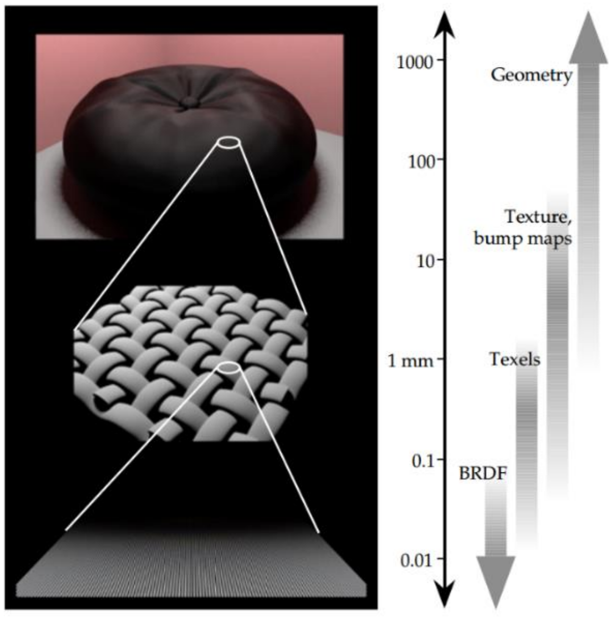


Figure 3.1 Different categories of geometric scale [48].

3.3 Representation of Surface Detail

Texturing on a 3D surface, on a mesostructure level can be categorised into three techniques; 2D textures, displacement mapping and 3D textures, all of whom represent surface detail [51].

3.3.1 2D Textures

The most common technique applied in CG to texture a surface is through 2D colour [52] or bump maps [53] . These two types of mappings are found in many of the 3D software packages. A colour map, sometimes referred as a diffuse map, defines all the surface colour which does not include any lighting or shading information.

A bump map is a greyscale image that acts as a pseudo height map [53]. The map does not create any additional geometry but creates an illusion of surface detail, with either raising or lowering the surface. For a lowered detail representation, the greyscale image is darker than 50%, and lighter parts represent heights [54].

A normal map is another type of 2D texture map. It is a map that indicates the direction of a surface normal [55]. These maps can affect the lightings within a scene which again creates an illusion of surface detail. A normal map applies the information of RBG, which corresponds directly with the texture space co-ordinates (x, y, z). There are two different types of normal maps; tangent space normal maps and object space normal maps. A tangent normal map is relatively common to the latter and is applied on deformable objects which is made up of a mixture of blues and purples. On the other hand an object space normal map is equipped for non-deformable surfaces, which are formed of a rainbow assortment.

3.3.2 Displacement Mapping

Displacement mapping, introduced by Cook [56], is a type of texturing technique that actually adds fine scale detail to the surface in contact to the illusion of bump and normal maps. The points of the geometry of the surface move or displace forming a sense of depth and detail [57]. This is applied through either height or vector fields. Height fields, or height maps, are formed by displacing the geometry points towards the surface normal. With the height field, $h(u,v)$, the point in object space would be calculated as [46],

$$\vec{p}(u,v) = \vec{S}(u,v) + h(u,v)\vec{N}(u,v). \tag{3.3}$$

The other type of application, vector field, is where the geometric point is displaced to an arbitrary direction. The surface displacement point can be calculated in object space, with the given vector displacement $\vec{d}(u,v)$,

$$\vec{p}(u,v) = \vec{S}(u,v) + \boldsymbol{M}(u,v)\vec{d}(u,v), \tag{3.4}$$

where the matrix $\boldsymbol{M}$ converts the vector displacement from texture to object space or vice versa.

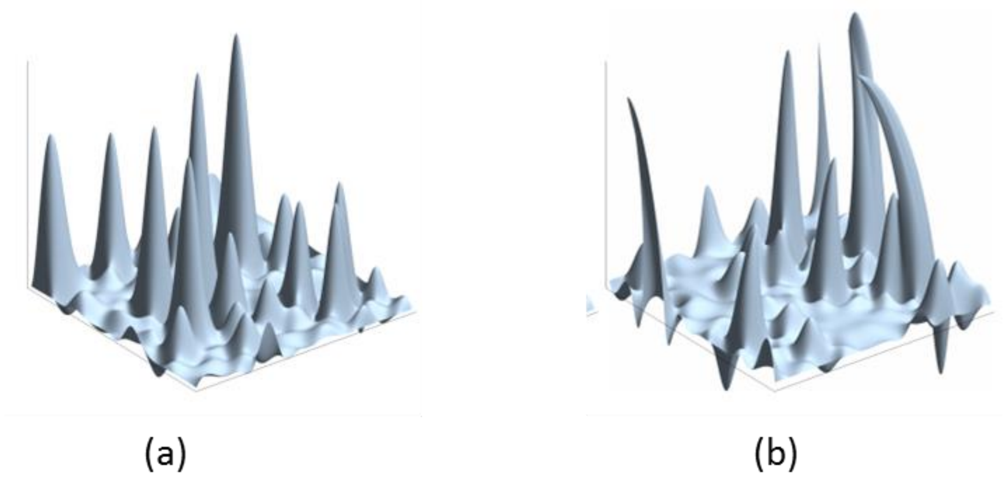


Figure 3.2 (a) Height fields (b) Vector fields [46].

Two types of displacement mappings that are commonly applied are 'Parallax' and 'Relief' mappings.

Parallax mapping

Kaneko et.al [58] introduced parallax mapping, a mapping that displaces the individual pixel heights of a surface. When the surface is viewed from a particular angle within a scene the high points on the surface conceals the low points allowing it to have a 3D effect. The height of the surface points that are displaced is formed through the greyscale height field. Although parallax mapping is cost efficient, the quality is poor. This sort of mapping works well on curves that allow a few centimetres of height, like texturing a brick or stone wall. There are different versions of parallax mapping such as coon step [59], steep parallax [60] and iterative parallax [61].

Relief Mapping

Relief mapping is a root-finding approach on a height field, introduced by Oliviera .et.al [62]. The process initiates with a viewing ray that is transformed into texture space and then locates an intersection on the given surface by performing a linear search. A binary search follows on from the linear search to locate a precise intersection point. A disadvantage however with the liner search is that a fixed step size is required, which is a process to create fine detail by increasing the number of steps to do so. Hence this accounts for accuracy but is poor in regards to performance.

3.3.3 3D Textures

3D textures can be categorised into three categories [63]:

- Volumetric textures
- 2.5D textures
- Dynamic textures

Volumetric textures

Volumetric textures are solid textures which define texture densities in a volume, $V_{x,y,z}$. They exist as a volumetric object in the form of, $\{V: x, y, z \in V_{x,y,z} \subset \mathbb{R}^3\}$.

A volumetric texture is usually generated by a data acquisition device, for example tomography or confocal imaging, mainly applied in the medical field.

2.5D textures

Textures that are placed on surfaces of 'hollow' objects or open surfaces are known as 2.5D textures which are represented as: $\{T: x, y, z \in T_{u,v} \subset \mathbb{R}^3\}$. 3D geometry is added onto a surface as in Kajiya and Kay [64] , Neyret [65] and Porumbescu et.al [66], through various storage techniques (explained in the next section).

Dynamic textures

Dynamic textures introduced by Polana and Nelson [67] are fluid motions, for example motions of rivers, foliages, flames, etc. They are represented in the form of 2D time sequences as $\{S: x, y, t \in S_{x,y,t} \subset \mathbb{R}^3\}$. A dynamic texture is usually viewed as a 3D data cube where the individual slice cuts preserves the texture motion. For example Figure 3.3 shows parts of two dynamic textures, where each part is sliced from the origin voxel $O(x, y, t)$, creating three planes $(\vec{x}\ O\ \vec{y})$, $(\vec{x}\ O\ \vec{t})$ and $(\vec{y}\ O\ \vec{t})$.

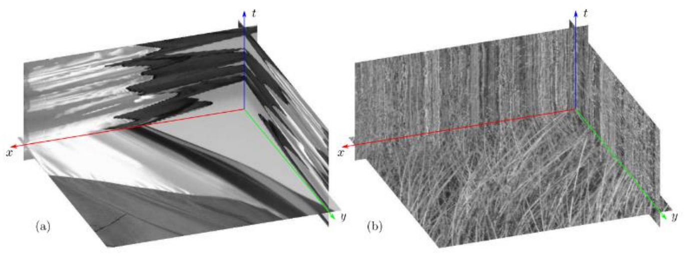


Figure 3.3 Dynamic texture of (a) flappig flag (b) grass [68].

As this research is on creating a mapping technique to map 3D geometry to form a surface or architectural structure, a 2.5D texture that is categorized under 3D textures, will be applied. A 2.5D texture forms a textured surface when it is placed on a surface. Applying this theory on a given point cloud will give a similar result.

3.4 Storage of Textures

Mesotructures can be stored in various forms, three of which are; texture maps, procedural textures or geometry itself [46]. These three techniques are explained in the following section with given implementation details.

3.4.1 Texture Maps

Most common form of storage of textures is through texture maps. They are usually generalized to either discrete or vector valued functions [46]. Texture maps can be categorized into three main categories; dense, hierarchical and high dimension textures:

Dense Textures

Dense textures are formed from multiple maps which can include colour maps, bumps maps, shadow maps, normal maps, relief maps, displacement maps etc. The layers are compressed. An example of a dense texture is presented by Policarpo.et.al [69] who extended 2D relief maps. The relief mapping method is generalized and can accumulate multiple layers. As in a normal relief map, both normal and depth maps are applied, in this extension however multiple layers of both these map types are used. For example with a four layer relief map, three textures need to be applied, firstly an RGBA texture that stores a depth layer in each channel, red, blue, green and alpha channels. Secoundly, for the surface normals, two RGBA textures [69].

Hierarchical Textures

A hierarchical data structure is a data structure with many levels that are arranged as a treelike structure. One type of hierarchical data structure is an octree. An octree can store texture data efficiently. It is formed up with nodes, where the initial node of the tree is a cube. The octree is partitioned recursively by dividing each node into 8 octants, see Figure 3.4 . Octrees were initially applied to texture data by Neyret [65], where every texel (volumetric texture) was stored as a 3D dataset, as a reference volume. Storing the texel in this manner formed compression within the texture data, allowing for lower computation time and inheriting a high level of detail (LOD). Octrees are applied widely in computer graphics with the recent advent for volumetric data rendering [70].

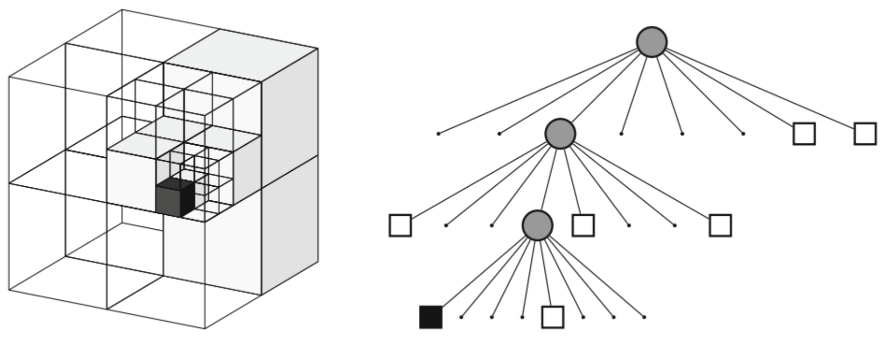


Figure 3.4 Example of an octree (Left) Volumetric model with (right) corresponding tree representation [71].

High Dimension Compressed textures

Volumetric materials can be represented by defining their appearance, or structure, as a function of many variables creating high dimensional compressed textures. Variables such as texture coordinates and viewing directions are compressed and stored within a function. One of the first types of high dimension textures is the Bidirectional Texture Function (BTF), which was introduced by Dana et.al [72]. BTF is a 6D function that models the structure and appearance of a material by its position (u, v), illumination (ϑ_i, φ_i) and viewing angles (ϑ_j, φ_j),

$$BTF\ (u, v, \vartheta_i, \varphi_i, \vartheta_j, \varphi_j). \tag{3.5}$$

A variety of lighting and shading effects like shadows, translucency and texture inter-reflections can be calculated with BTF. The main application of the BTF is the appearance of the mesostructure hence small patches of the content are optimal in this case. Deforming or modifying the surface with the texture map can create artefacts which can change the local curvature or scale of the surface.

Based on these problems, another technique that applies higher dimension textures is the ‘Generalized Displacement Mapping’ (GDM) which improves on the BTF. GDM is a 5D texture map that defines the distance of every viewing direction nearest to the surface of the mesostructure. For every point p (x, y, z), a ray is cast in the direction it is viewed in V (θ, φ) forming $dGDM(x, y, z, \theta, \varphi)$.

3.4.2 Procedural

Creating and storing complex textures is a manually tedious task. This is where the advantage of procedural textures takes place. A procedural texture is formed of algorithms, introduced by Perlin [73], who formed procedural noise functions to create stochastic effects. It is a widely applied noise function, as noise functions create randomness and fractals to form a more realistic textured image. Perlin and Hoffert [74] applied the Perlin noise function to create ‘hypertexture’, a volumetric mesostruture. The hypertexture was formed by applying Density Modulation Function (DMF), a function which modulates the density of a given point in $\mathbb{R}^3$, to the object’s Object Density Function (ODF), which is the function representing the density of a 3D shape in $\mathbb{R}^3$. These functions are applied through given primitive functions such as gain, bias, noise, turbulence with the addition of arithmetic functions. Although procedural textures can be difficult to build and the outcome is revealed after the texture is produced it has the advantage of covering an arbitrary large area, as they are not limited in size and the LOD is high as the resolution is not fixed.

3.4.3 Geometry

The final and basic storage category is actual geometry. Textures can be stored as mesostructures of geometry in texture space that can be mapped onto the macrostructure. The geometry is mapped through a shell that contains the geometry. It can be attached onto the surface of the macrostructure in three different ways; point attachment, area attachment and curve attachment (Figure 3.5). For a point attachment, the geometry is directly positioned onto the given point on the surface with the aligned orientation according to the axis of the surface by the transformation computed. Texture coordinates of the area are determined for the area attachment by projecting the geometric detail on the surface. The curved area is followed by the geometry defining the distance of the geometry by the y-axis. The geometry is attached on a curve by defining the texture coordinates of the user-defined curve function on the surface. From the source points of the geometry, one point is taken as a parameter for the given curve with the remaining curve points being determined by the final position of the surface that is orthogonal to the curve.

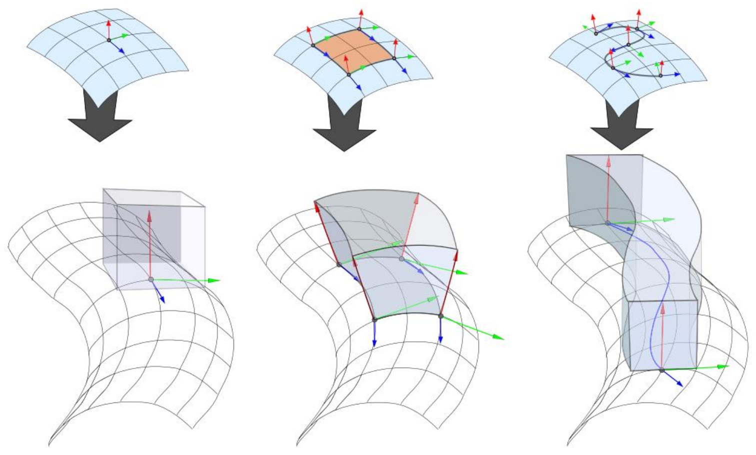


Figure 3.5 Geometry attached through a (a) Point (b) area (c) curve [75].

Storage wise, the method of shape grammar that will be used to construct 3D IGM, is a procedural technique as the shape rules act as algorithms to form a pattern. However at the stage of manipulating and mapping the 3D IGM, the 2.5D textures will be in the form of actual geometry. The reason being is that forming the 3D IGM is a separate step from mapping it. Hence the next section will provide a mapping method that implements actual geometry.

3.5 Volumetric Mapping Methods

Many of the volumetric methods for applying mesostructure on surfaces is through texture and shell space. Shell space is defined as a layer that covers the base of a surface and is calculated as,

$$\vec{G}(u,v,w) = \vec{S}(u,v) + wH(u,v)\hat{d}(u,v), \tag{3.6}$$

where $\vec{S}(u,v)$ is a 2D parameterized surface, $H(u,v)$, defines the scalar thickness per surface point and $\hat{d}(u,v)$ represents the surface normal. Shell space was introduced by Kajiya and Kay [64], as they created texels, 3D volumes that contain parameters in a 3D array approximating to the visual properties of micro-surfaces, which were mapped onto bilinear patches. The next section introduces the reader to shell maps, a technique that integrates texture space and shell maps.

3.5.1 Shell Maps

A shell map is a bijective mapping between two spaces, shell space and texture space, defined by Porumbescu et.al. [66]. It is a mapping technique that applies volumetric detail onto a given surface, employing both texture and geometric methods. The mapping is initiated with the formation of the shell space. With a given triangulated base surface, S (a textural surface) and S_0, (an offset surface), are created; the shell space then forms between these two surfaces.

There are properties that are conditional in forming an offset surface are:

- Both surfaces, S and S_0, must acquire identical mesh topology, that is the same number of triangles, connectivity and assigned texture co-ordinates.
- Every triangle (T), $T \epsilon S$ corresponds with a $T_0 \epsilon S_0$.
- Every vertex (V), $V \epsilon S$ corresponds with a $V_0 \epsilon S_0$.
- Offset surface should not intersect or have any self-intersections with the base surface.

The shell within the shell space is created when vertices of the associated triangles between the two surfaces are connected. The addition of three non-planar quadrilaterals alongside the two corresponding triangles, T and T_0, forms a prism, P. Like the triangles and vertices, every P corresponds with a texture space prism, P_t.

In shell space,

$$P = T + T_0, \tag{3.7}$$

where T is formed of vertices $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, and T_0 is formed of $\mathbf{v}_{01}$, $\mathbf{v}_{02}$, $\mathbf{v}_{03}$, .

In texture space,

$$P_t = T_t + T_{0t} \ , \tag{3.8}$$

where the vertices, $(u_i, v_i, 0)$ and (u_i, v_i, k) $(i = 1,2,3)$ forms T_t and T_{0t}, respectively. The height, k, is calculated by averaging the triangle edge lengths in S in both spaces, and multiplying it by the maximum offset height, h,

$$k = \frac{a_t}{a} * h. \tag{3.9}$$

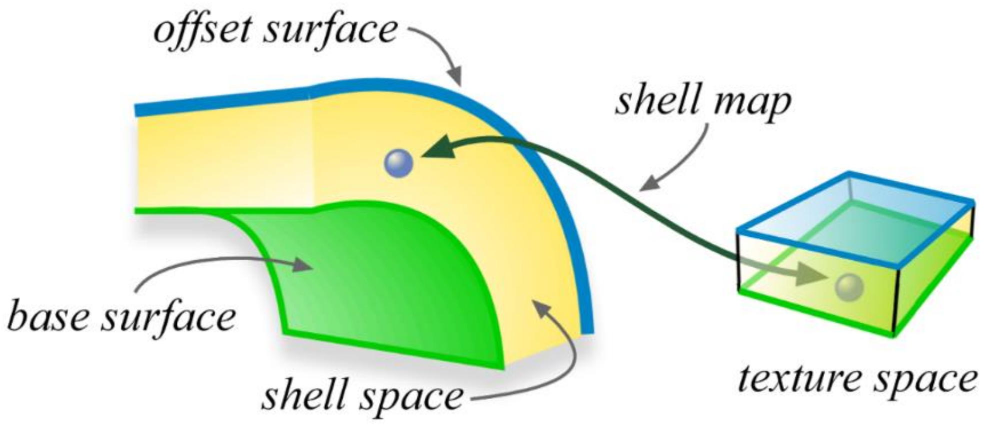


Figure 3.6 Shell space region and texture space in shell mapping [66].

For consistency and forming a robust mapping, the two corresponding prisms in the two spaces are each split into three tetrahedrons by triangulating the quadrilateral planes connecting the two triangles. This is done through the rippling algorithm that was formed by Erleben and Dolhmann [76]. Each quadrilateral edge is labelled either rising (R) or falling (F), where each configuration has two Fs and one R, or vice versa, but none of the configurations are either (FFF) or (RRR) as they are not validated tetrahedrons. Once again, each tetrahedron in both spaces are associated with each other to map the volumetric texture correctly.

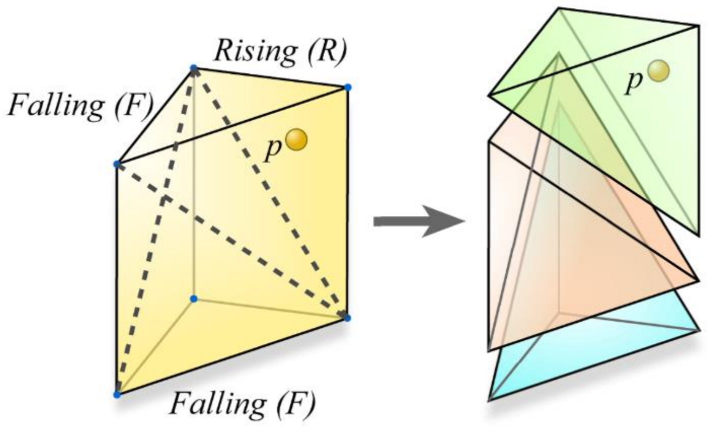


Figure 3.7 Prisms split into tetrahedrons through rippling algorithm [66].

With the shell space formed, the points from the texture space are mapped to the shell space by the combination of the point location algorithm [77] and Barycentric coordinates.

The shell mapping function is,

$$\boldsymbol{p}_t = \varphi\,(T_t, B(T, \boldsymbol{p})),$$
$$\boldsymbol{p} = \varphi\,(T, B(T, \boldsymbol{p}_t)), \tag{3.10}$$

where $\boldsymbol{p}$ and $\boldsymbol{p}_t$ are points in tetrahedrons in shell and texture spaces, respectively. B defines the Barycentric coordinates of the points and φ defines a point with Barycentric coordinates.

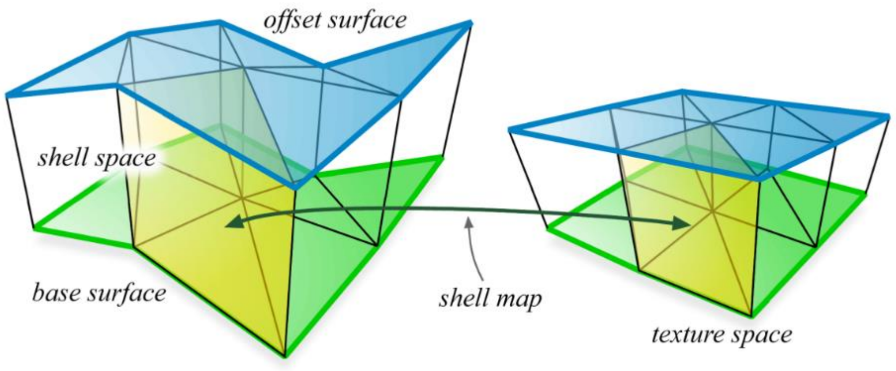


Figure 3.8 Correpondence of prisms in shell and texture spaces [66].

The texture space within shell maps allows various content storage including geometry and procedural volumetric textures, creating flexibility. The formed shell maps on the surface also coincide with existing rendering techniques, which enables one to render a shell mapped surface effectively. As shell maps allows the addition of volumetric detail within the shell map region, this does not distort the original surface like displacement mapping. For attaching overhanging volumetric geometry onto a surface, shell maps are the ideal choice.

3.5.2 Shell Mapping Implementation

Although shell mapping was defined by Porumbescu, the formation and mapping in shell space was applied earlier than its stated development. Texels that were formed by Kajiya and Kay [64] were applied and stored in shell space by extruding a bilinear patch towards the surface normal. To avoid self-intersections of the surface extruded, Peng. et.al [78] applied the use of multiple layers within each shell space volume. The layers were either interior, exterior or envelope, layers that are above or below the surface. The height of the extrusion was however followed by vectors known as line directors.

The use of applying a shell map on a triangulated surface was introduced by Wang.et.al. [79]. The prisms were however triangulated by their fins (outer shell of prism), into two triangles per fin. The splitting of the prism into three tetrahedrons was proposed and applied by Hirche.et al [80], which applied the Barycentric mapping.

A similar technique to Porumbescu [66], was applied by Dufort et.al [51] who used the shell mapping technique to render semi-transparent volumetric detail. The shell space was formed from an extruded triangular mesh that directed the offset surface to the base surface normal. The prisms were also spit into tetrahedrons. Applying volumetric data into prisms that are later split results in a piecewise problem, where artifacts can be formed. To solve this problem, Jeschke et.al [81] used a triangular mesh which was extruded, but instead of splitting the prism, a coons patch was added at the intersection of w –coordinate in the (u,v,w) texture space, which formed the corresponding world space triangle. There are a number of other implementations of shell mapping by [82], [83].

The initial step in every implemented technique described above was forming the shell space. However, there is one particular method that follows the technique of shell mapping without a shell map. Brodersen et.al [84] used a particle tracker known as Lagrangian tracker particles. The particle tracker distributes particles evenly at texture coordinates over the given surface patch. The particles create 2D grids to form a 3D lattice.

3.6 Summary

The main aim of this research is to manipulate the 3D IGM to form architectural structures as a mapping technique. With the various methods and techniques described in this chapter it is clear that the scale of the geometry should be of mesostructure size, as this forms fine detail. The procedural technique of shape grammar generates the 3D IGM but the 2.5D texture, the generated 3D motif, is stored as actual geometry.

As shell maps are an efficient method for mapping geometry onto surfaces, shell mapping is the optimal method that will be applied in this research. However the method is formed of different techniques which include; geometrical techniques of forming shell and texture space, triangulation of prisms; the use of a triangulated mesh and the Barycentric mapping. With all these techniques, the mapping is very costly, storage wise. Eliminating the fact that the surface (formed) is not triangulated, can ideally map any shaped motif. Also triangulating the shell and texture space is an additional process. Hence in this research the process of triangulation and triangulated surfaces will not be applied.

4. 3D Islamic Geometric Motifs

"To understand is to perceive patterns."[4]

4.1 Introduction

A number of techniques have been applied in tackling the problem of pattern generation resulting in various computer graphic toolkits for diverse pattern formations.

IGP are very similar to fractals [85] as both are characterised with self-similarity, and therefore, methods like the Lindenmayer System (L-system) and Iterated Functions could be applied. However, literature on both traditional and computerized construction techniques of IGP, showed the relevance of shape grammar, as the method only applied the use of a single shape throughout the entire pattern formation.

Although shape grammar is a method that was previously utilized to form IGP [4] it did not create complete forms, or even a numerous amount of IGP [43]. None of the techniques described in chapter 2, which applied shape grammar as a pattern formation method, were able to create a 3D IGP with generalized shape

[4] *Isaiah Berlin.*

grammar rules. In this research the shape grammar is devised to create complete forms of *n*-fold IGM in 3D. Additionally, novel forms are generated with these rules, by the use of parameterization, formally by a parametric shape grammar.

4.1.1 Parametric Shape Grammar

Parametric shape grammar is an extension to shape grammar [39]. It is more flexible and the rules in the grammar create a diverse group within itself. The shape rule,

$$A \rightarrow B, \tag{4.1}$$

becomes,

$$g(A) \rightarrow g(B), \tag{4.2}$$

where g is the parameterized function. To illustrate the use of parameterization, an example is given below.

A translational shape rule of +1cm in the standard shape grammar, which takes a shape from its original position and translates it in the x-axis by 1 unit up, would simply translate it. But if this rule was parameterized, one could translate the shape x amount in the x-axis in either the positive or negative x-axis with parameters of 0.1, 02, 0.3…etc., see Figure 4.1.

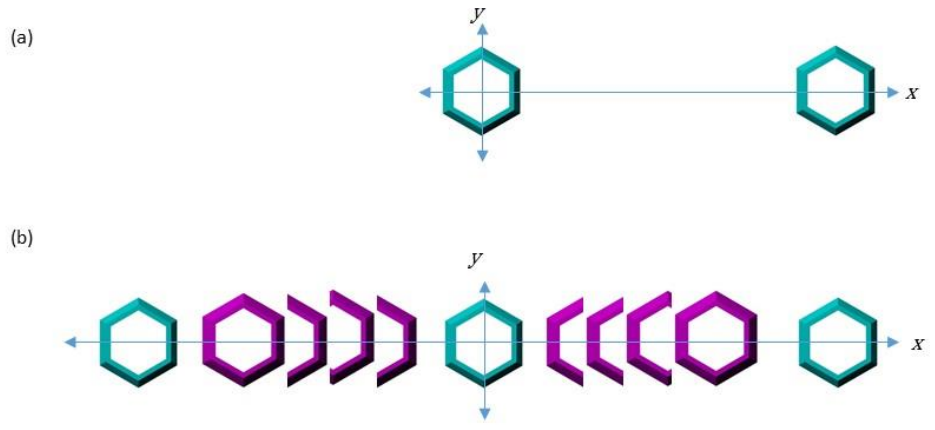


Figure 4.1(a) Shape grammar rule. (b) Parameterized shape grammar rule.

4.1.2 Algebra of Shapes

In shape grammar, the shape is the important feature. The spatial dimension is the relationship between the grammar and the shape [86]. To define a shape, one can deconstruct it to its basic elements that form the algebra of a shape, which are the points, the lines, the planes and the solids. To illustrate this, a 2x2x2 cube is formed of 8 solids, 36 planes, 54 lines and 27 points (Figure 4.2).

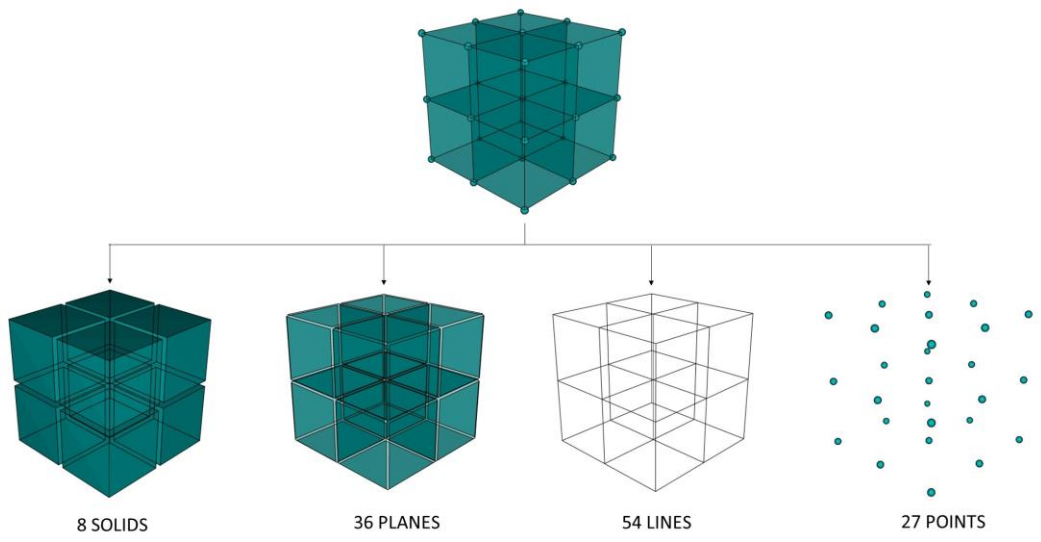


Figure 4.2 Deconstruction of a cube into its' basic elements.

These elements are converted from shape algebras (SA) definitions to computer graphic (CG) definitions as:

$$\begin{tikzcd}
\text{SA} & \text{POINTS} \arrow[d] & \text{LINES} \arrow[d] & \text{PLANES} \arrow[d] & \text{SOLIDS} \arrow[d] \\
\text{CG} & \text{VERTICIES} & \text{EDGES} & \text{FACES} & \text{OBJECTS}
\end{tikzcd}$$

The algebraic theory of shape grammar is defined by 5 important concepts [86].

Elements. The basic elements, points, lines, planes and solids.

Properties**.** The properties of the elements are; dimension, boundary, content and 3D Medium.

Spatial dimension**.** Each element has a corresponding dimension, point being zero dimension (0D) and a solid, 3D.

Shape boundaries**.** Boundaries create the shape, for example a line is bounded by two points and a solid is bounded by a minimum of 4 planes.

Relationship**.** How the element relates to the properties.

Table 4.1 summarises the properties of the basic elements. In regards to a point, it is the smallest element with zero dimension and no boundary or content, hence it is indivisible. The line (1D), the plane (2D) and the solid (3D) are all divisible elements; the line into points, plane into lines and the solid into planes.

A non-zero dimension is essential for shape formation, from 1D that represents a line to 3D that forms a solid. The algebras of shape are broken down into 3 categories:

1. The shape itself
2. Part relationships
3. Euclidean Transformations

Table 4.1 Properties of Basic Elements (Reproduced from [86]).

Basic Element	Dimension	Boundary	Content	3D Medium
Point	0	None	None	Vertex
Line	1	2 Points	Length	Edge
Plane	2	3 Lines (Minimum)	Area	Face
Solid	3	4 Planes (Minimum)	Volume	Object

The shape is essential as it contains all the basic elementary properties. Part relationships are formed when shapes interact with each other during the process of the shape grammar, this includes the Boolean operations, and the Euclidean transformations that transform the shape.

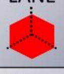
The U_{ij} is the algebraic numerical form of a shape [86]. The index i defines the dimension of the basic element and the index j is the resulting transformed shape dimension. From Table 4.2 it can be seen that U_{0j} forms point-based shapes, U_{1j}

are line-based shapes, U_{2j} are plane-based shapes and U_{3j} are solid-based shapes. When two shapes are formed the indices, i and j should follow the criterion,

$$i \geq 0,\ j \geq i. \tag{4.3}$$

This is because when manipulating a shape within the grammar, the dimension of the resulting shape should be greater than or equal to the basic element that initiates the shape formation process. The algebras is this research are enumerated up to 3 dimensions as the objective is to form, 3D IGM, however the enumeration can increase with added dimensions [86]. In Table 4.2 the enumeration of the algebra of shapes is formalised with given illustrated examples below them.

Table 4.2 Properties of Shapes (Reproduced from [86]).

Algebra U_{ij}	U_{i0}	U_{i1}	U_{i2}	U_{i3}
U_{0j}	U_{00} POINT ●	U_{01} POINT ● ●	U_{02} POINT ● ● ●	U_{03} POINT ● ● ●
U_{1j}		U_{11} LINE	U_{12} LINE	U_{13} LINE
U_{2j}			U_{22} PLANE	U_{23} PLANE 
U_{3j}				U_{33} SOLID

In this chapter, the methodology undertaken in this research to create 3D IGM, using the parametric shape grammar approach, is presented. The proposed framework, the IGMBS framework, is discussed providing detailed descriptions on the use and change of the initial shape whilst the importance of parameterizing the grammar rules are also reasoned. Lastly, the methodology to create 3D IGM is provided, producing good aesthetic results in the implementation section. The results are then analyzed, concluding the chapter with next steps to further the research.

4.2 Proposed Framework – IGMBS

The IGMBS framework is based on creating both the 3D motifs and motif-based structures. However, in this chapter the initiation and generation of the 3D motifs is the main discussion. The IGMBS framework is made up of three important steps which are:

- Select IS
- Apply PSG rules to generate motif
- Apply APSG rules to generate motifs

4.2.1 Initial Shape

The IS is the starting point for constructing 3D IGM; it is the shape that forms the motif. For example, to construct a 6-pointed star the first step would be to start off with a regular polygon, *n*-gon, e.g. a hexagon. In the shape grammar methods described in chapter 2, only one initial shape was adopted throughout the pattern generation, allowing the grammar to produce patterns with a single shape. In this grammar, the initial shape can be changed. The changeability will create flexibility in the shape grammar to produce various categories of IGM. For example, with a hexagon as an IS, one is limited to generate only 6- or 12-fold motifs whereas changing the IS from a hexagon to an octagon, the grammar can create 4-, 6-, 8- and 12-fold motifs. The initial shape is however constrained to a regular polygon of *n*-sides, as IGP are created with regular polygons [4]. The circle, although it is not a polygon, is included in the set of initial shapes as every polygonal shape initiates its form from it. Figure 4.3 shows a few examples of IGP with proposed initial shapes indicated in red.

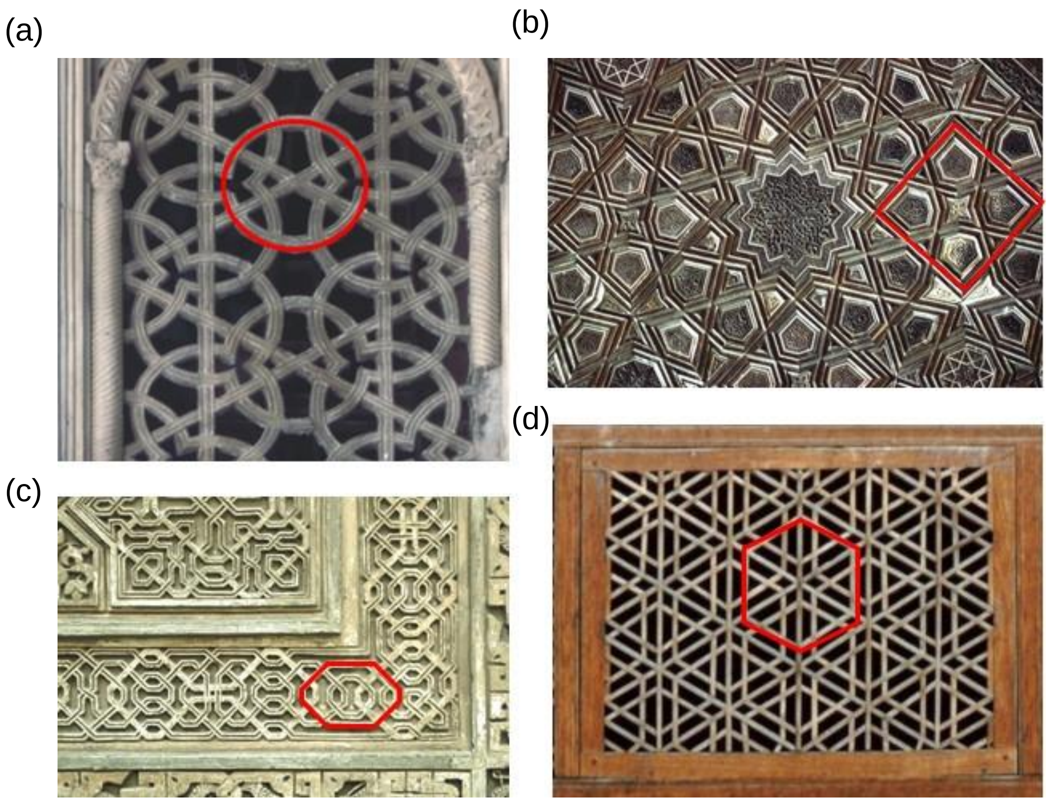


Figure 4.3 Examples of proposed initial shapes in various IGP [25].

Figure 4.3 (a) is a lattice screen from the Great Mosque in Damascus, Syria, which was built in 715 AD during the Ummayad dynasty. A part of the minbar (pulpit) decoration is shown in (b) from Al-Salih Tala'I Mosque in Cairo, Egypt. This mosque was built in 1160 AD during the Fatimid dynasty. The decorative 4-fold patterned panel (c) is situated in Alhambra Palace in Granada, Spain; a well-known historical monument built by the Nasrid dynasty between 13th- 15th AD. Finally, another lattice screen was found in the Tilya-Kan Madrassa, which was built in 1660 AD by the Shaybanid dynasty in Samarkand, Transoxiana.

4.2.2 Parameterized Shape Grammar (PSG)

Following the example of the 6-pointed star, the shape requires rules to transform it into a derived design or pattern. To form the 6-pointed star, rules of translation and rotation need to be applied. These rules are in fact parameterized in this shape grammar to form the parametric shape grammar rules. The parameterization gives the advantage of generating numerous patterns with the single initial shape. For example, this can be achieved by varying its size (using the scale rule), or changing the parameter of the translation rule.

4.3 Methodology: Generating the Motif

In accordance to the shape grammar definition, I is a regular polygon, e.g. triangle, quadrilateral, pentagon, hexagon, heptagon, octagon etc., with the exception of the circle, in this grammar. V_T is the terminal shape (IS without the marker), V_M is the spatial marker which positions the successive assemblage of the shape by its given shape rule (example shown in Figure 4.4) and lastly, the shape rules, *R*, are the rules generated to construct *n*-fold geometry.

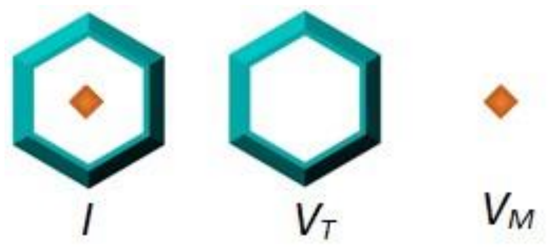


Figure 4.4 Shape grammar notation.

4.3.1 Analyzing IGP

The circle is a symbolic shape in Islamic art. It symbolizes one God and also Mecca as it is the central place for Muslims as they face towards it during prayer. Hence it is the shape that forms the initiation of all the IGP, the central point. Three fundamental shapes are derived from the circle; the triangle, the square and the hexagon. These three shapes symbolize [87]:

- Harmony (triangle)
- Materiality (square)
- Heaven (hexagon)

It is these shapes that derive the complexity and represent symbolism within IGP. With this in mind, analyzing the IGP was initiated with these three fundamental shapes.

As the IS is changeable in this grammar, the category of IGP will be of *n*-fold geometry. From [88] many 4-, 5-, 6- and 7-fold patterns of varying complexity, containing both stars and polygons, were analyzed to derive the grammar rules. The central motif of every pattern was extracted and deconstructed, to visualize how it could be constructed with a single regular polygon; the extraction process was done visually. As many of the patterns are formed by repetition of regions within the pattern, it was a clear observation that the central motif would be suitable to analyze and create. Also, as it was mentioned previously, the same motif is applied in different patterns that are found in different regions.

Figure 4.5 shows two examples, one that is bounded by a hexagon and the other by a decagonal motif.

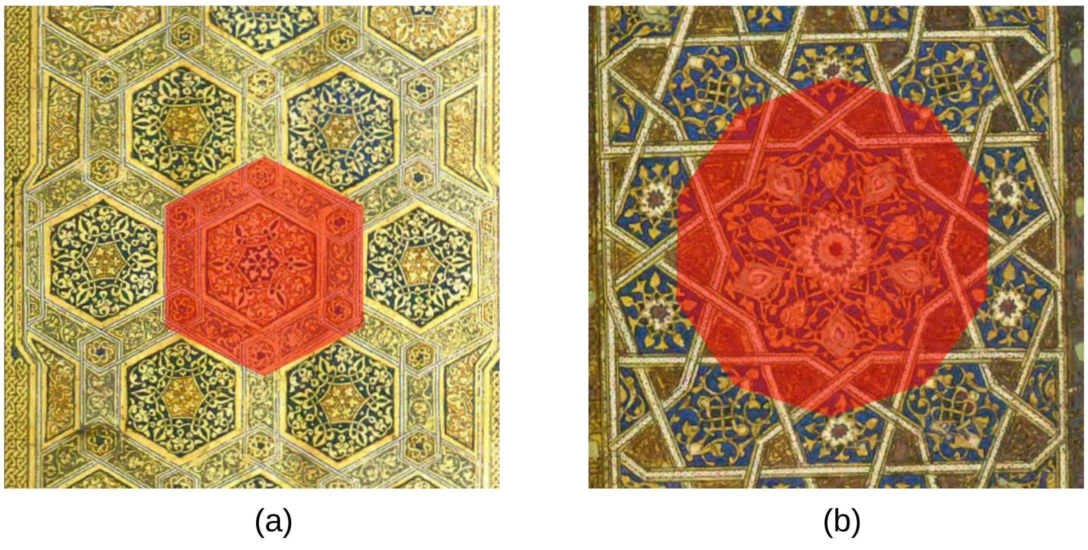

(a) (b)

Figure 4.5 Examples of motifs bounded by a regular (a) hexagon (b) decagon [8].

Both examples in Figure 4.5 are from Qur'anic manuscripts. Figure 4.5(a) is a carpet page taken from the 25th volume of the Uljatu Qur'an from Iraq, produced in the 13th century. The carpet page with the decagonal motif (b) is from the 8th volume of the Sultan Faraj ibn Barquq's 30 volume Qur'an. This Qur'anic manuscript was produced in 14th century, in Egypt.

There are however some patterns that contain more than one motif. The pattern in Figure 4.6 is one case. Visualizing how many motifs and the type of motif (*n*-fold) the pattern contains can be done by the central polygon or star polygon. In this pattern, one can see both a polygon, which is a regular octagon that is rotated at an angle, and a star polygon. To create the motif containing the octagon at its centre, the motif initiates with the square, which is rotated approximately at a given angle. It is then reflected or rotated at its centre of origin by 180º. Next, it is rotated at the centre origin four times to complete the motif. When the motif is tessellated to form the pattern, the motif is overlapped.

For the star polygon, again the same initial shape, the square, is applied by translating and overlapping when rotated (blue parts in motif 2) to form the motif. The motif is then tessellated to create the 2D IGP. The motif is tessellated by overlapping a part of the motif onto the next consecutive motif. The overlapping creates a Boolean operation. Both motifs tessellate in the Cartesian grid, creating a periodic pattern.

From analyzing the patterns, it was seen that geometric transformations, translation, rotation and scale were the most common transformations to create the motif. Constructive Solid Geometry (CSG) was also in play in the generation of the motifs.

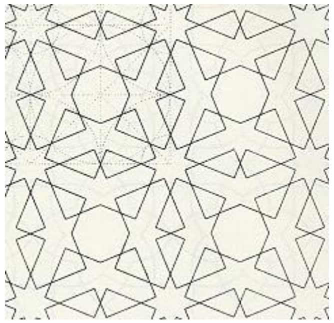


Plate 44 [88]

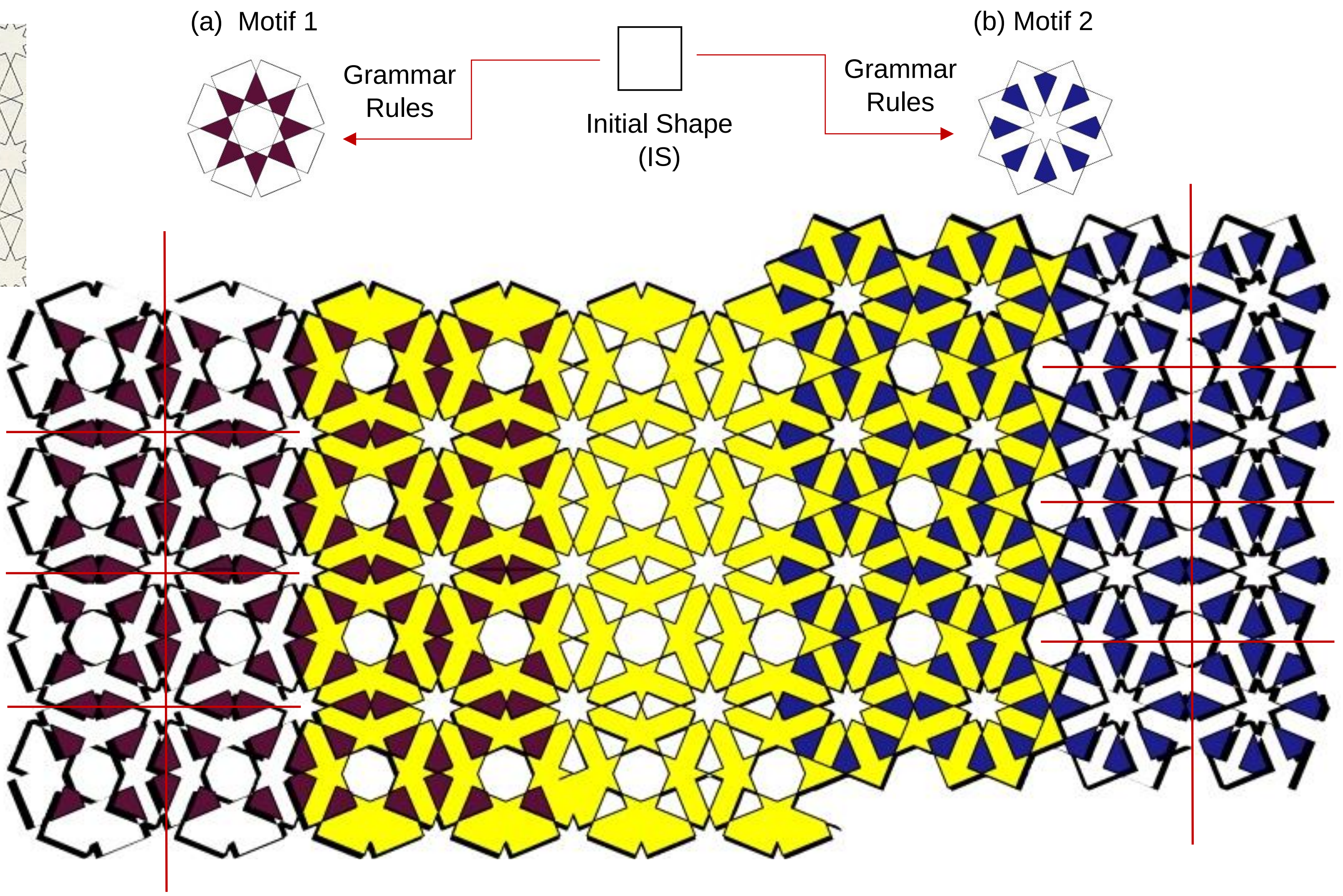


Figure 4.6 Analyzing a pattern created from two different motifs both generated from a square.

CSG is a method for creating solid objects from other solids. It has 3 operators; union, intersection and difference. These operators are known as Boolean operations. The union unites the two solid objects at the given position, the intersection creates the part that is included in both the objects and the difference subtracts one object from another. Hence, when the IS or the motif is overlapped in any kind of manner, the Boolean operations occur. The grammar rules are discussed in detail in the next section.

4.3.2 Parameterized Shape Grammar (PSG) Rules

The PSG rules were derived from analyzing patterns from [88] with reference to the three distinct features of IGP; symmetry, emergence and self-similarity. The symmetry and emergence features both generate the rules of translation and rotation. A star emerges by translating and rotating (from the centre) a single hexagon generating a symmetrical 6-pointed star with an order of symmetry 6 (Figure 4.7). Scaling the hexagon proportionally retains the original state of the shape and is a cause for generating self-similarity within the derived pattern. Furthermore, Boolean operations are also utilized to generate emergence as it produces new emergent designs.

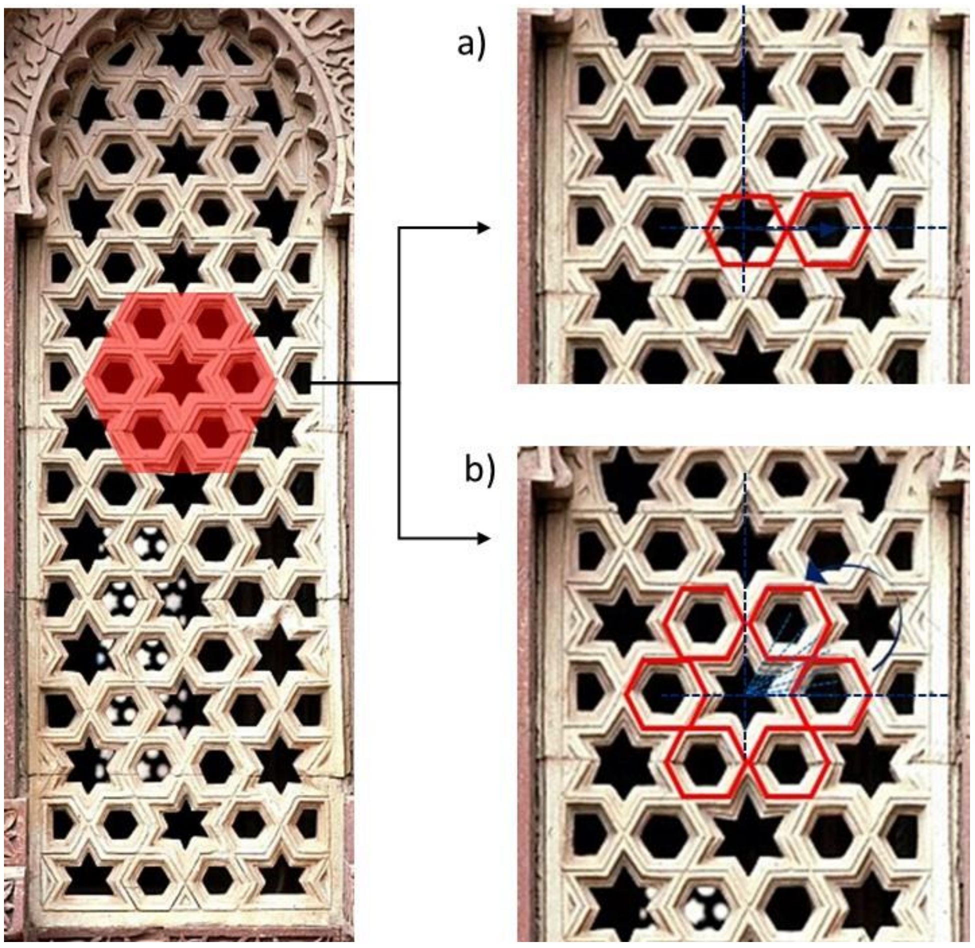


Figure 4.7 6-Pointed Star formed from a) translating and b) rotating the hexagon.

Figure 4.8 shows the grammar rules generated from numerous patterns. Rule 1 (R1) translates in either x- or y- axis. Rule 2 (R2) rotates the shape with parameter r, where r is calculated as,

$$r = 360/n. \tag{4.4}$$

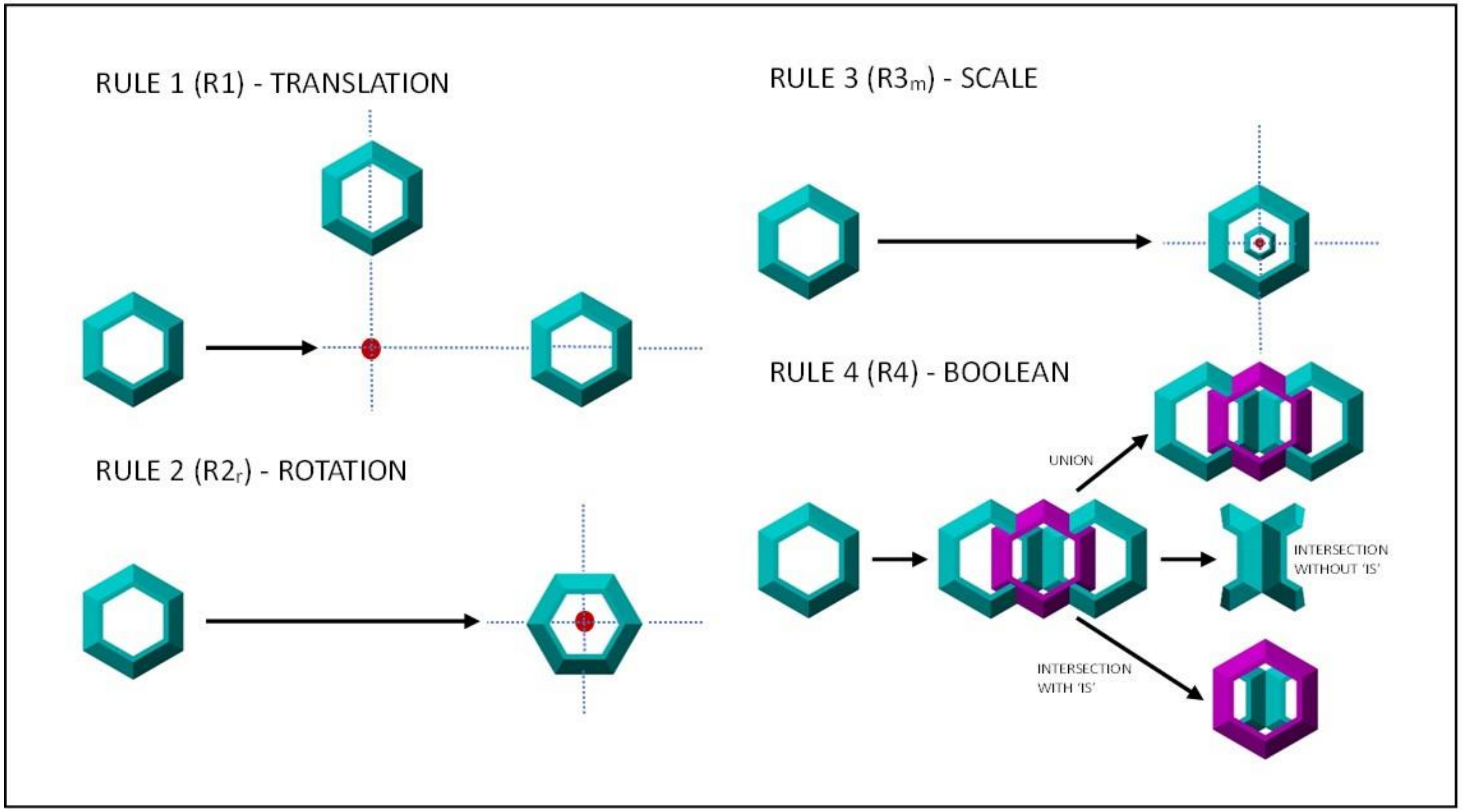

RULE 1 (R1) - TRANSLATION
RULE 2 ($R2_r$) - ROTATION
RULE 3 ($R3_m$) - SCALE
RULE 4 (R4) - BOOLEAN
UNION
INTERSECTION WITHOUT 'IS'
INTERSECTION WITH 'IS'

Figure 4.8 Parameterized Shape Grammar (PSG) rules.

As symmetry is a very important feature, to distinguish the *n*-fold in a motif, the user is restricted to the parameterization of this rule, which can only be computed by equation (4.4). For example, to create a 4-fold motif, the angle of each fold has to be a quarter of 2π, to create a regular 4-fold motif. Hence, the user would simply specify the n-fold. The scale transformation is rule 3, (R3), with parameter, m, and lastly the Boolean operation of uniting and intersecting with or without the IS, is rule 4 (R4). The grammar also includes an Add and Duplicate rule. By definition this is now a Parameterized Shape Grammar (PSG).

4.4 Implementation

The implementation of the PSG was processed within Autodesk Maya, a 3D software package, as it visualized the IGM from the written code. The three dimensionality of the motif was initiated from the IS, the IS being 3D. This was created through a piece of code which was written in Python, an integrated programming language within Maya. The algorithm to create the 3D IS is :

1. *Create the shape with depth (x,y,z).*
2. *Duplicate, scale down and Boolean operator (Difference) to create the shape outline.*
3. *Extrude and scale the front face of the shape to create the third dimension.*

The Python code is:

```
n = 6                                                          Number of sides of the initial shape

cmds.polyPrism(n = 'shape', l=0.5, w=1, ns=n)                  Create the shape with depth (x,y,z)
cmds.rotate (90,0,30)

cmds.duplicate('shape')
cmds.select('shape1')                                          Duplicate, scale down and Boolean
cmds.scale(0.9,1,0.9)                                          operator (Difference) to create the
cmds.polyBoolOp( 'shape', 'shape1', op=2, n='IS' )             shape outline.
cmds.delete(ch=True)
cmds.select('IS.f[n] ', r=True)
                                                               Extrude and scale the front face of
cmds.polyExtrudeFacet( 'IS.f[n]',kft=True, ltz=0.2, ls=(.8, .8, 0))   the shape to create the third
cmds.selectMode(object=True)                                   dimension.
```

Figure 4.9 shows the process of converting a regular polygon into a 3D polygon, creating the 3D IS to initiate the motif formation.

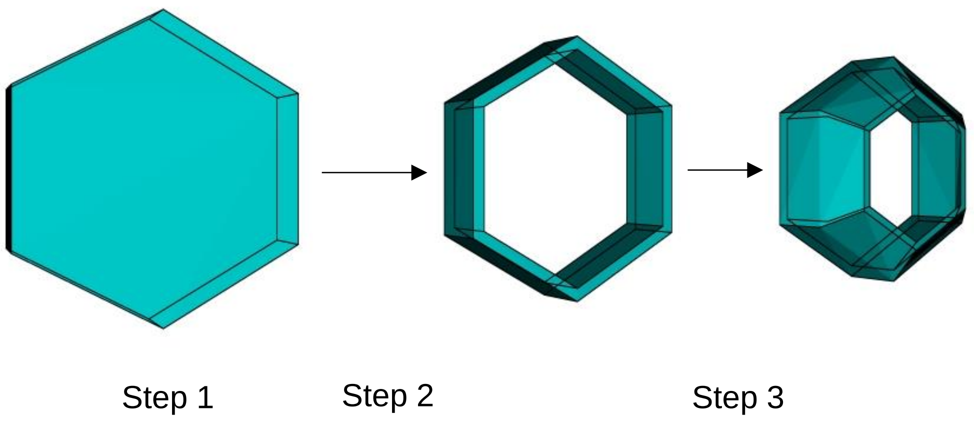


Figure 4.9 Process for creating a 3D IS.

To illustrate the methodology to generate the motifs, a few examples are shown in Figure 4.10 -Figure 4.13 with the given Python code. Each code is provided with comments written in red. Figure 4.10 is an example of how an original IGP can be generated by applying the PSG rules.

bbox = cmds.exactWorldBoundingBox('motif') *c = bbox[0]* *d = bbox[3]* *w = (d-c)*	*Calculating the size of the initial shape*
*r = (n*w)/(2*pi)*	*The radius of the Initial Shape*

for i in range(n):
    cmds.move(0,r/1.5,0)
    cmds.xform(ws=True, piv=(0,0,0))
    cmds.rotate (0,0,(i(360/n)))*
    cmds.select('motif')
    cmds.duplicate('motif')
cmds.delete()
	Application of the rules of translation, rotation and duplicate

cmds.select(all=True, r = True)
cmds.polyUnite(n='pattern')
cmds.delete(ch=True)
	Uniting of the pattern

cmds.polyPrism(n = 'shape', l=0.5, w=1, ns=n)
cmds.rotate (90,0,30)
cmds.select('shape.f[n+1]', r=True)
cmds.polyExtrudeFacet('shape.f[n+1]',kft=True, ltz=0.2, ls=(.8, .8, 0))
cmds.selectMode(object=True)
	Addition of an initial shape

cmds.polyBoolOp('shape', 'motif', op=2, n='IS')
cmds.delete(ch=True)
	Intersecting the pattern including the initial shape

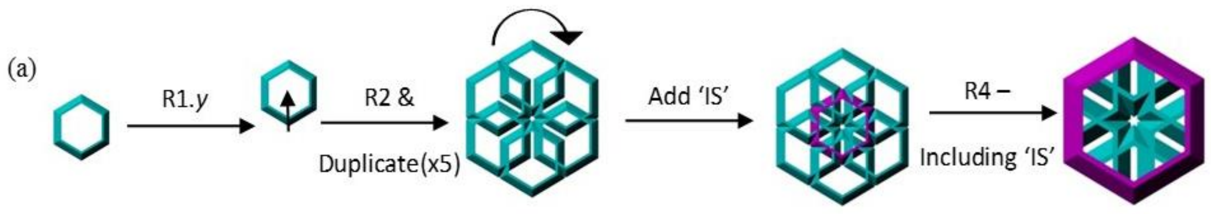


Figure 4.10 Formation of an orginal IGP with its' rules.

If the translation parameter is changed in the previous example, leaving the rest of the other rules and parameters the same, another original IGP can be derived, as shown in Figure 4.11.

```
for i in range(n):
    cmds.move(0,r/1.25,0)                                  Translation parameter changed
    cmds.xform(ws=True, piv=(0,0,0))
    cmds.rotate (0,0,(i*(360/n)))
    cmds.select('motif')
    cmds.duplicate('motif')
cmds.delete()
```

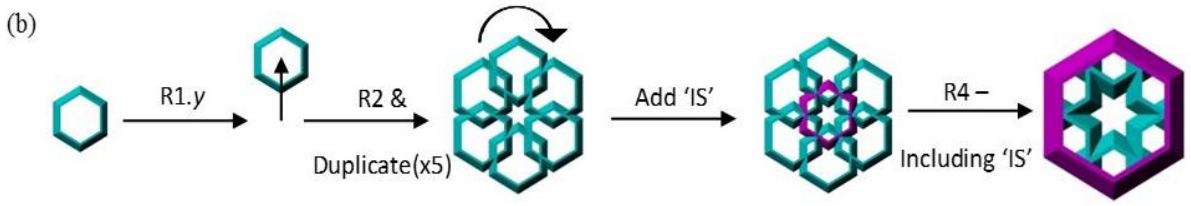


Figure 4.11 Translation parameter changed to form an original pattern.

By changing the initial shape from a hexagon to a decagon, this also generates an original motif (see Figure 4.12).

```
n = 10                                                     Changing the initial shape

cmds.polyPrism(n = 'shape', l=0.5, w=1, ns=n)
```

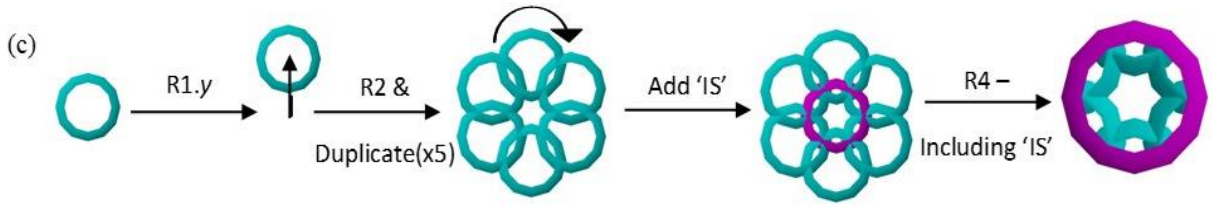


Figure 4.12 Initial shape changed to form an original pattern.

Finally, Figure 4.13 demonstrates how removing a rule and changing the initial shape again produces an original motif.

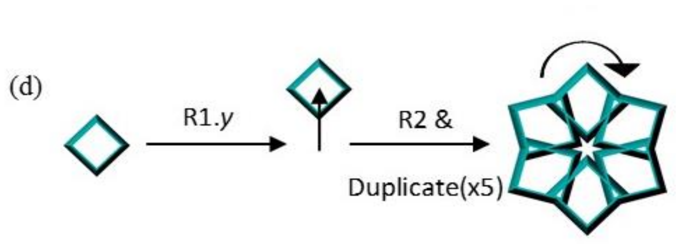


Figure 4.13 Initial shape changed and rule removed to form an original pattern.

4.4.1 3D IGM

Figure 4.14 shows the Graphical User Interface (GUI) of a set of 3D motifs implemented in Maya Python. The rendered 3D generated motifs are shown in Figure 4.15. The motifs are generated using various initial shapes from a triangle to a dodecagon. The PSG can create over 50 different 3D IGM.

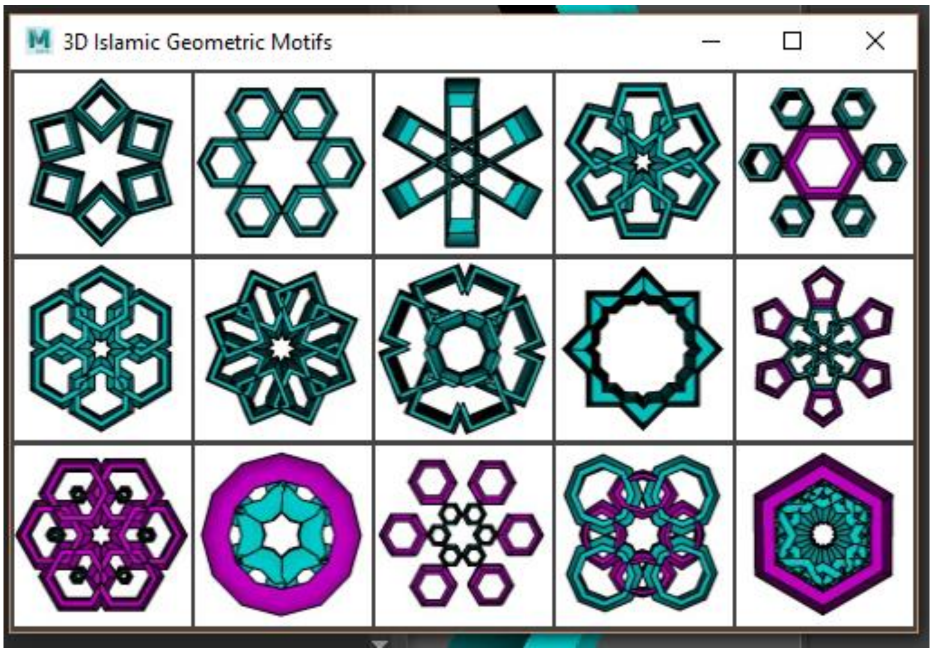


Figure 4.14 GUI of the selection of 3D IGM.

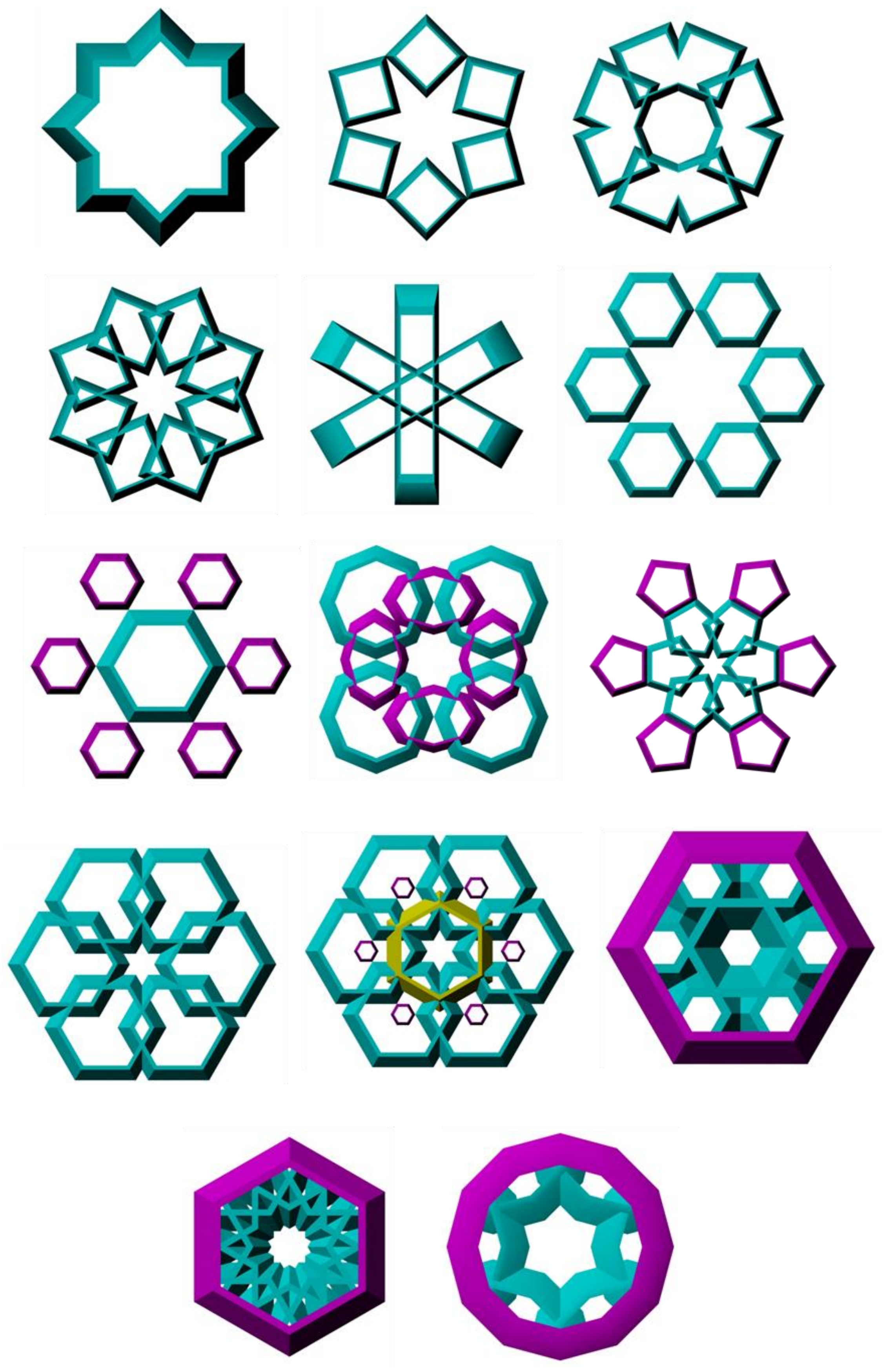


Figure 4.15 Rendered set of 3D IGM .

The PSG can create over 50 different motifs of varying categories, 4-6 fold, with the use of a single changeable initial shape. With shape grammar one can see how a pattern is formed throughout the entire generation and parameterizing the rules created flexibility in generating a variety of motifs, applying the same rules as seen in Figure 4.10Figure 4.13. By the addition of an extra dimension, the grammar is not restricted to form 2D motifs, but it can create 3D motifs, with a carved appearance.

4.5 Summary

The first objective in this research has been achieved. The PSG successfully shows how parameterization within a shape grammar, with the use of the changeable initial shape, can create numerous 3D IGM in reference to [88]. The 3D IGM are now ready to be manipulated to form architectural structures by adopting the same transformational rules from the PSG. In the next chapter, the final step in the proposed framework, IGMBS, will be discussed with detailed theory on the importance of tessellations with regularity within a pattern and the applications of shape grammar in the architectural world.

5. Architectural Structures

"Form follows function."[5]

5.1 Introduction

In the previous chapter, the generated PSG showed how a variety of 3D *n*-fold IGM can be formed without the use of a grid, with Euclidean transformations. The question now is: What are the motifs used for? A motif is a repeat unit, and when tessellated creates a pattern, like a brick, the layering of which forms a brick wall. Islamic architectural surfaces are adorned with IGP, the patterns are either engraved or carved out. The applications of shape grammar go beyond creating patterns and designs to forming architectural building plans [38], [39], [42], [89]. In this modern age, 2D plans lack visual appeal in comparison to complete 3D models. Applying the two principles, the methodology here will not only apply the 3D motif as a volumetric texture but it will use the single 3D motif to form a 3D architectural structure through given rules.

In the following chapter, the final step of the IGMBS framework is discussed in detail. This is the step that forms the architectural structures with an automated parameterized shape grammar. The chapter introduces the reader to the

[5] *Louis Sullivan.*

importance of tessellations within Islamic patterns continuing onto the main features of Islamic architecture. An accurate description of application of the APSG rules is provided with the methodology, later described in detail, providing algorithms for each architectural structure. It is then concluded by implementing the methodology and visualizing 3D IGMBS.

5.2 Tessellations within Islamic patterns

When a surface, S, is covered with tiles that form no gaps or overlaps it is known as a tiling or tessellation in mathematical terms, in 2D. Tessellations can be extended to higher dimensions that are then known as space-filling, like a honeycomb. The mathematical definition of a tessellation, $T,$ is [90],

$$T = \{T_1, T_2, \ldots..T_n\}\ , \tag{5.1}$$

where T_1, T_2,…. T_n, are tiles that form T. For the tile to tessellate, it has to be enclosed with a boundary to abide with the closed set condition. The union of all the closed sets, i.e. the tiles, forms the ‘no gaps’ where,

$$\{T_1 \cup T_2, \ldots..\cup T_n\} = S. \tag{5.2}$$

The interior of the sets are disjoint and therefore they do not form overlaps in the surface [90],

$$\{interior\ of\ T_i \cap interior\ of\ T_j\} = \varphi, \tag{5.3}$$

where i and j are the tile numbers and $i \neq j$ and φ is the disjoint set of a pair of tiles.

5.2.1 Periodicity in IGP

A variety of shapes can form a tessellation. However, to produce periodicity within a tessellation, the tiles/shapes should transform periodically restricting it to the condition of applying rigid Euclidean transformations. As the research applies the use of regular polygons, including the exception of the circle, as the initial shape that forms the 3D motif, the tessellations will be produced with this set of shapes.

Mathematical Proof of regular tessellations

The interior angle of the regular polygon $\frac{180(p-2)}{p}$, where p is the number of sides of the regular polygon should equal to a whole number that is divisible by 360˚[91]. Multiplying this by q, which is number of polygons meeting at a point thus results in 360˚,

$$\frac{180(p-2)}{p}(q) = 360\ . \tag{5.4}$$

Hence the only possible values are, {3,6},{4,4} and {6,3}, these are written in the form of Schläfli symbol $\{ p, q \}$ (see Figure 5.1).

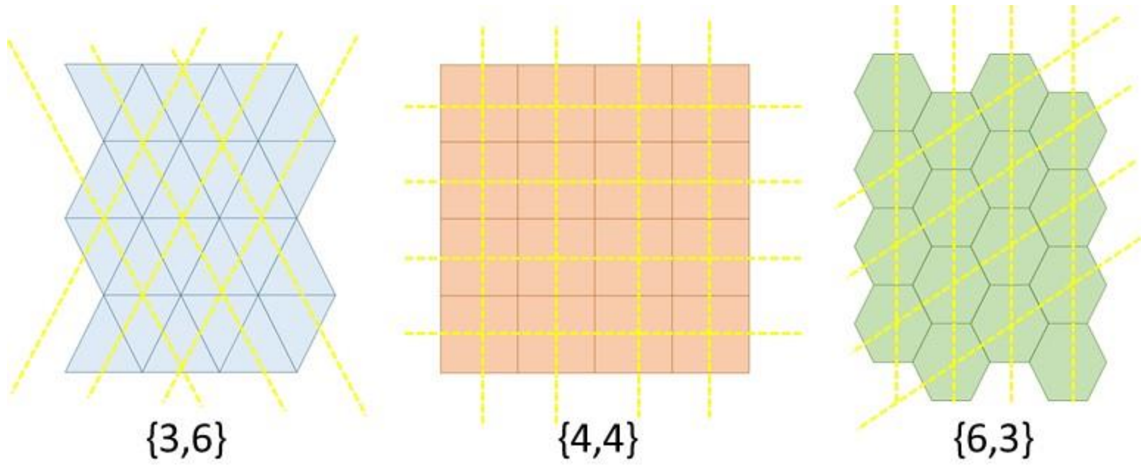


Figure 5.1 Regular tessellations of triangle, square and hexagon.

Other than the regular tessellations, the semi-regular or Archemidean tiling, tessellates a plane with 2 or more regular polygons such that the same polygons surround the same vertex, which forms a periodic tessellation. However, in this research, the aim is to apply only one motif to map and construct the architectural structures. Hence, the regular tessellations will be implemented. From analyzing the IGP, it was also seen that all the IGP that had a periodic tessellation, either tessellated in the Cartesian (square) Figure 5.2 or the hexagonal grids Figure 5.3. The reason why a triangle is not applied as a tessellation, is firstly, six regular triangles form a hexagon and second, the triangle has the smallest area compared to the square and hexagon [24]. For this reason, the Cartesian and hexagonal grid will be used as a mapping template.

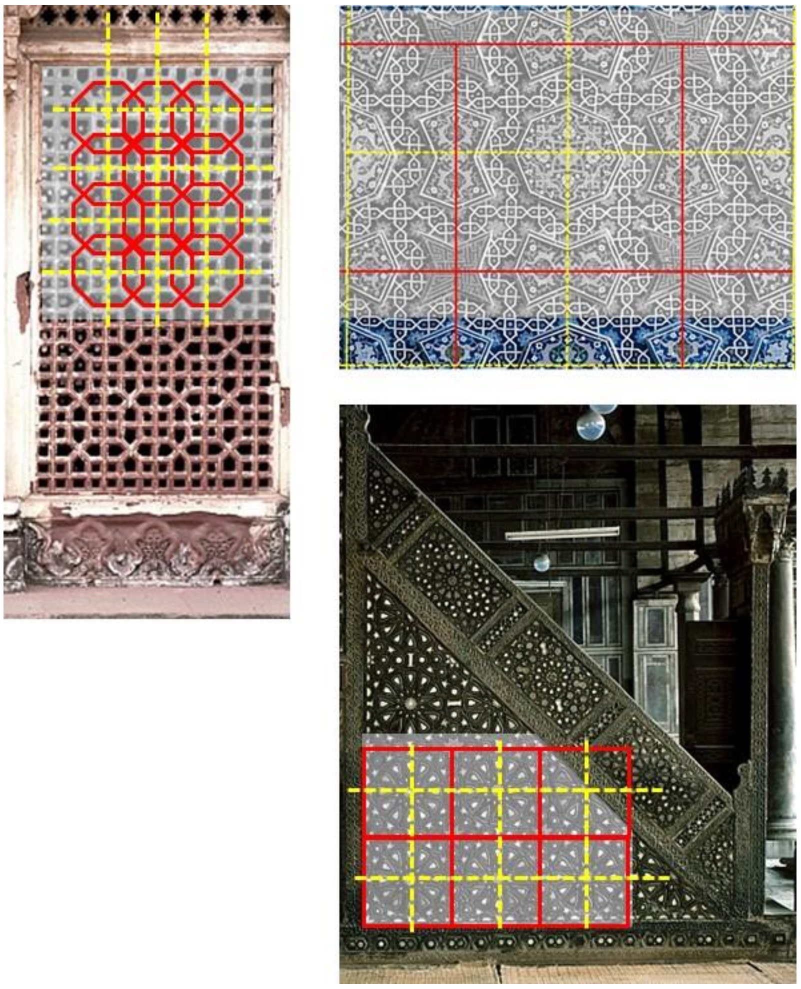


Figure 5.2 Tessellation in the Cartsian grid [25].

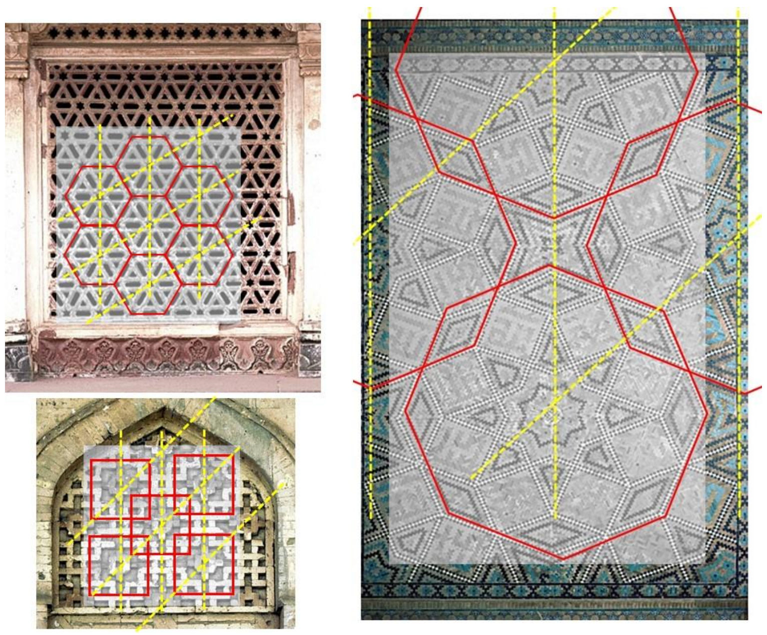


Figure 5.3 Tessellation in the hexagonal grid [25].

5.3 Features of Islamic Architecture

The most common features in Islamic architecture are:

Jali, meaning ‘net’, is a lattice formed from either calligraphy or geometry and is used in producing windows or arches. They were commonly found in the Indian region during the time of the Mughal Empire. Originally formed from carving stone, they act as air vaults, lowering the temperature inside a building. The air is compressed through the holes [24].

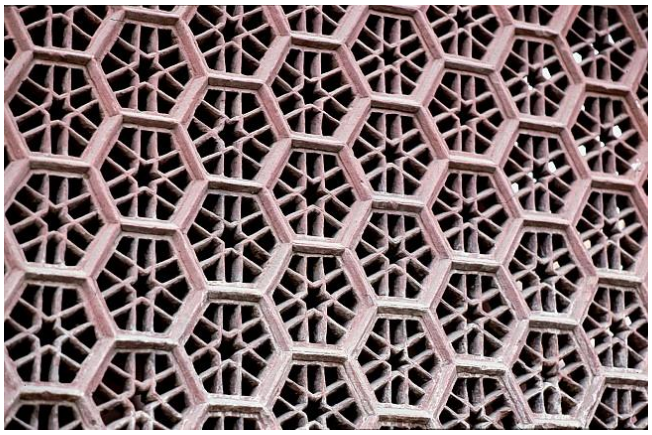


Figure 5.4 Monument – Red Fort (India) [25].

A pillar or a column is a cylindrical architectural feature used to support a roof or an entire building. Many architectural buildings in early Islam were built with classical columns that were spolia, architectural elements that were reused from earlier Christian buildings.

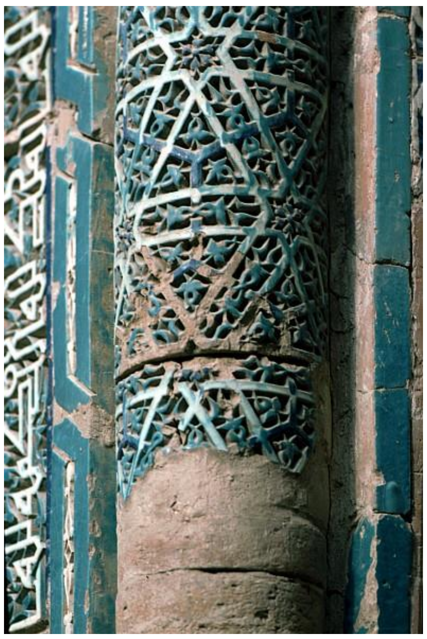


Figure 5.5 Shakh-i-Zindeh complex (Transoxiana) [25].

The dome is the striking hemisphere that stands out as a central feature found on mosques and mausoleums. Domes in the Islamic era were first formed during the Fatimid dynasty in the 10th century. The domes found in Cairo, or otherwise known as the *'Domes of Mamluk Mausolea'* are the certain type of domes that have been lavishly decorated with either arabesque or geometric patterns.

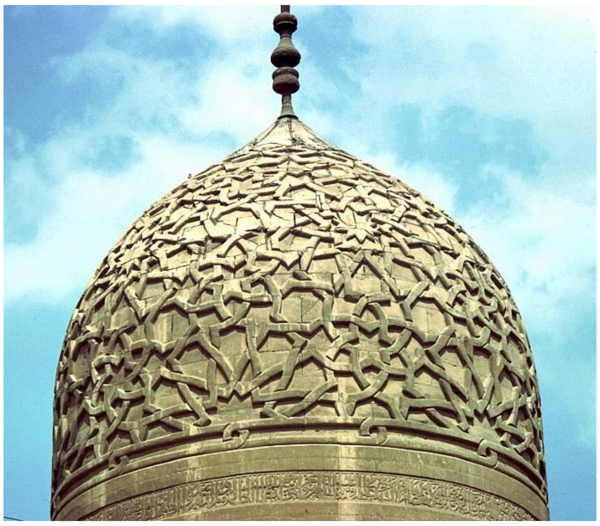


Figure 5.6 Dome of Mosque of Barsbey (Egypt) [24].

The Mamluk sultuns who ruled about 250 years between 1250 and 1517 AD were prominently known by their tombs, which were built during their lifetime [92]. Every proceeding ruler exceeded the size of the dome of the mausoleum who ruled before them, forming twice or three times the size of the earlier domes. The decoration style of the Mamluk domes was formed through repeat units, i.e. patterned tiles that created a sphere. The usual shape of the tiles were square, hence following on from the multiples of 4 (8, 16, 32, 64) that based their design [92].

Fountains are a source where water can be found. They are also used to an ornamental place, where the fountain was decorated by the craftsmen who built the architecture building. Romans were the first to decorate their fountain areas with bronze or stone masks of animals. This decorative style was then adopted by Moorish and Muslim garden designers who applied it to form smaller versions of the gardens of paradise. The decoration on a fountain is of concentric circle repetition where the pattern is radially repeated (Figure 5.7).

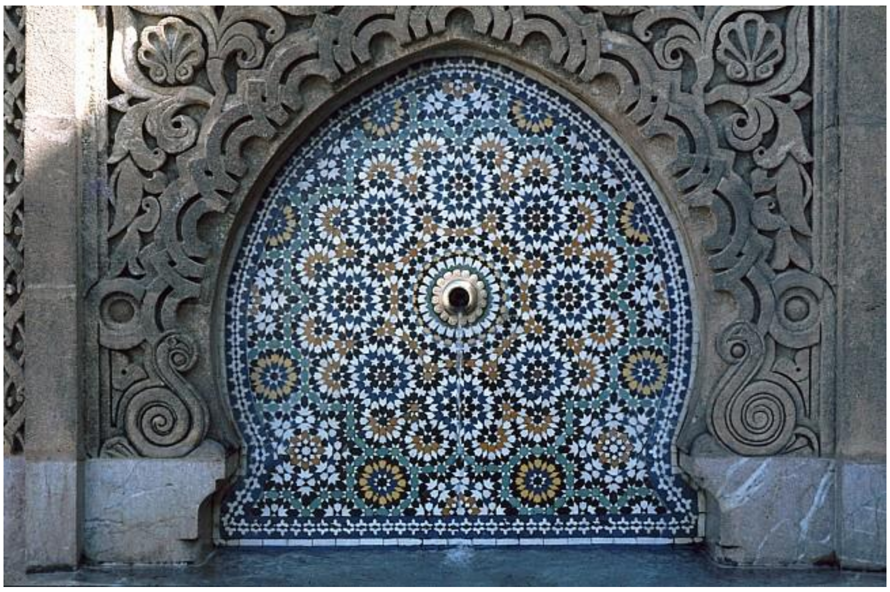


Figure 5.7 Rabat Mosque (Morocco) [25].

Deconstructing each feature, we can derive the common shape grammar rules, i.e. translation, rotation and scale in all three axes, x -, y- and z -axis.

5.4 Proposed Framework: IGMBS continued

5.4.1 Auto-Parameterized Shape Grammar (APSG)

The APSG rules are applied to construct the 3D IGMBS. These structures are constructed from the motifs by manipulating them in a 3D space, generating a structural surface; flat or curved. These rules are auto-parameterized. This means that the parameters of the rules are calculated automatically according to the user’s requirements. Again, the rules are based on Euclidean transformations including translation, rotation and scaling. In general, the more complex the structure the more rules are applied.

5.4.2 Hierarchy of features

The tree diagram (Figure 5.8) shows the relationship between the two grammars, and the hierarchy of each structure. The more rules applied, the more complex the structure. For a user to construct structures of a required size, specific algorithms are applied to model them. Each algorithm has a number of equations associated with it.

5.5 Methodology

The next section consists of the methodology that forms the wall, the column, the dome and the self-similarity star with given algorithms.

5.5.1 The Wall

As the shape grammar rules are designed to create *n*-fold geometry, with an IS of a regular polygon, both the square and hexagonal grids are used to tessellate the generated motifs in 3D space. The use of both grids allow a variety of structural designs of varying pattern complexity. For the square grid, a simple translation in the x- and y-axis is applied, multiplying the number of motifs translating horizontally and vertically by their width (w_1) and height (h_1).(vector equations ((5.5), (5.6)), respectively (Figure 5.9).

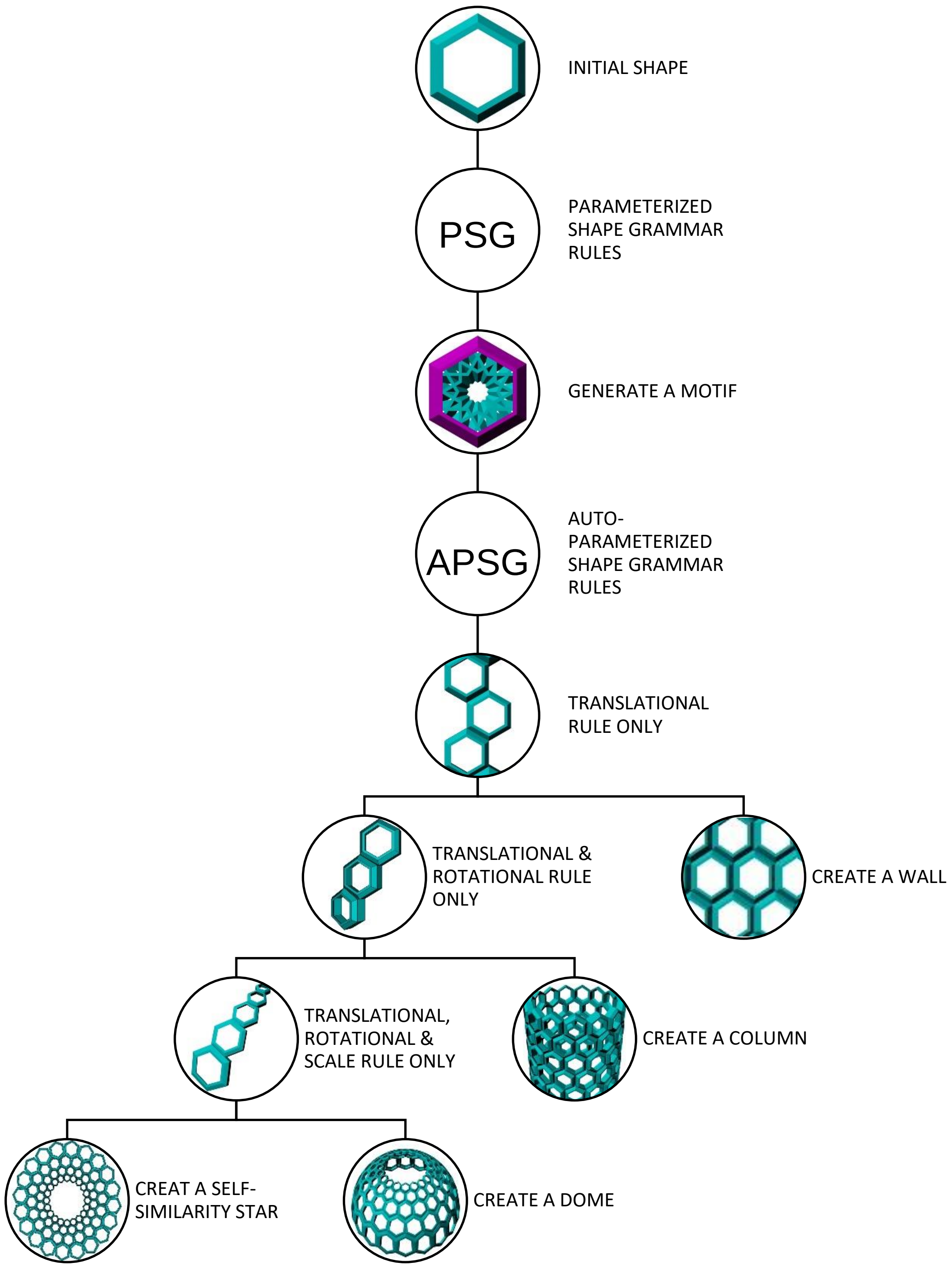

INITIAL SHAPE
PSG
PARAMETERIZED SHAPE GRAMMAR RULES
GENERATE A MOTIF
APSG
AUTO-PARAMETERIZED SHAPE GRAMMAR RULES
TRANSLATIONAL RULE ONLY
TRANSLATIONAL & ROTATIONAL RULE ONLY
CREATE A WALL
TRANSLATIONAL, ROTATIONAL & SCALE RULE ONLY
CREATE A COLUMN
CREAT A SELF-SIMILARITY STAR
CREATE A DOME

Figure 5.8 Hierarchy of Motif-Based Structures.

The motif is translated in the square grid horizontally, S_x, by vector equation (5.7) and vertically, S_y, by vector equation (5.7), (Figure 5.9).

$$S_x = (n_x(w_1, 0)), \tag{5.5}$$

$$S_y = (n_y(0, h_1)), \tag{5.6}$$

where n_x and n_y denote the number of motifs used to tessellate in the x - and y-axis respectively.

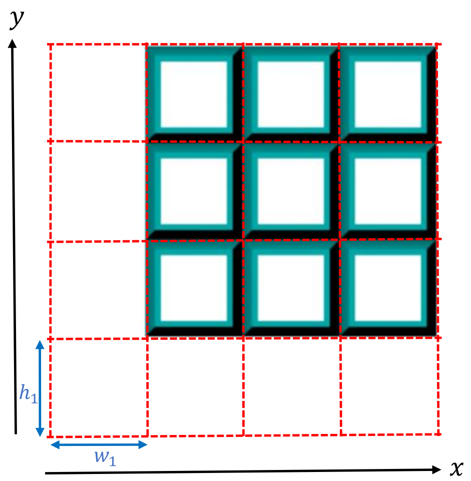


Figure 5.9 Translation in the square grid.

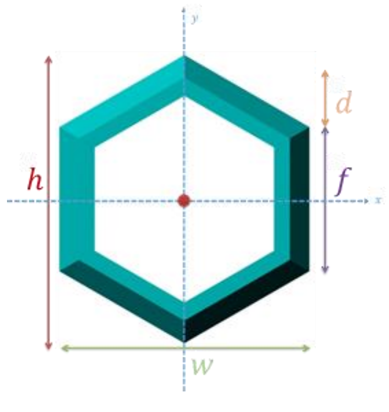
The hexagonal grid is however slightly complex. The hexagons are positioned in a staggered fashion, creating a diagonal line as shown in Figure 5.10 (right). The tessellation creates no gaps generating a periodic pattern as the hexagon is one of the fundamental shapes. For a motif to tessellate in the hexagonal grid, the offset translation is applied by vector equation (5.8) for the vertical translation in the y-axis with the following coordinates where $x = \frac{w_2}{2}$ and $y = (f + d)$. The shape parameters are defined as (Figure 5.10 (right)):

- h_2 is the height of the hexagon
- w_2 is the width of the hexagon
- f is the length of the side of the hexagon = ½ (h_2)
- d is the height difference from the length = ½ $(h_2 - f)$

The motif is translated horizontally by vector equation (5.7).

$$T_x = (w_2(n_x, 0)), \qquad (5.7)$$

$$T_y = (x(-1)^{n_y}, y), \qquad (5.8)$$



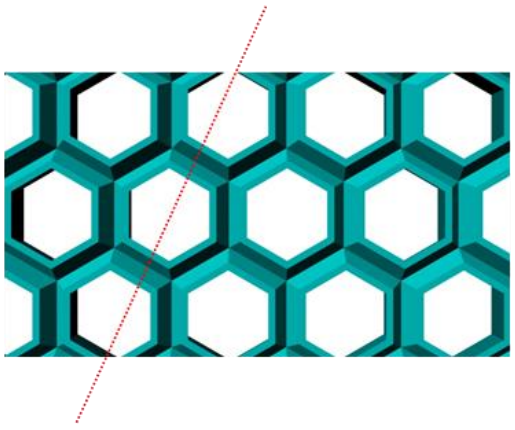


Figure 5.10 (Left) Parameters of the hexagon (Right) Hexagonal grid.

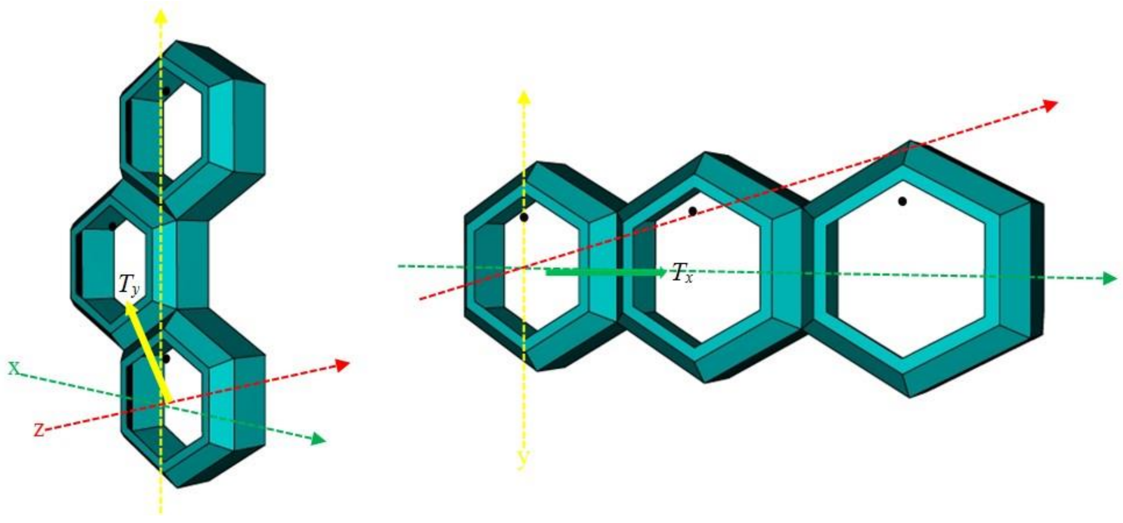


Figure 5.11 Translating the motifs to tessellate periodically in a hexagonal grid, creating a wall.

5.5.2 The Column

The column is of cylindrical shape, hence the structure has depth as well as height and width. Again, there is a difference in creating a square and hexagonal grid to form the column shape but the underlying principles are the same. Like the wall, the motif is tessellated vertically for the square grid, with the fixed translation, vector equation (5.6), but it is also translated in the z -axis, equation (5.9). This forms the radius of the column which is calculated from the equation of the circumference of a circle,

$$r_c = w_i(n_x)/2. \tag{5.9}$$

To create the curvature of the column a rotational angle is applied to the stack of vertical motifs. This is calculated by dividing 360º by the number of motifs translated horizontally (equation (5.10)). The stack is then rotated and duplicated accordingly (Figure 5.13 (left)),

$$\theta = 360/n_x. \tag{5.10}$$

To create the hexagonal grid, the initial steps are the same, taking the height of the motif as the calculated coordinate y. However, to follow the periodic tessellation of the motifs to tessellate in the hexagonal grid, another rotational rule is applied for the motifs to tessellate offset vertically, where the rotation is half of θ, μ (equation (5.11)),

$$\mu = 180/n_x. \tag{5.11}$$

This creates a spiral of n_y motifs tessellated vertically offset, which are then combined and duplicated n_x times to tessellate horizontally (Figure 5.13, right).

The algorithm to create the column is:

1) *Translate the motif in the z-axis; this creates the radius of the column.*
2) *Translate and duplicate the motif in the y-axis, to tessellate the motifs vertically.*
    i) *Rotate in the y-axis from the centre, for the motifs to tessellate the offset (hexagonal grid).*
3) *Combine, duplicate and rotate in the y-axis from the centre to form the complete column.*

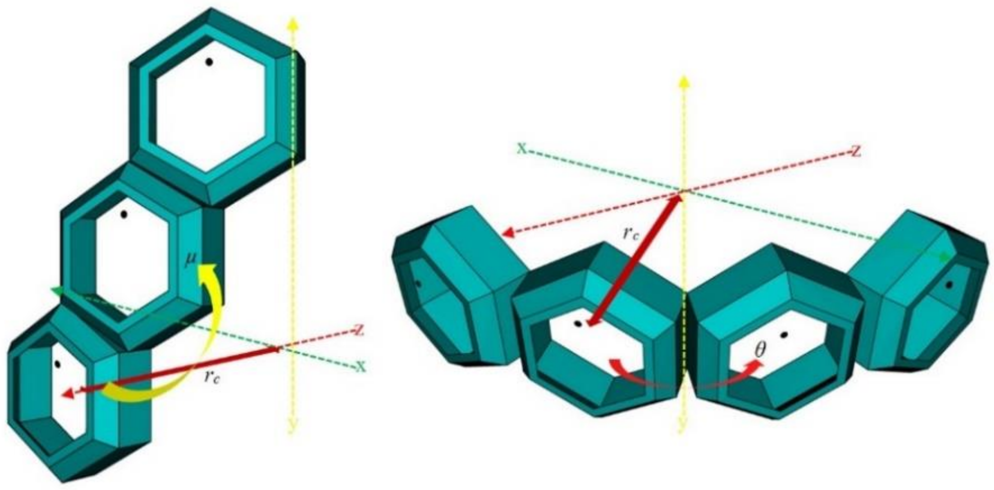


Figure 5.12 Translation and rotation of the motifs to tessellate periodically to construct a column.

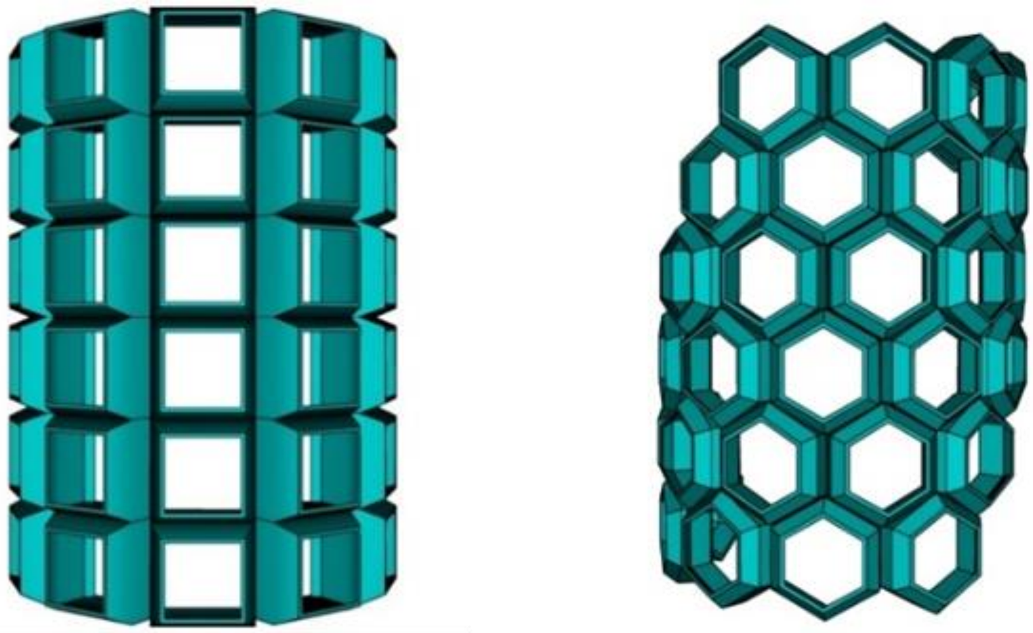


Figure 5.13 (Left) Square grid (Right) Hexagonal grid applied to form the column struture.

5.5.3 The Dome

There are various types of domes constructed in Islamic Architecture. In this research the construction of a simple dome, a hemisphere, will be analyzed. This will generate a foundational algorithm which can be varied in the future to construct different styles of domes.

The special characteristic of a dome that is adorned by IGP is the central star that is generated at its apex (Figure 5.6). This is a key aspect that is applied in this algorithm. The approach taken to construct the dome is different to the wall and the column. The wall and the column had a linear progression horizontally and vertically, whereas the dome varies from the top to the bottom. To generate the algorithm for the dome two methods were approached, firstly the packing of space and secondly proportionally scaling to fill space. The packing of space required the use of additional generated motifs to fill in the gaps, as seen in Figure 5.14 (left), whereas the proportional scaling method, scales the motifs to fill in the gaps Figure 5.14 (right). For the purpose of using the same number of motifs, the latter method was adopted. Proportionally scaling the motifs will in theory create self-similarity, which is a feature in IGP.

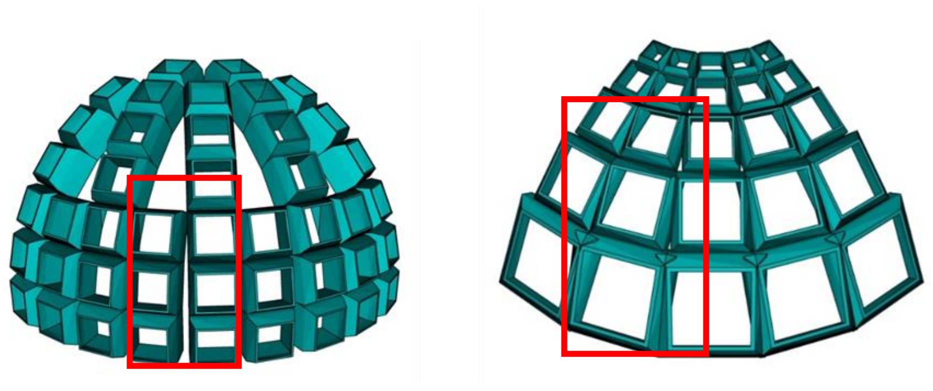


Figure 5.14 (Top) Packing of space method (Bottom) Proportionally scaling method.

Using the design of the apex star of the dome as an initial starting point, translate, rotate and scale transformations can be applied to the motifs. The motifs however tessellate from a single radial point (centre (0, 0, 0)), to create periodicity. This generates radial symmetry within the structure. The algorithm for the dome structure uses a number of radii, all of which are calculated using the equation of the circumference. Figure 5.15 refers to the algorithm described below.

The algorithm to construct a dome is initiated with the radius of the initial star, r_s, that is generated at the apex of the dome,

$$r_s = wn_s/2\pi, \tag{5.12}$$

where n_s is associated with the number of star-points of the initial star, i.e. the number of motifs to create the star at the apex of the dome and w is the width of the motif. This applies to both the square and hexagonal grid. The scale factor to proportionally scale the motif, in the x- and z-axis, is calculated using the following equation,

$$s = \pi(2r_s + y)/wn_s - y\pi, \tag{5.13}$$

where y is the height of the motif that was calculated previously for the construction of the wall and the column.

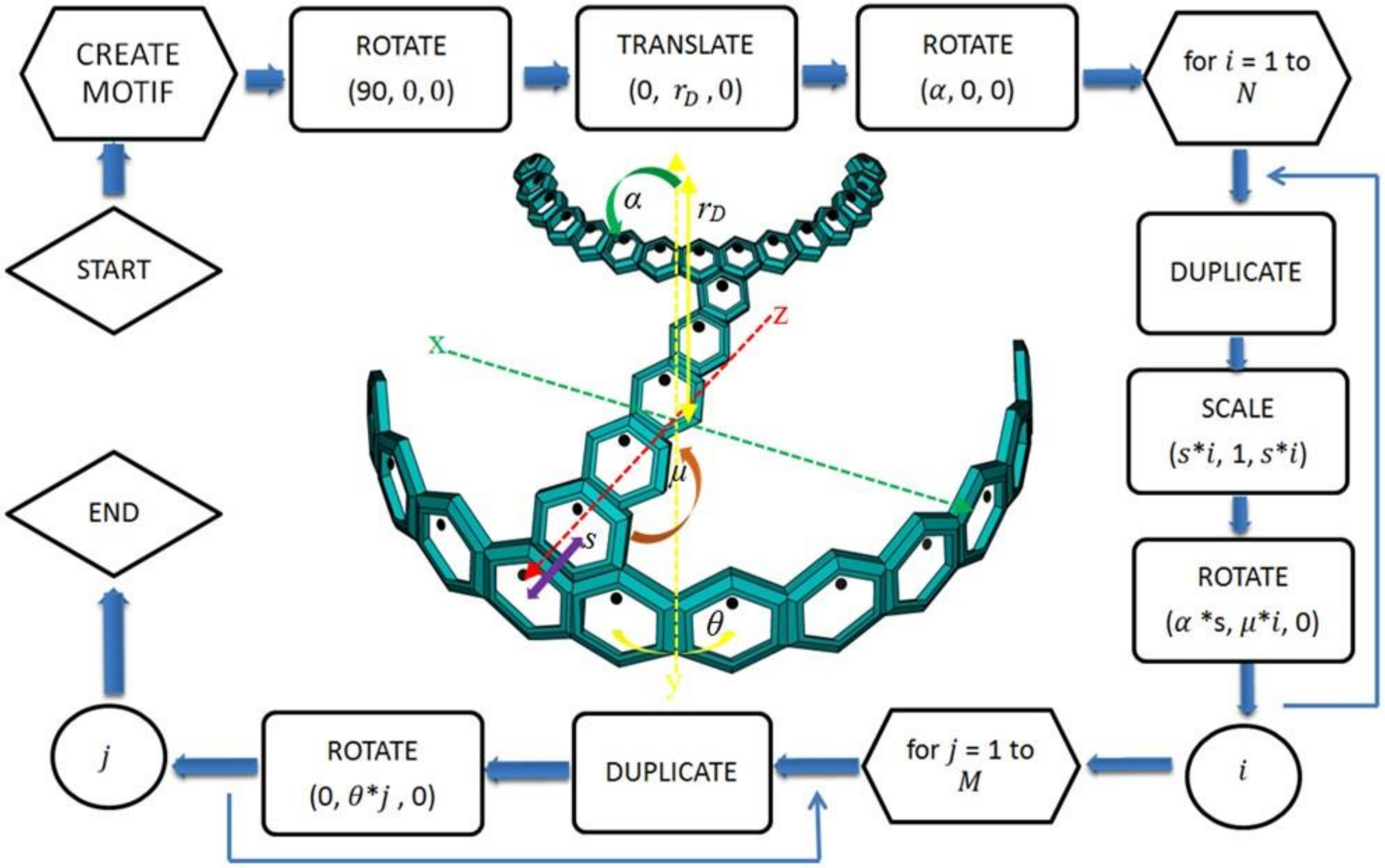


Figure 5.15 (Centre) The structure of the dome. (Outside) Flowchart to construct a dome in the hexagonal grid.

As the motifs are rotated and scaled 90° from the top to the bottom, this portion of the dome only covers a quarter of the complete sphere, hence the ratio of the number of levels, L, in a dome in relation with the number of star-points, n_s, is 1:4. This gives the equation,

$$L = n_s/4. \tag{5.14}$$

The motif is translated in the y-axis to the top of the dome, i.e. by a distance equivalent to the radius of the dome (Figure 5.15, yellow arrow), r_D , which is the same as the radius of circle of motifs generated at the bottom level of the dome, hence,

$$r_D = \left(\frac{nw}{2\pi}\right) s^{L-1}. \tag{5.15}$$

Lastly, the initial angle,α (Figure 5.15, green arrow), of the first motif is calculated by the cosine rule,

$$\cos(\alpha) = \frac{2r_D^2 - r_s^2}{2r_D^2}, \tag{5.16}$$

which is rotated about the x-axis. The scale factor is multiplied with the angle, α, to generate proportionally scaled motifs as it is rotated from the top to the bottom of the dome (Figure 5.15, purple arrow). With the rotation in the x-axis, which forms the curvature of the dome, μ (Figure 5.15, yellow arrow), which was calculated previously for the column, is again applied in the y-axis, to create the offset position in the hexagonal grid for the motifs to form the dome. This would fit the motifs in proportion to each other. To complete the dome, the last rule of rotation, θ (Figure 5.15, brown arrow), described previously is applied. This algorithm creates a similar spiral to the column but the motifs are scaled to proportion and rotated away from the centre to form the complete dome for the hexagonal grid.

The algorithm to create the dome is:

1) *Rotate 90º (motif faces the top view)*
2) *Translate in the y-axis to form the radius*
3) *Rotate in the x-axis with the initial angle from the centre.*
4) *Scale, rotate and duplicate to form a curved edge of the dome.*
    i) *Additional rotation in y-axis to form offset (hexagonal grid).*
5) *Combine, duplicate and rotate in the y-axis from the centre to form the complete dome.*

The flowchart shown in Figure 5.15, describes the steps taken to construct a dome in the hexagonal grid, where N and M calculate the motifs tessellating vertically and horizontally, respectively.

5.5.4 Self-Similarity Star

Self-similarity is a key characteristic in IGP. It generates the same shape by enlarging or shrinking it proportionally. The self-similarity star which is generated uses a similar algorithm to the dome. It scales the motifs to show the self-similar characteristic, using the equation of s, as described for the dome, but it scales in the x- and y-axis. The radius, r_s is taken as the radius of the inner star. Also, θ and μ (described previously) are the rotations used to create the star and offset position of the motifs for the hexagonal grid, respectively. However both rotations are rotated about the z-axis, as shown in Figure 5.16.

The algorithm to create the self-similarity star is:

1) *Translate in the y-axis to form the radius of the inner star.*
2) *Scale and duplicate to form a strand of motifs.*
    i) *Additional rotation in z-axis to form offset position (hexagonal grid).*
3) *Combine, duplicate and rotate in the z-axis from the centre to form the complete star.*

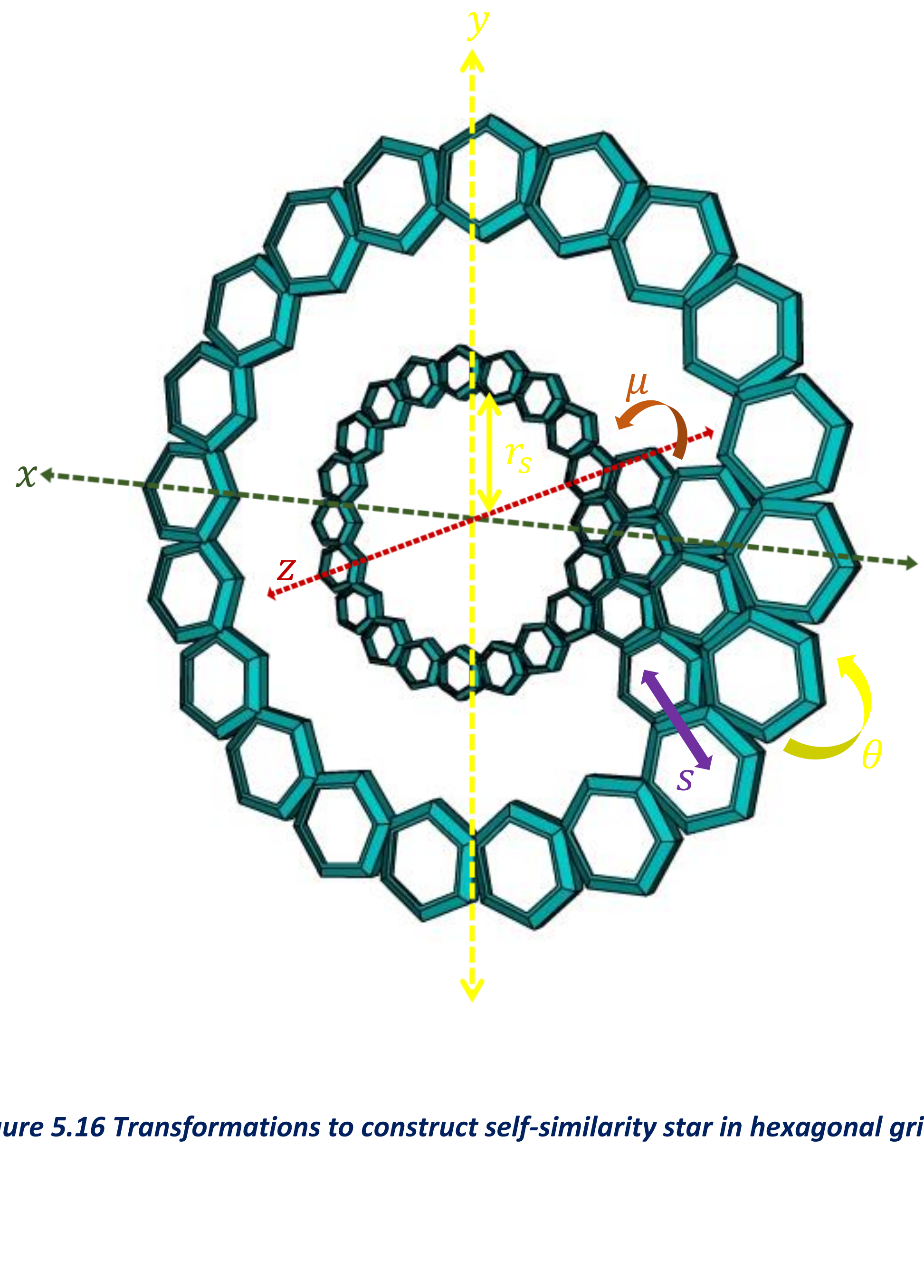


Figure 5.16 Transformations to construct self-similarity star in hexagonal grid.

5.6 Implementation

Using the generated motifs and applying them to the construction algorithms of their structures, one can construct motif-based structures. Figure 5.17 shows the GUI of forming 3D IGMBS.

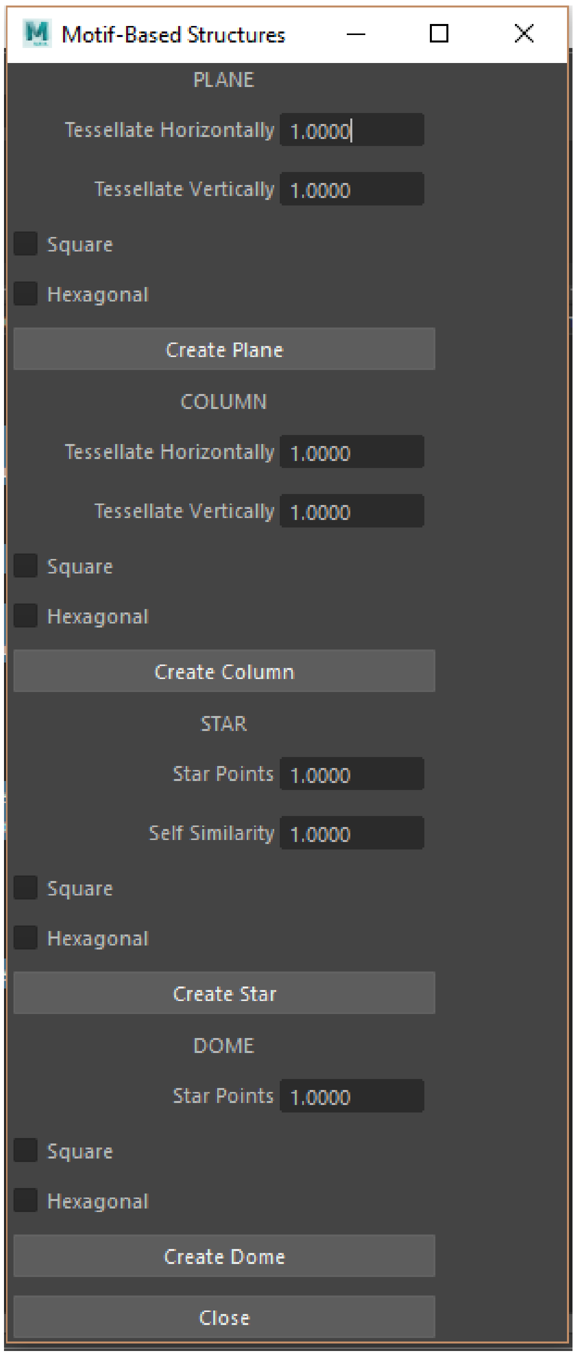


Figure 5.17 GUI to construct motif based structures.

Figure 5.19Figure 5.30, illustrates how the four structural algorithms are applied to a 3D generated motif to create the structures. Each structure has been rendered in three different views, either front, back, top, bottom or perspective. Figure 5.19Figure 5.21 shows the renders of a wall in front, perspective and back views respectively. The wall structure is formed from a hexagonal motif and the tessellation is done in a hexagonal grid. The same motif (Figure 5.18) has been applied to form each of the other three architectural structures, the column (Figure 5.22Figure 5.24), the dome (Figure 5.25Figure 5.27) and the self-similarity star (Figure 5.28Figure 5.30). Figure 5.22Figure 5.30 also shows the analytical view through zoomed areas of the renders. This will be explained in the Section 5.7.

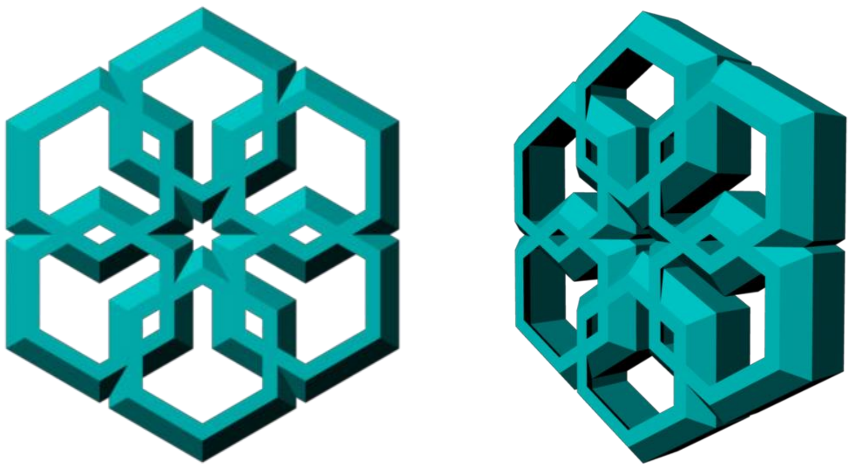


Figure 5.18 The 3D motif applied to form all the architectural structures (Left) Front view render. (Right) Perspective view render.

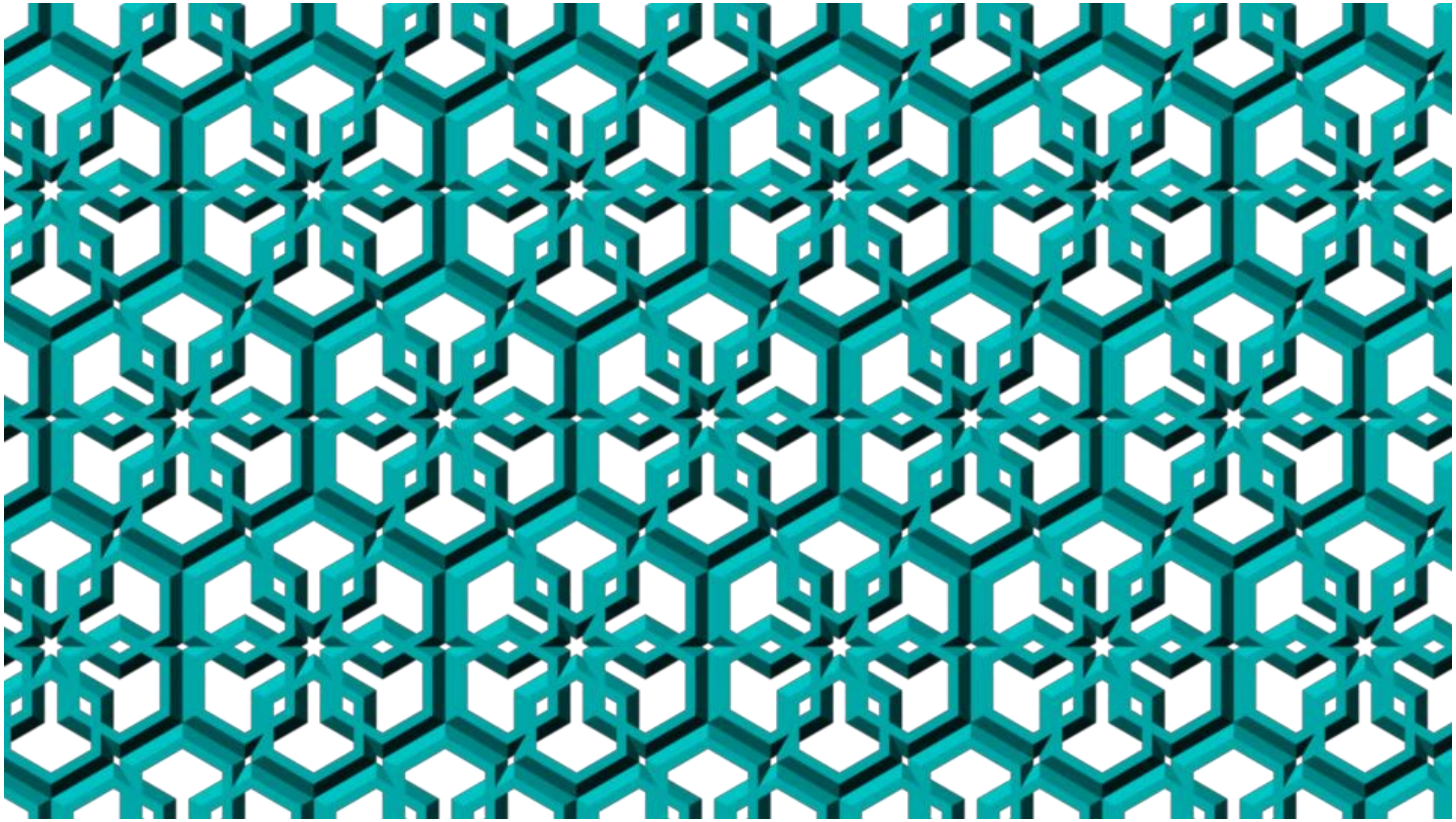


Figure 5.19 Front view render of the wall.

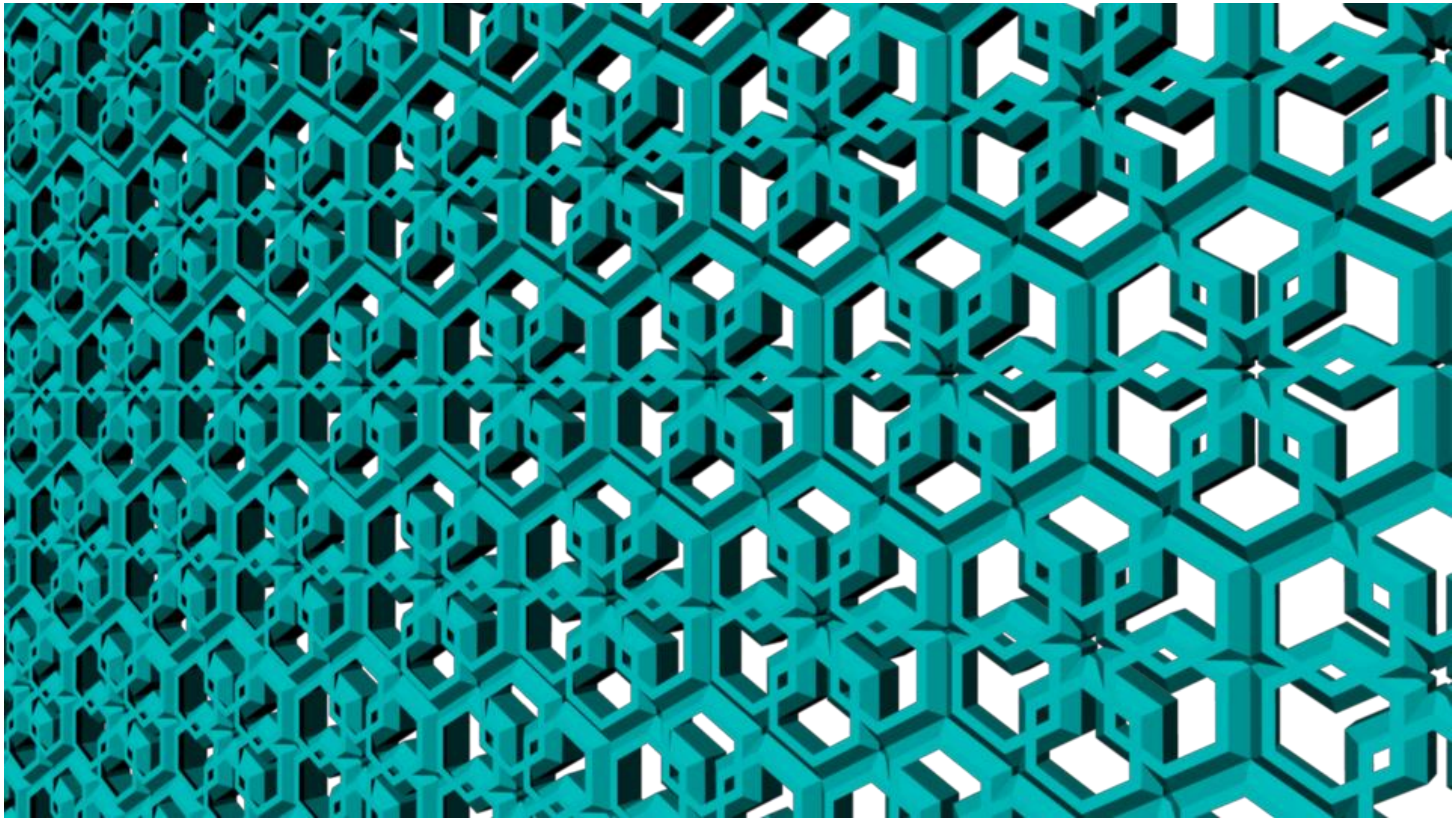


Figure 5.20 Perspective view render of the wall.

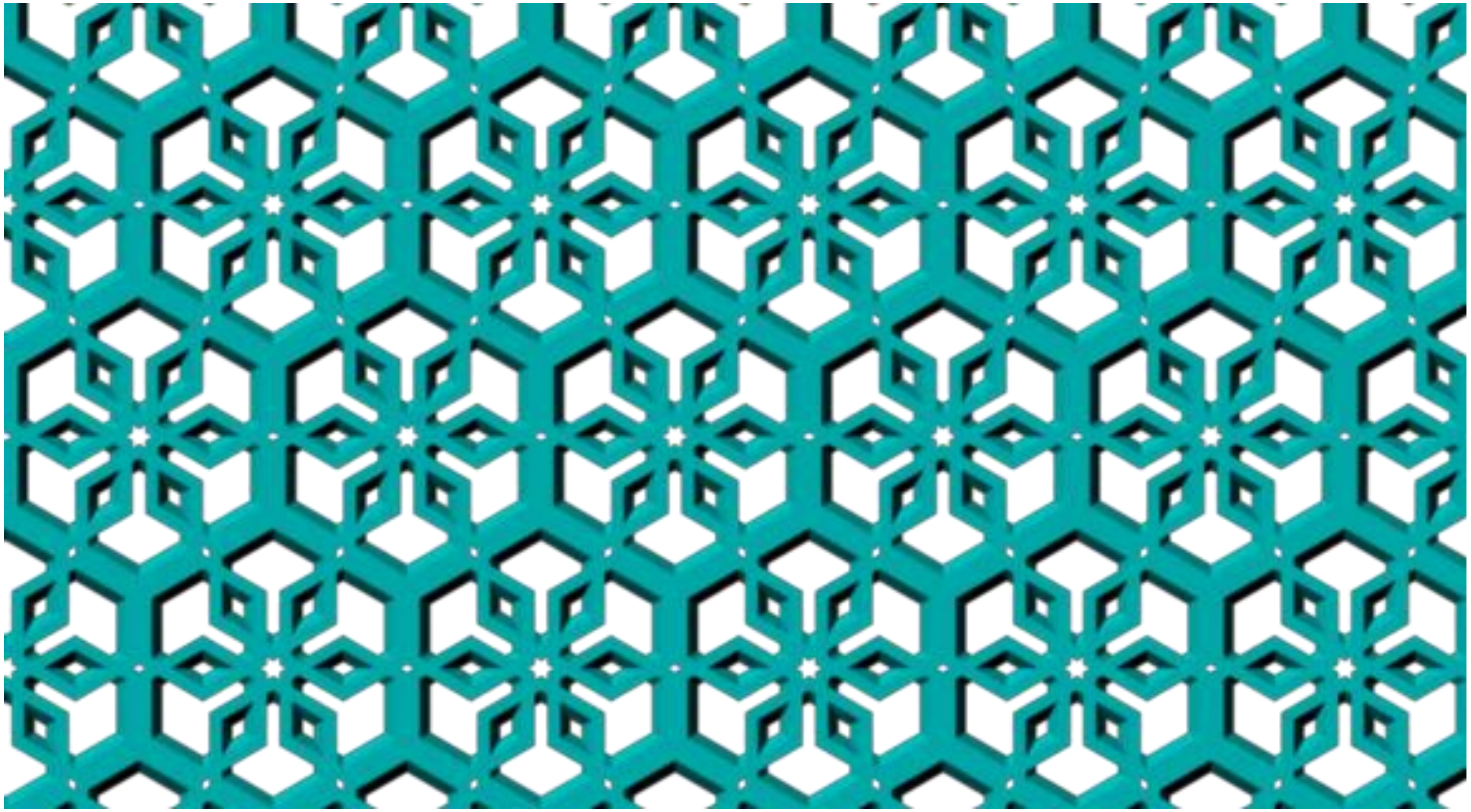


Figure 5.21 Back view render of the wall.

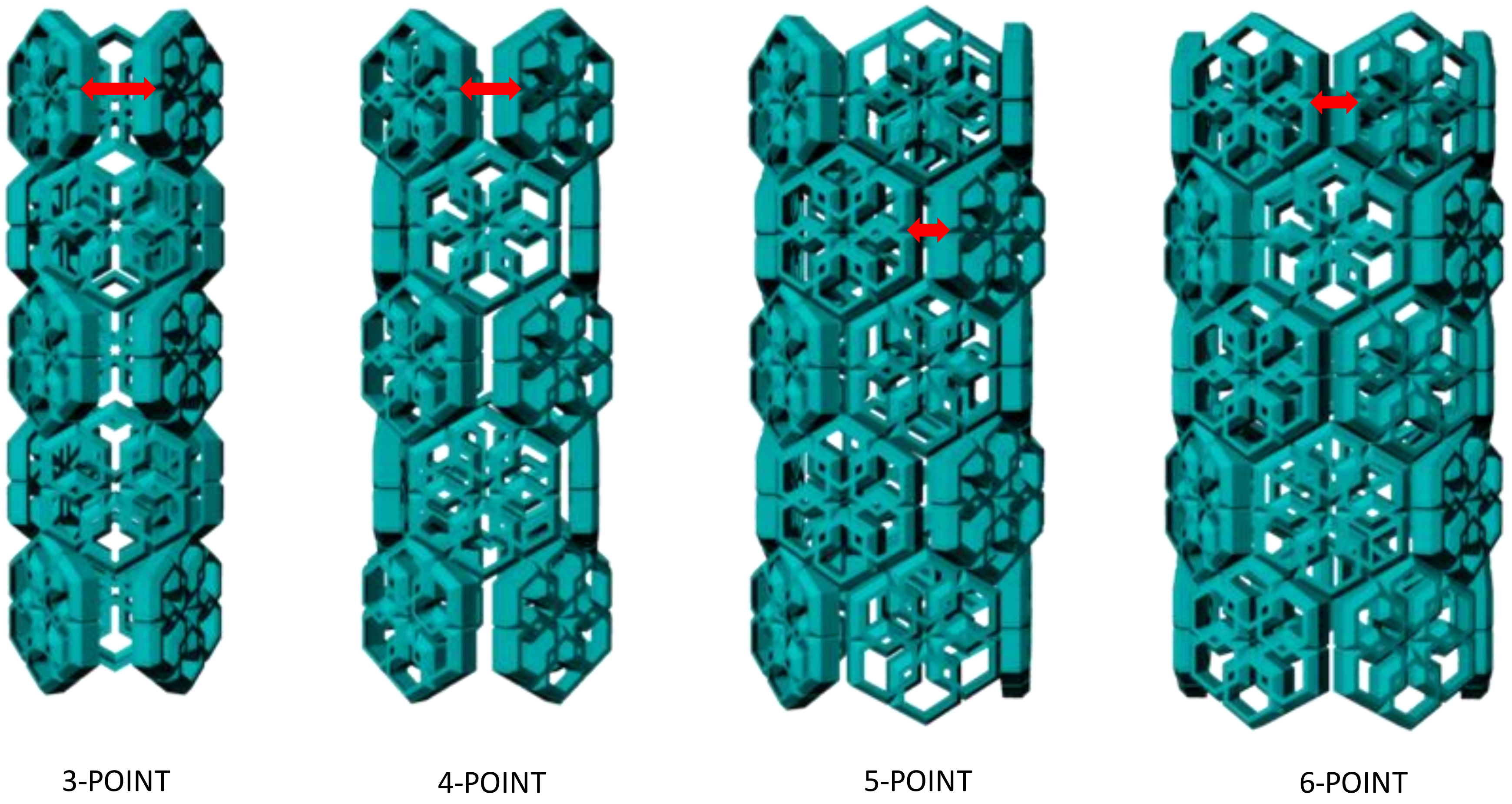
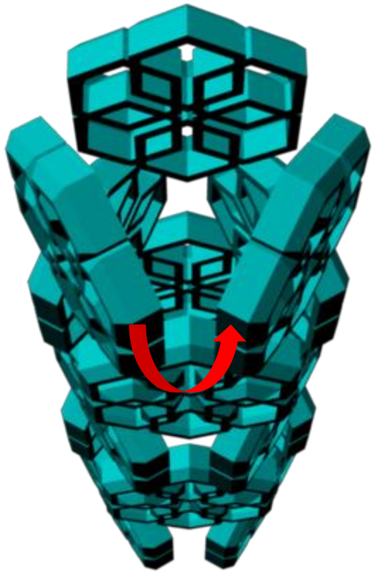
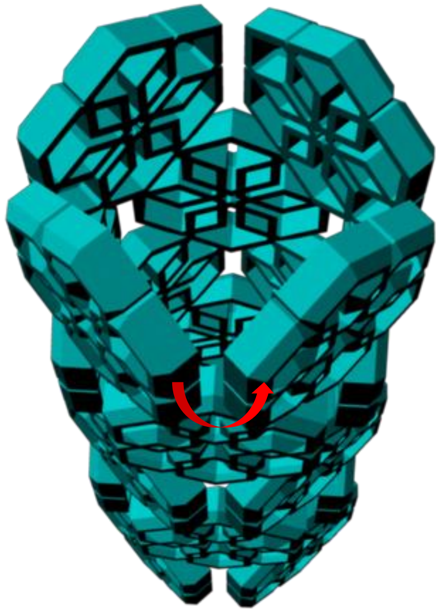


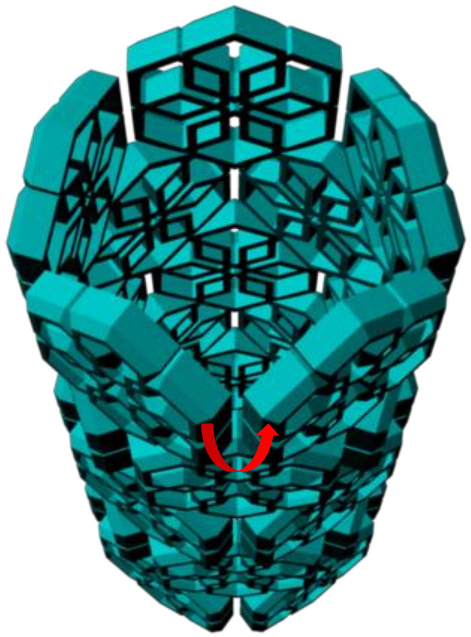
Figure 5.22 Front View renders of 3, 4,5 and 6 motif point columns.



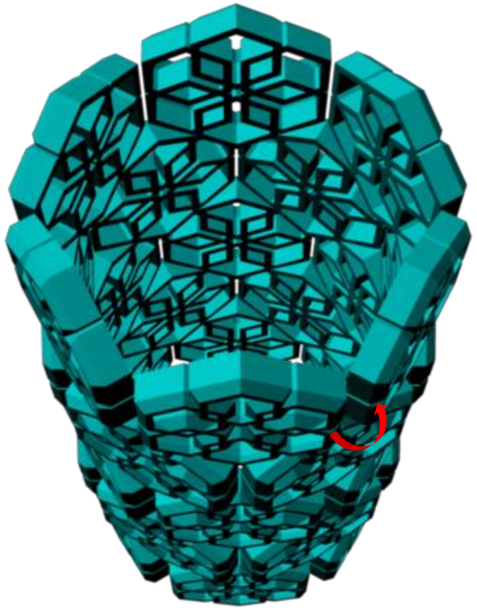
3-POINT



4-POINT



5-POINT



6-POINT

Figure 5.23 Perspective view renders of 3, 4,5 and 6 motif point columns.

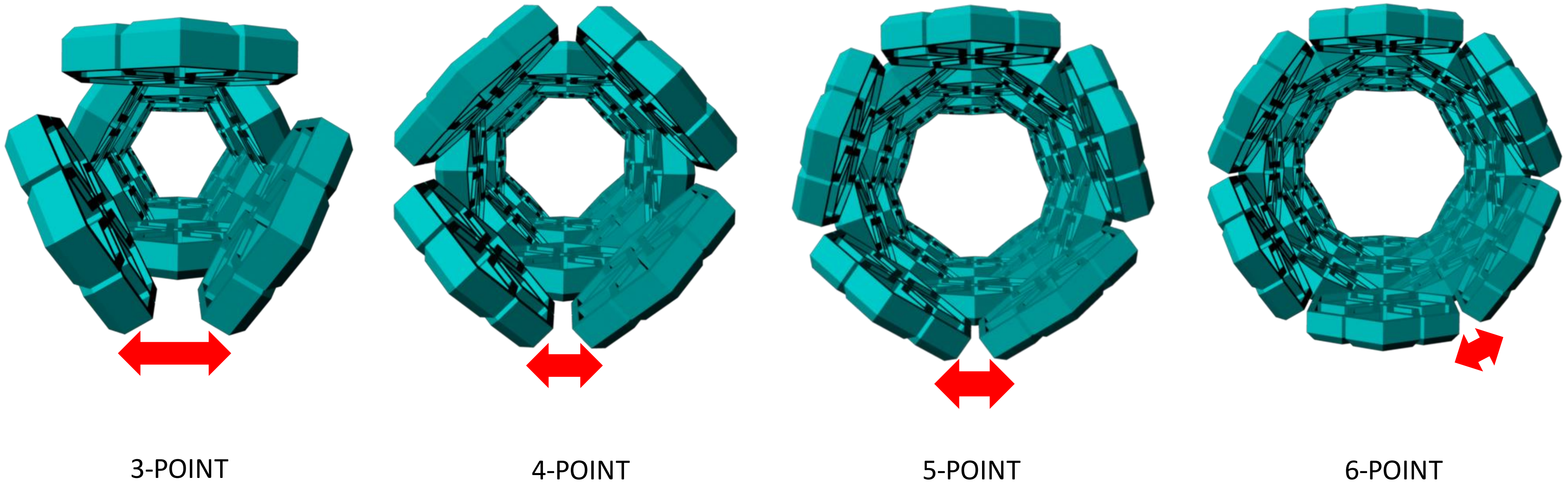


Figure 5.24 Top View renders of 3, 4,5 and 6 motif point columns.

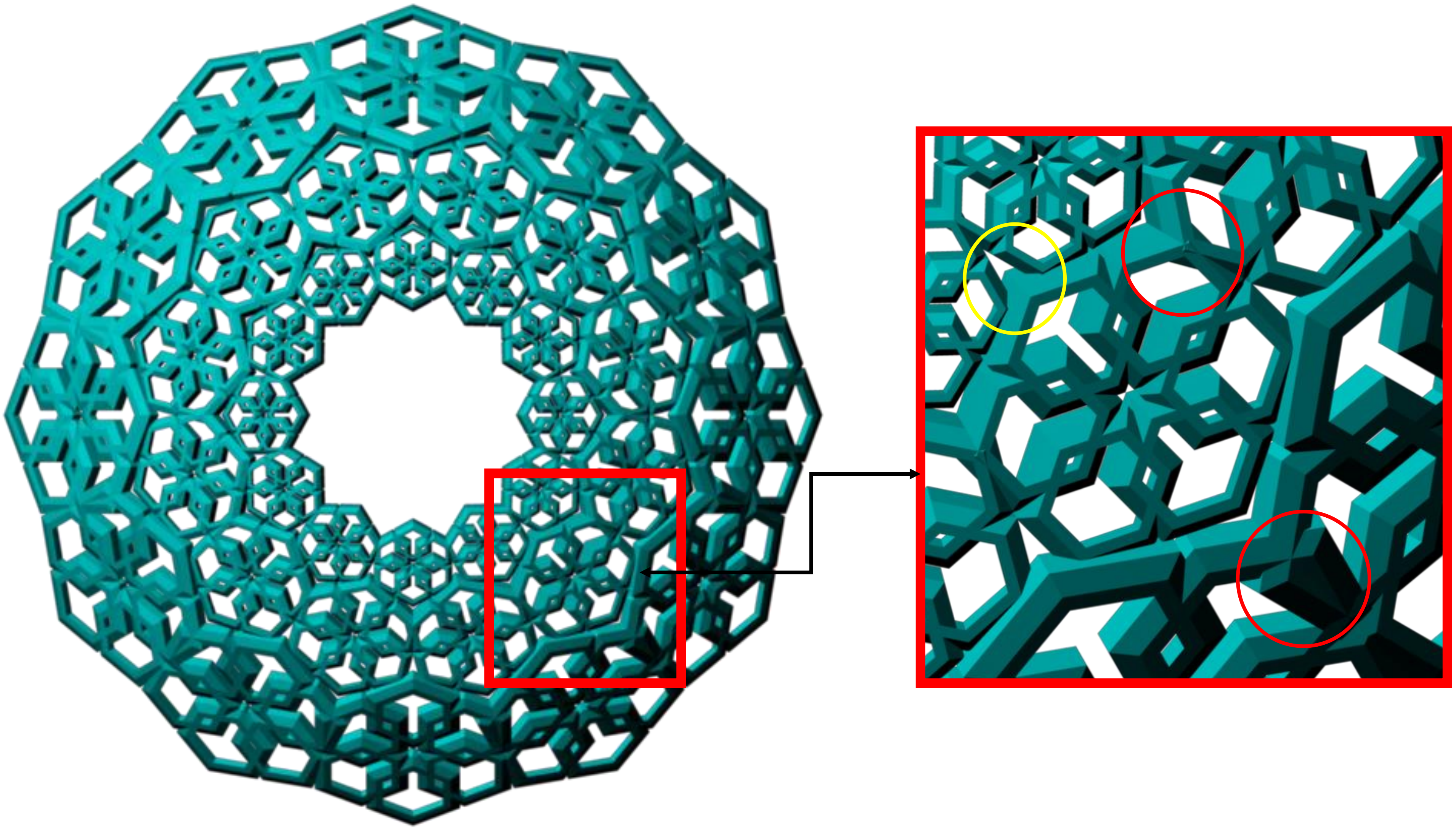


Figure 5.25 (Left) Top view render of the dome. (Right) Enlarged analysis image.

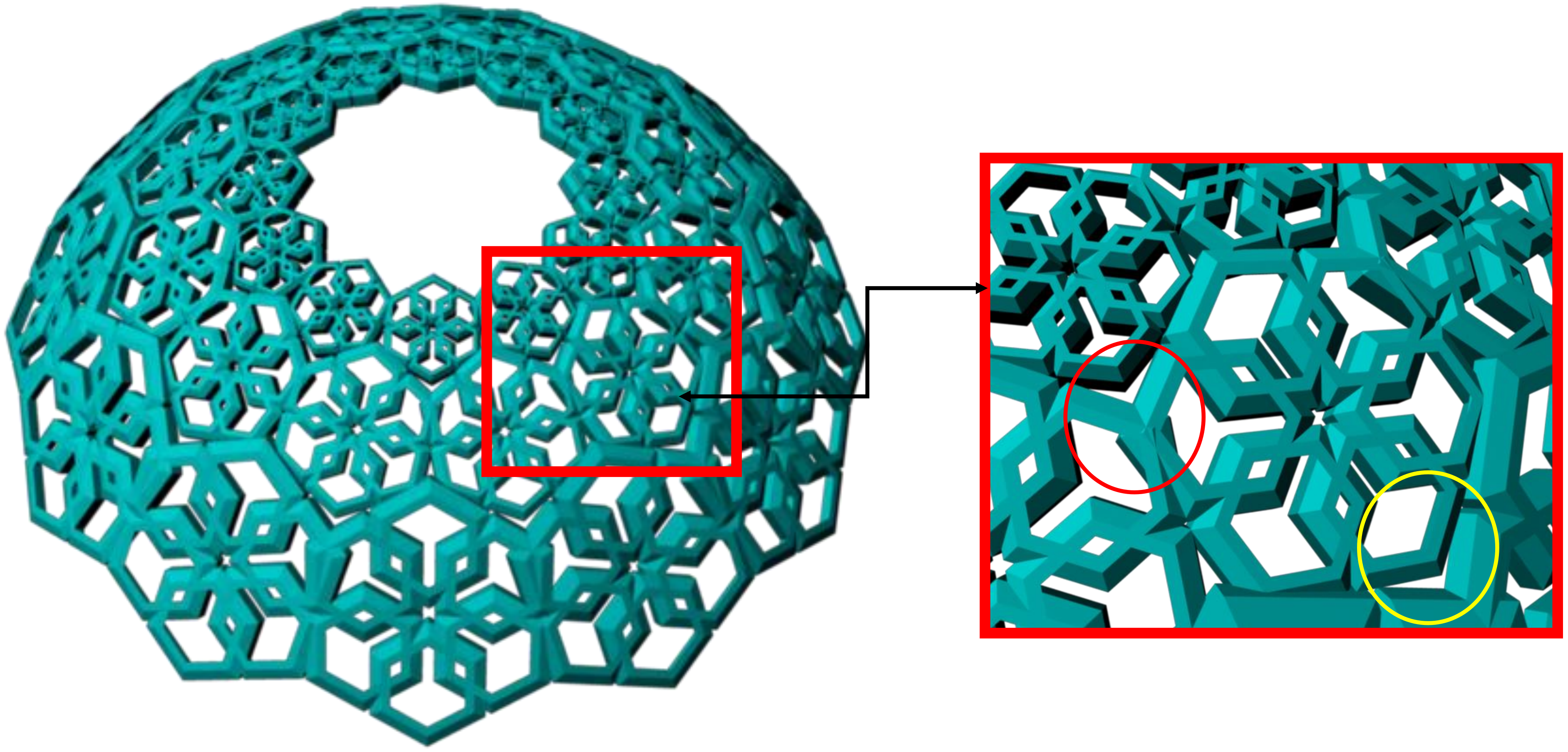


Figure 5.26 (Left) Perspective view render of the dome. (Right) Enlarged analysis image.

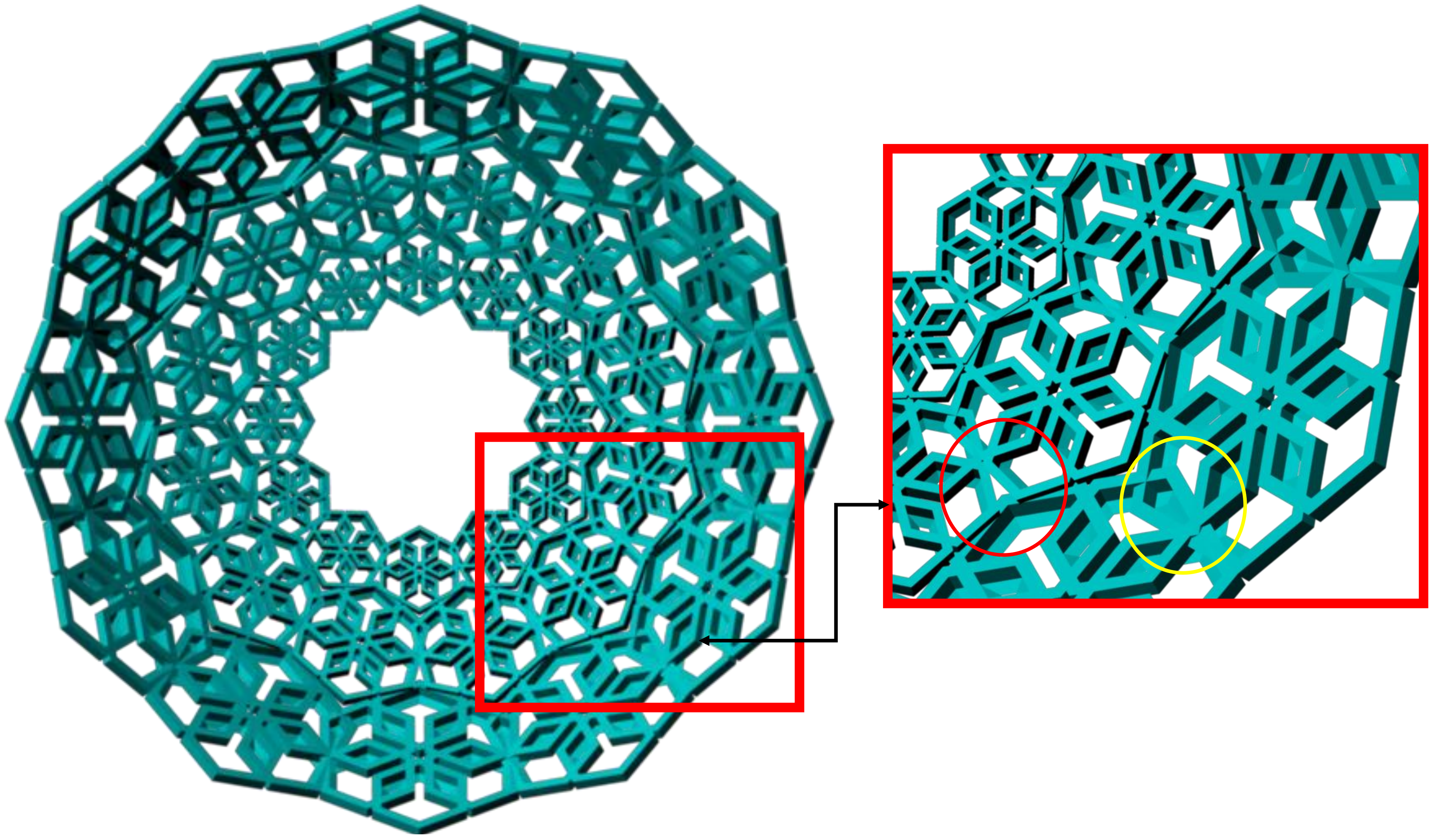


Figure 5.27 (Left) Bottom view render of the dome. (Right) Enlarged analysis image.

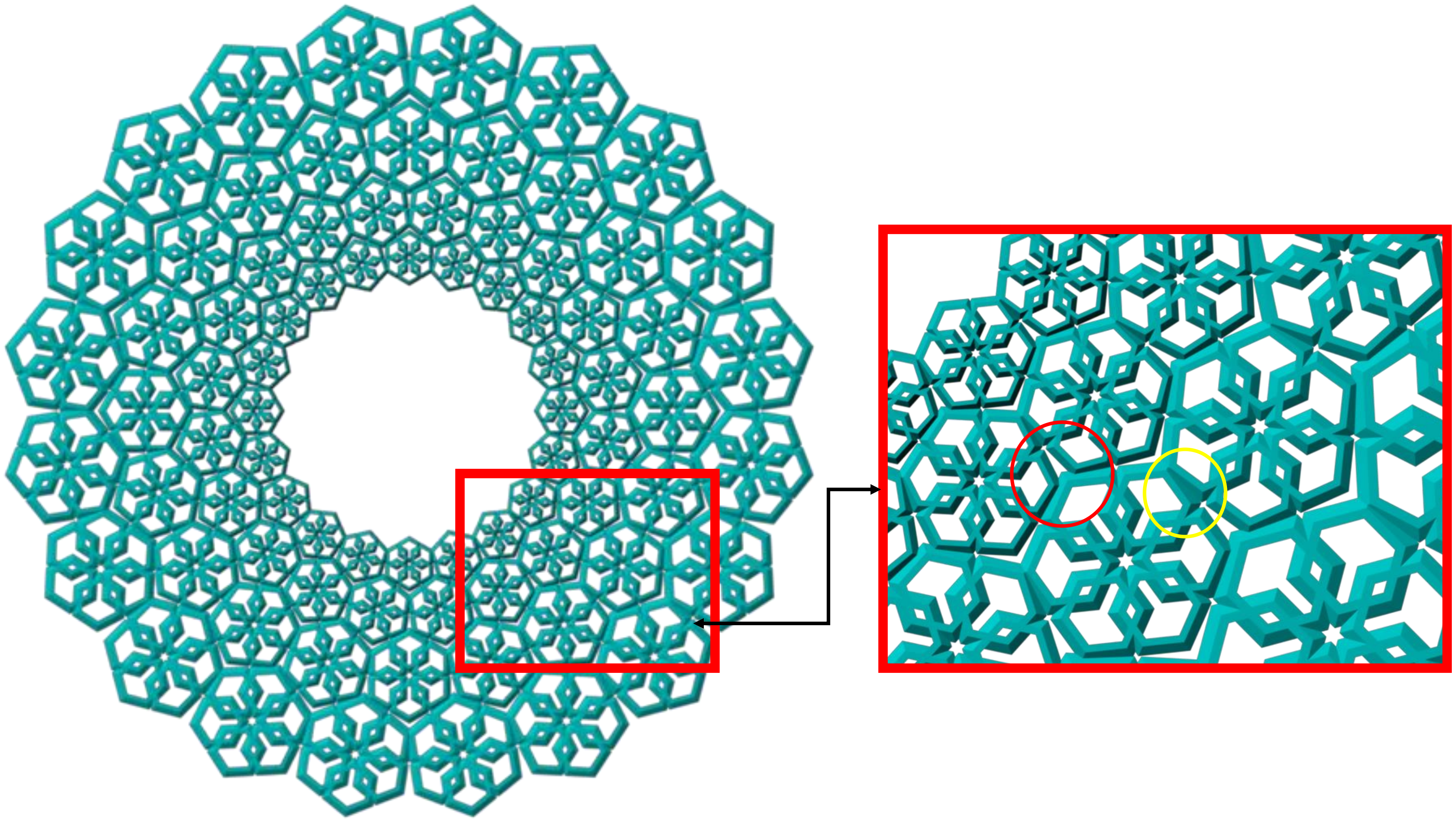


Figure 5.28 (Left) Front view render of the star. (Right) Enlarged analysis image.

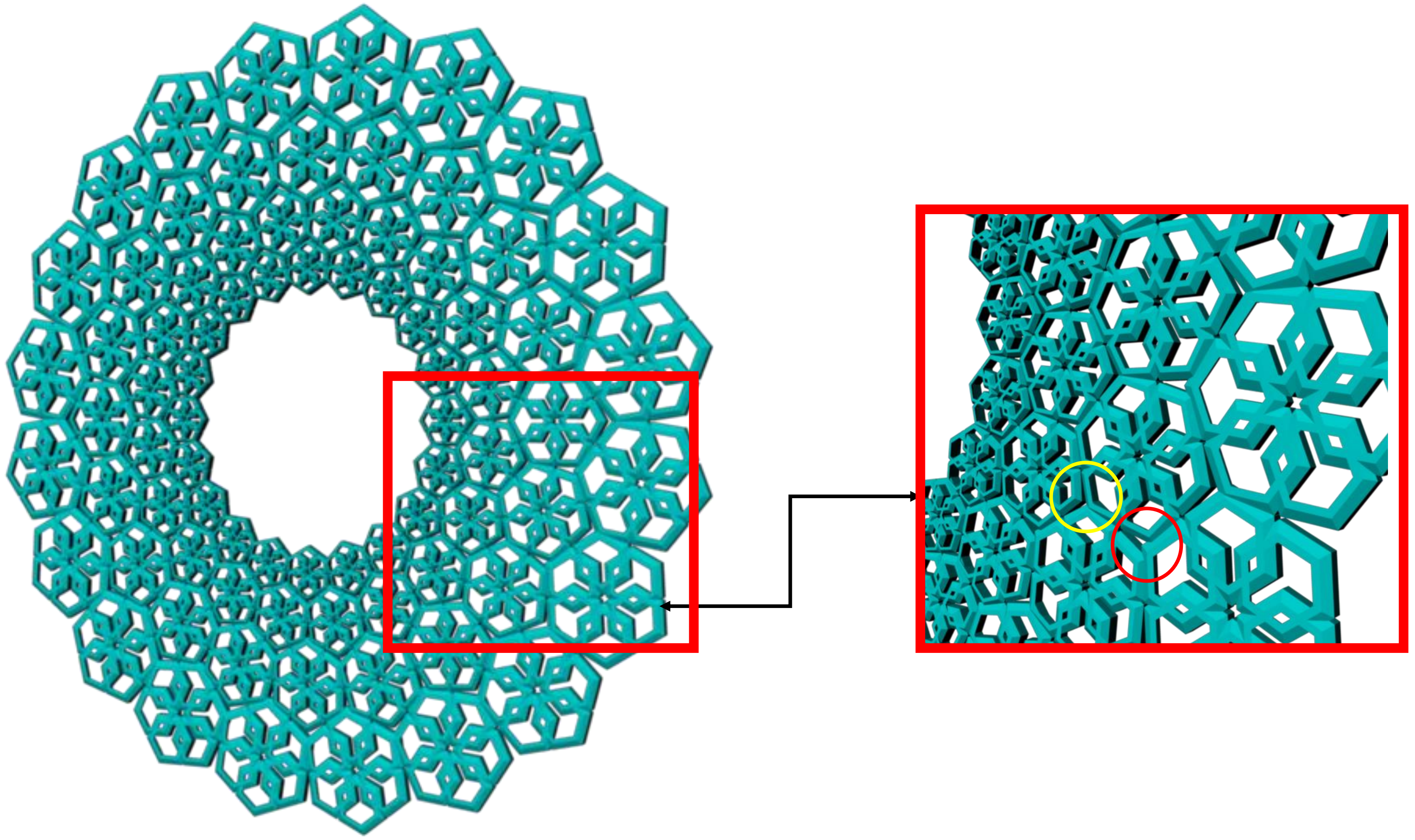


Figure 5.29 (Left) Perspective view render of the star. (Right) Enlarged analysis image.

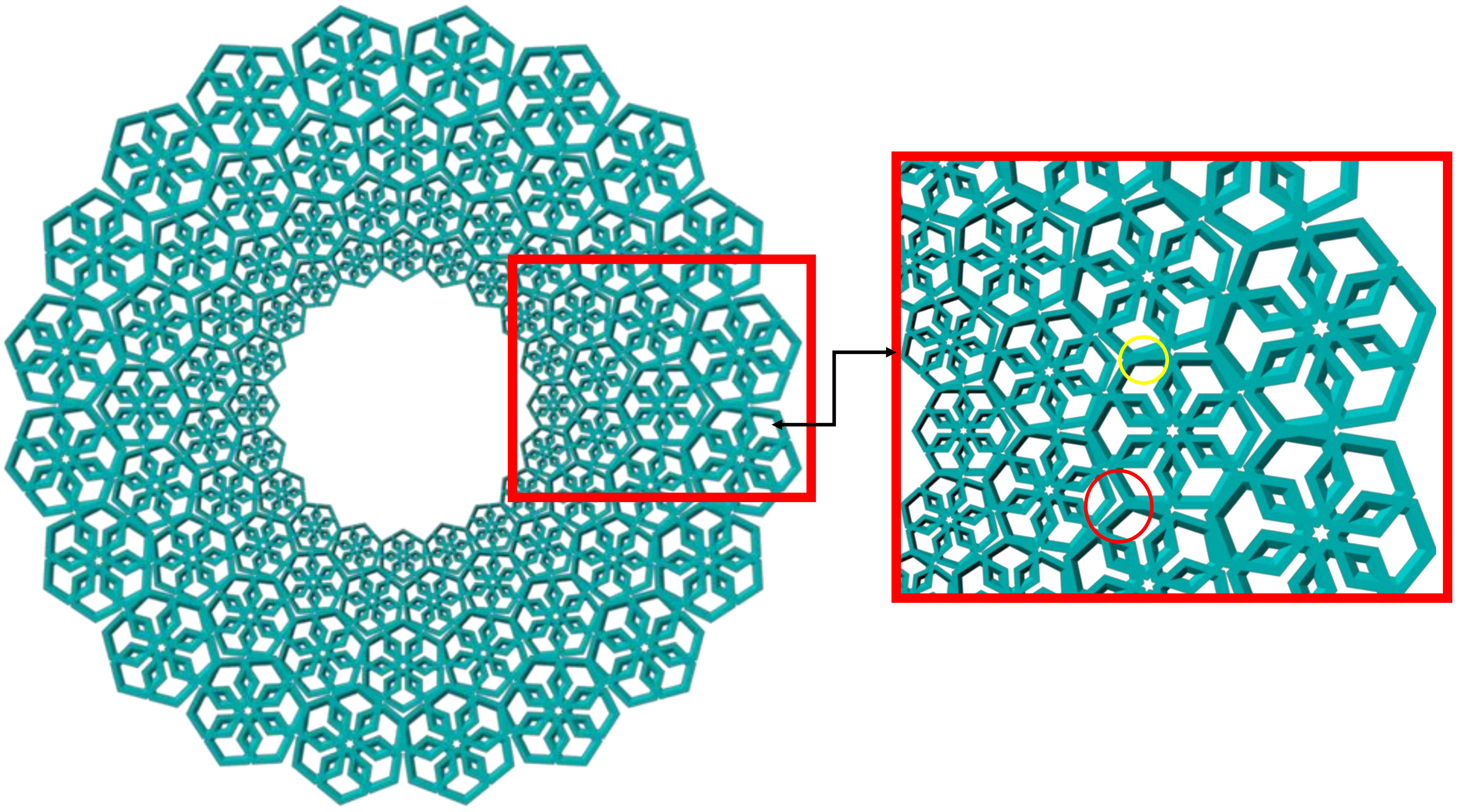


Figure 5.30 (Left) Back view render of the star. (Right) Enlarged analysis image.

5.7 Analysis

Previous methods that implemented shape grammar to form architectural building or ground plans, all methods worked in 2D space, with translation as their main transformation. The results shown in the previous section it can be seen that shape grammar can be used as a mapping method, and can form 3D surfaces without the use of underlying grids or surfaces. The methodology applied allows one to form four different structures, with two styles of grid mappings, either in the square grid or the hexagonal grid.

Although shape grammar has been seen as a mapping technique, the rules that are permitted within the grammar do not adhere with the definition of a tessellation. For a surface to be tessellated, the tessellation cannot form any gaps or overlaps. Figures 5.22-5.24 shows the renders of a column in the front, perspective and top views respectively. The algorithm to form the column was initiated by using the smallest number of motifs, 3, that would form a complete closed polygonal shape, i.e a triangle. However in each of the three different views, one can see huge gaps between each motif (red arrows). The figures also show renders of columns formed of 4, 5 and 6 motifs (the circumference of the column). The column formed of 6 motifs is the closest to forming a complete column as each of the other renders formed massive gaps between each other, lessening as the motif number increased. However the column formed of 6 motifs still has gaps as the edges do not join completely, forming rigid polygonal columns rather than a smooth curved column, as seen in the top view (Figure 5.23).

The dome renders, Figures 5.25-5.27, shows renders in top, perspective and the bottom view respectively. The smallest number of motifs that can be applied to form a dome is 12, which creates a three layer dome by the formed dome algorithm. However by enlarging parts of the complete structural renders, alongside gaps between motifs (circled in yellow), the algorithm also forms overlaps (circled in red). The gaps and overlaps can be seen in each of the different view renders of the dome. Also as the motif increases in size within each layer by the scale factor, the dome formed isn’t a complete smooth curved structure.

Finally Figures 5.28-5.30 shows the renders of the self-similarity star in the front, perspective and back views respectively. A larger number of motifs was applied to form the star structure to see if there was any difference in the number of motifs used. This structure is formed of 20 motifs (circumference) with 4 layers. Although the tessellation is more dense and concise, the gaps and overlaps are lessened however they are still formed within the structure as highlighted within the red and yellow circles.

In the four architectural structures it was seen that the curved surfaces, the column resulted in gaps and the dome and the self-similarity star, formed both gaps and overlaps. The algorithm to form these curved structures is created of rigid transformations, hence the artefacts. The other limitation of this grammar is that it only forms four specific features, hence it is not generalized.

5.8 Summary

To overcome the gaps and or overlaps with the shape grammar rules within the Cartesian and hexagonal grid, the grammar needs to be extended by integrating mapping and transformation methods. To do this a more generalized mapping grammar will be formed that will be adapted within the current APSG. In the next chapter, a Point Set Registration (PSR) technique, which will create an accurate transformation and a mapping method that will map the 3D IGM in the correct position will be integrated in the APSG. A theoretical background of the different PSR methods will be described with a detailed description of the chosen method, Coherent Point Drift (CPD). The integration of the two techniques of mapping and transformation will be discussed in the methodology that will form the Volumetric Shell Shape Grammar (VSSM).

6. Mapping on Point Clouds

“Pure mathematics is, in its way, the poetry of logical ideas.”[6]

6.1 Introduction

In the previous chapter shape grammar was applied to form architectural structures. However, it was seen that the method only allowed rigid transformations which were defined by the shape grammar rules. This created gaps or overlaps between the motifs when they were tessellated on curved surfaces like the dome. From researching the different methods on mapping volumetric textures in chapter 3, shell mapping was seen to be the most efficient in applying geometry within a shell space. Finding the optimal transformation in a mapping technique is however the important aspect. As the mapping of motifs is not applied onto any given surface, but formed within the mapping itself, the underlying grid to work best on was point clouds formed of parametric equations. One technique that maps a point set to another is Point Set registration (PSR).

PSR is usually applied to large point cloud data, and data that inherits outliers. Having such data does make the mapping complex. However in the data required to map the motifs, the point clouds that have been formed do not include outliers.

[6] *Albert Einstein.*

In the following chapter, the PSR is described with different algorithms that have been implemented within it. The generalised algorithm for mapping 3D motifs will be discussed. A flowchart is initially provided, with detailed descriptions of each step following on from it. Firstly, the formation of the point cloud of data is described, including the process of how the data has been obtained and why it is accurate to use. Following on, the formation of the volumetric texture is discussed. Later, the method of Coherent Point Drift (CPD) is described in detail. The chapter is concluded by a set of results produced from the mapping algorithm.

6.2 Point Set Registration

PSR is divided into two main categories, rigid and non-rigid, depending on the transformation. Rigid transformations, as previously stated, preserve all the distances of the given object. Rigid transformations include translation, rotation, scale or a combination of all three. Deformations such as bending or non-linear scaling is what causes non-rigidity within an object, it mainly consists of transformations applied on curves. The simplest non-rigid transformation is affine.

Many algorithms exists for both rigid and non-rigid PSR. Methods like Iterative Closest Point (ICP) are applied to calculate rigid transformations [93]. ICP is a technique that iteratively defines the correspondences between points that are close to each other, providing a least-square rigid transformation. The method continues until it reaches its optimal transformation. However, the limitation within ICP is that the point sets should be close to each other.

With this limitation within ICP, methods developed by [94], [95] propose to overcome it by probabilistic techniques. One popular method is the Robust Point Matching (RPM) algorithm that was initialised by [96]. RPM applies soft assignment of correspondences between the two point sets allowing global to local search. However, this method is improved on by [97] relating the RPM method to the Expectation Maximisation (EM) algorithm for mixture models. This is formulated through calculating the centroids of mixture models. A probabilistic method similar to this is known as the Coherent Point Drift (CPD).

6.2.1 Coherent Point Drift

Coherent Point Drift (CPD) allows both rigid and non-rigid point registrations. It is formed through the Motion Coherence Theorem which allows the points to move coherently to the target. CPD is a probabilistic technique that is taken as an assumption of a Maximum Likelihood (ML) approximation problem. With two given point sets, one set is defined by the Gaussian Mixture Model (GMM) centroids that are placed onto the second set by maximising the likelihood. This is iteratively calculated until the posterior GMM probability reaches its optimum level. The centroids flow in a coherent fashion as a whole set of points to preserve the topological structure of the set of points. CPD has the advantage of approximating non-rigid transformations within highly complex point sets that include both outliers and randomly placed points.

With two given point sets:

$X_{N\times D}$ (The reference point set or the target set) in a matrix form where N is the number of points in the set X and D represents the dimension of the point.

$Y_{M\times D}$ (The template point set or source set), where M is the number of points in the set Y these points are considered as centroids of GMM.

The GMM Probability Density Function (PDF) is defined as,

$$p(x) = \sum_{m=1}^{M+1} P(m)p(x|m), \tag{6.1}$$

where $P(m) = 1/M$ for $m \epsilon \{1, \dots, M\}$ and,

$$p(x|m) = \frac{1}{(2\pi\sigma^2)^{D/2}} e^{(\|x-y_n\|)/2\sigma^2}. \tag{6.2}$$

The maximum value of the mixture model is calculated through the GMM centroid location, θ, which is the equivalent to minimizing the log function E,

$$E(\theta, \sigma^2) = -\sum_{n=1}^{N} log \sum_{m=1}^{M+1} P(m)p(x|m). \tag{6.3}$$

The Expectation Maximization (EM) is then applied to calculate the optimal transformation, τ, through iterations. Every iteration improves the objective function Q that was previously calculated,

$$Q(\theta,\sigma^2) = \frac{1}{2\sigma^2}\sum_{n=1}^{N}\sum_{m=1}^{M+1} p^{prev}\,(m|x_n) \parallel x_n - \tau(y_m,\theta) \parallel^2 + \frac{NpD}{2}\log(\sigma^2). \qquad (6.4)$$

For non-rigid registration the transformation is represented by the velocity function v where $\tau(Y,v) = Y + v(Y)$. Equation (6.10) is minimized by function v:

$$v(z) = \sum_{m=1}^{M} w_m G(z - y_m), \qquad (6.5)$$

where G is the Gaussian kernel and $w_m = \sum_{n,m=1}^{N} p^{prev}(m|x_n)(x_n - \big(y_m + v(y_m)\big))$. The value of σ^2 is recalculated before the completion of the EM step by adding the derivative of Q with the new transformation τ of y_m to zero,

$$\sigma^2 = \frac{1}{N_P D}\sum_{m,n=1}^{M,N} \parallel x_n - \tau(y,W) \parallel^2, \qquad (6.6)$$

where W is the $M \times D$ matrix of the coefficients. Figure 6.1 shows how one point set (blue point set) is registered onto the target (red) point set.

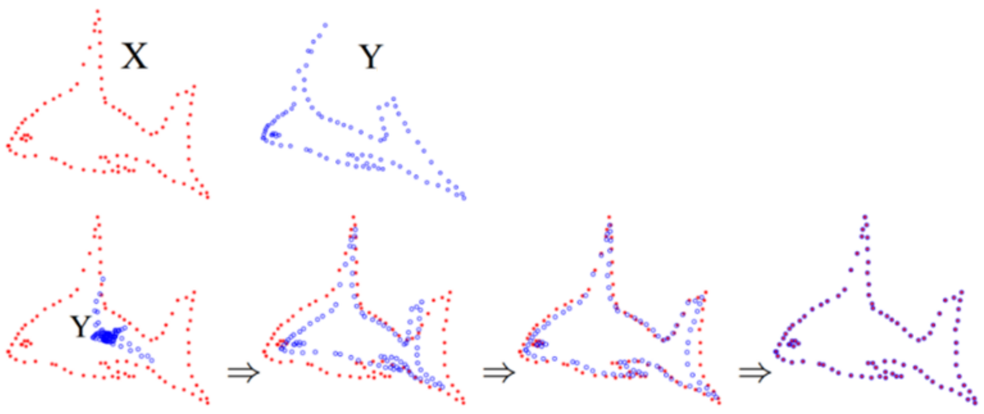


Figure 6.1 CPD, the blue point set are being registered to red point set.

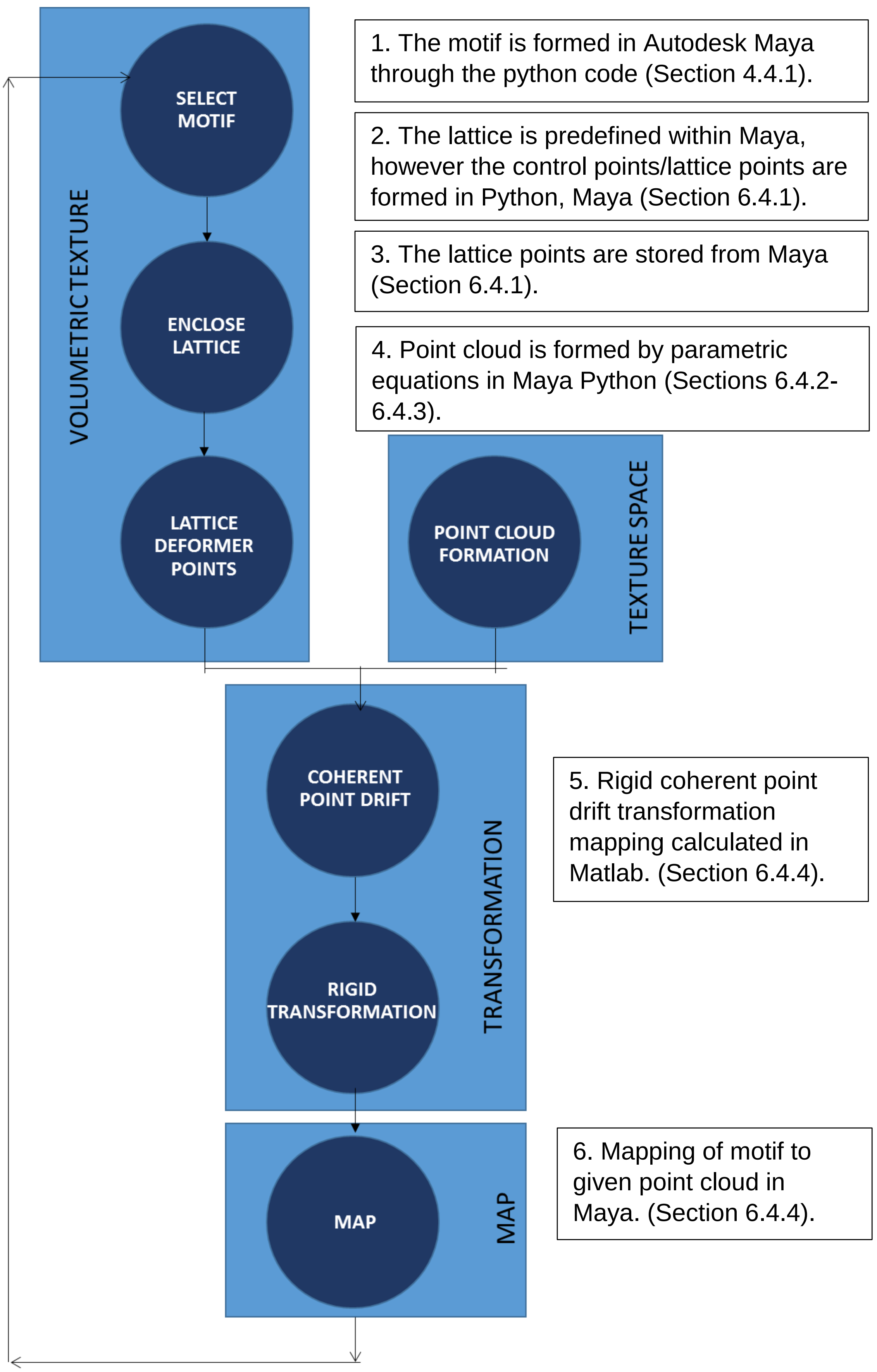
The next section explains how CPD is implemented into the PSG, to form a shape grammar mapping technique.

6.3 Proposed Framework: Volumetric Shell Shape Grammar (VSSG)

The Volumetric Shall Mapping (VSM) algorithm is formed of four crucial steps.

1. Texture space
2. Volumetric Texture
3. Transformation
4. Map

In the next section each of the four steps will be described in detail. Figure 6.1 shows the VSSG algorithm in a flowchart.



1. The motif is formed in Autodesk Maya through the python code (Section 4.4.1).

2. The lattice is predefined within Maya, however the control points/lattice points are formed in Python, Maya (Section 6.4.1).

3. The lattice points are stored from Maya (Section 6.4.1).

4. Point cloud is formed by parametric equations in Maya Python (Sections 6.4.2-6.4.3).

5. Rigid coherent point drift transformation mapping calculated in Matlab. (Section 6.4.4).

6. Mapping of motif to given point cloud in Maya. (Section 6.4.4).

Figure 6.2 Flowchart for mapping onto point clouds.

6.4 Methodology

6.4.1 Volumetric Texture

For the motif to map, the ideal way to store the motif is as a volumetric texture. This is done through the use of a lattice deformer. A deformer manipulates a set of vertices and forms new vertex positions for the same exact vertex. A deformer is usually in the form of a vector-valued function $f(x, y, z)$ where the function f is formed with the combination of three scalar component functions, f_x, f_y, f_z,

$$f(x,y,z) = \left(f_x(x,y,z), f_y(x,y,z), f_z(x,y,z)\right). \tag{6.7}$$

Translation deformers are the simplest forms of deformers as it creates a new translated vertex coordinate from the original vertex,

$$(f_x, f_y, f_z) = (x, y+2, z). \tag{6.8}$$

Many CG software applications, including Autodesk Maya, contain different types of deformers that are already embedded in the software. With the variety of deformers in Maya, the best deformer to use for the purpose of mapping motifs was the lattice deformer. As many of the deformers are controlled by user-defined parameters, the lattice deformer has lattice control points that can be controlled. The structure of the lattice is formed of points in a cage like structure. As the motifs are of a mesostructure scale, the lattice deformer is efficient to return a distinguishable Level of detail (LOD) of the motif after it has been mapped or in this case deformed to the correct positions.

Following on from the previous chapter, both the square and hexagonal tiling systems will be applied, hence lattices of both shapes are formed. The square and hexagonal lattices are both written in Maya python as the lattice deformer is included in the Maya software. Figure 6.3 shows how both the lattices enclose the same motif; the red points represent the lattice control points.

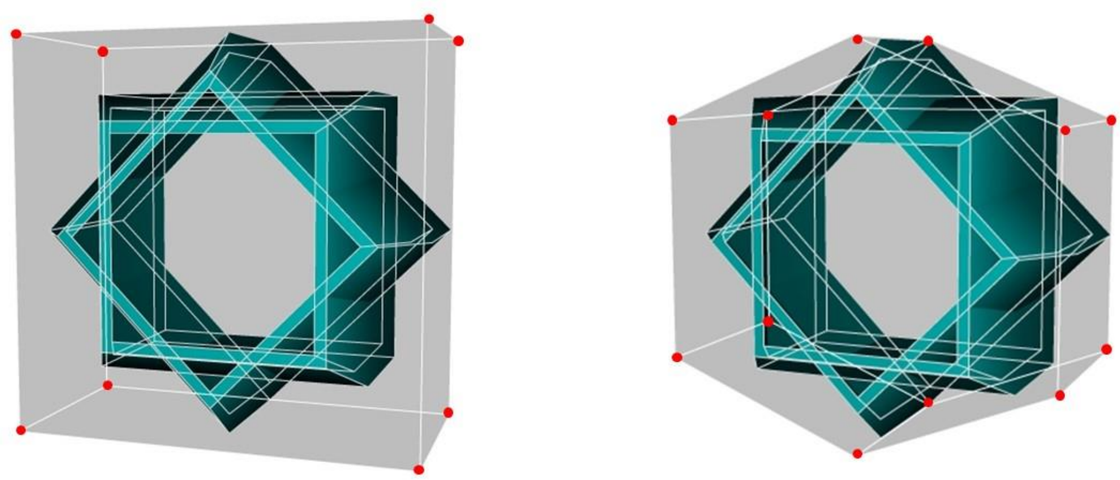


Figure 6.3 Lattice for (Left) Square grid mapping (Right) Hexagonal grid mapping.

6.4.2 Texture Space formation

The point cloud is formed of architectural structures that were defined in the previous chapter. The structures were previously formed by rigid transformations going through a certain algorithm step by step. In this chapter the structures will be created by analytical representations of the structural surfaces. The point clouds will be derived by implementing parametric equations of the surface.

Mathematical representation of a surface

There are three ways a surface can represented mathematically, either implicitly, explicitly or through parametric equations. Implicit representation is in the form of $f(x, y, z) = C$, where x, y, z are Cartesian coordinates and C is any constant in 3D space, forming a function f. For a function to represent a plane, the function must be linear in the form of $ax + by + cz + d = 0$. If the function is of order 2, which is a quadratic function, for example, $x^2 + y^2 + z^2 = r^2$, this represents a surface, the surface of a sphere in this case. One can easily test if a point lies inside, outside or on the surface by the following conditions:

- $f(x, y, z) < 0$ (inside the surface),
- $f(x, y, z) = 0$ (on the surface),
- $f(x, y, z) > 0$ (outside the surface).

This gives the advantage of determining if a point is on the surface or not.

Explicit functions are coordinate system dependent. An explicit surface representation is in the form of $z = f(x, y)$, where x and y are both independent variables and z a dependent variable. They are usually used to determine height fields as the function determines the 3rd dimension, z. They cannot represent a full sphere, $z = \sqrt{(r^2 - x^2 - y^2)}$. Due to these implications explicit surface representations are rarely used in CG.

Parametric equations on the other hand cannot determine if a point is on a surface easily. However it does have the advantage of defining a set of control points by its parameters. Parametric equations are the most common type of surface representation in CG. They can create complex geometries that are defined by a set of control points. They also have the advantage of easily manipulating a surface. Parametric representation maps from a domain $\mathbb{R}^2$ to range $\mathbb{R}^3$, mapping two parameters u, v to three functions in the general 3D form of,

$$f(u,v) = \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix}. \tag{6.9}$$

A polygonal surface can be formed by parametric equations as it approximates a surface efficiently and systematically generates the points. As the inverse of the function, $(u,v) = f^{-1}(x)$, can be computed easily, parameters u and v can define an index of a texture map. Due to these advantages the structural surfaces will be represented parametrically in this research as the parametric surface points will act as point clouds.

A given point cloud as described earlier are points that form a surface, they do not however create a volumetric shell of cloud points, which is the texture space for mapping the motifs in. One way to form a volumetric point cloud shell, for example imagine a hollow cylinder, would be to compute an interior and exterior surface of data points using parametric equations of the cylinder. However, this will return a mathematically complex computation. A simpler yet effective method would be to duplicate and combine two parametric surfaces. To do this one can create a surface of the cylinder and extrude it in or out, from the origin (0,0), to form the volumetric shell. Another method would be to create two solid cylinders, both with the same centres but with varying radii, forming concentric cylinders and then applying the Boolean operation of intersection, forming a hollow cylinder. Using the initial process of the latter method, a volumetric point cloud shell can be generated, by forming and combining two data sets with the same parametric representation but with varying radii.

For example taking the parametric equations of a cylinder, one would compute the equations with radius, r_1 (see Figure 6.4):

$$\begin{gathered} x = r_1 cos(x) \\ y = r_1 sin(x) \\ z = h \end{gathered}$$

And a second set of parametric equations with a radius, r_2 (see Figure 6.4):

$$\begin{gathered} x = r_2 cos(x) \\ y = r_2 sin(x) \\ z = h \end{gathered}$$

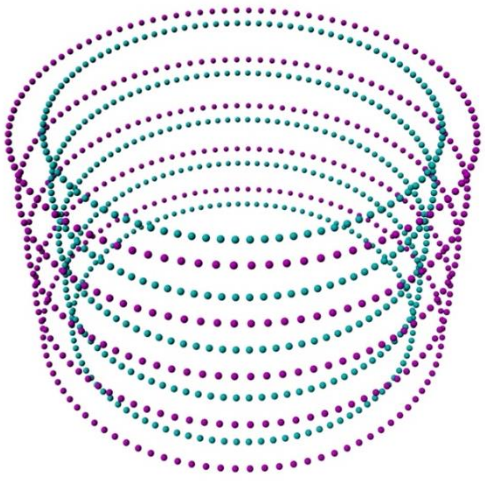


Figure 6.4 Parametric representation of the two cylinders with radius of 3cm (cyan points) and radius of 3.5cm (purple points).

The difference between the two radii of the parametric representation of the cylinders is 5mm in this example, this is just enough texture space between the two point cloud surfaces to map the motifs in the individual volumetric blocks. However the difference is dimensionless and can be set by the user forming the parametric surface in this initial stage of the mapping.

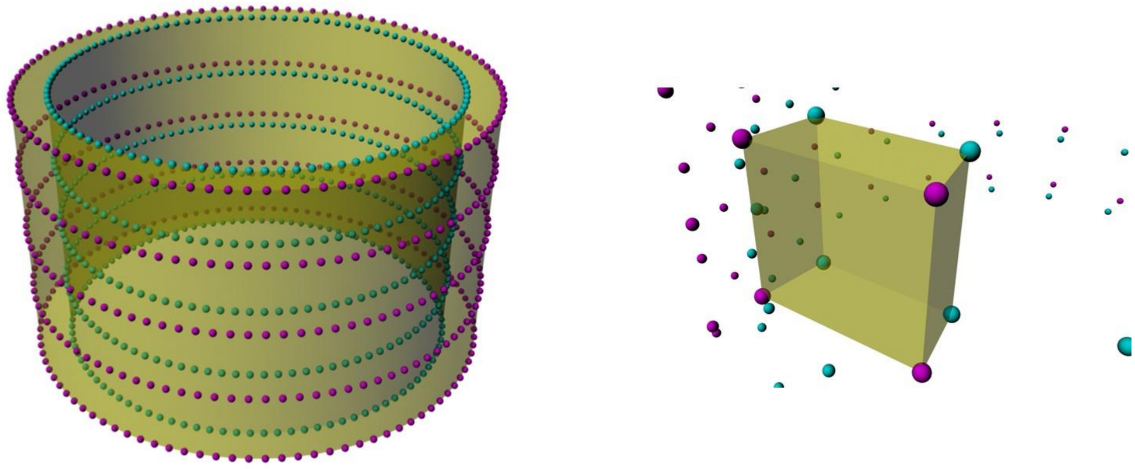


Figure 6.5 (Left) Parametric surface representation with volumetric shell (Right) Volumetric block of the texture space for mapping a motif.

As there is a point cloud with data points representing a surface with a shell, the next step is to calculate the texture space, where each of the individual motifs will be mapped to.

6.4.3 Grid formations

As parametric equations by default map the points in a consecutive manner in the Cartesian grid, they automatically form the square shape boundary within the grid. Hence it is simple to form the quad texture space.

For the quad volumes, the volumes can be created by grouping the data points into groups of 8 points with x, y and z coordinates of the parametric equations. This would be 4 points from the 1st set of parametric equations of the surface and the rest of the points from the 2nd set. The procedure to form a square grid is,

$$\begin{aligned} a &= i + j(n), \\ b &= i + j(n) + 1, \\ c &= i + (j+1)n + 1, \\ d &= i + (j+1)n, \end{aligned} \tag{6.10}$$

where $i, j = 0,1,2 \ldots r$, n is the number of points and $r = n - 1$. Figure 6.6 shows an example of how the quad volumetric space is formed with equation 6.4, by grouping the points together.

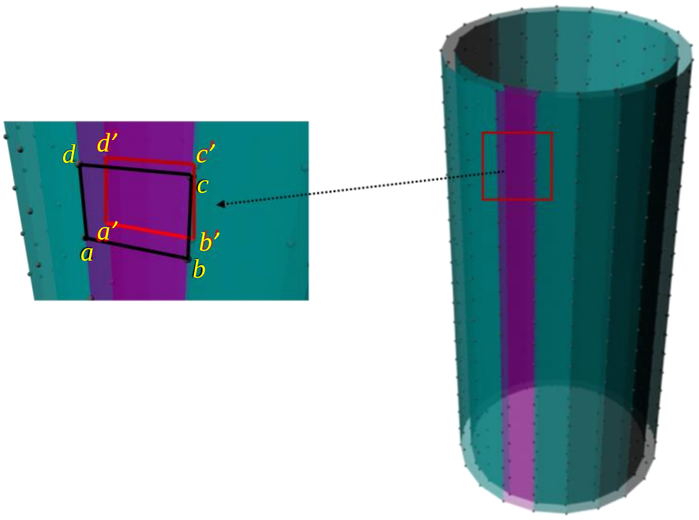


Figure 6.6 (Right) Quad volumetric space (left) enlarged.

The UV map of the cylinder is shown below, Figure 6.7. For any open surface, like a cylinder, where the surface is joined from only one side, an extra set of equations, equation 6.11, are calculated (indicated in purple), to join the surface.

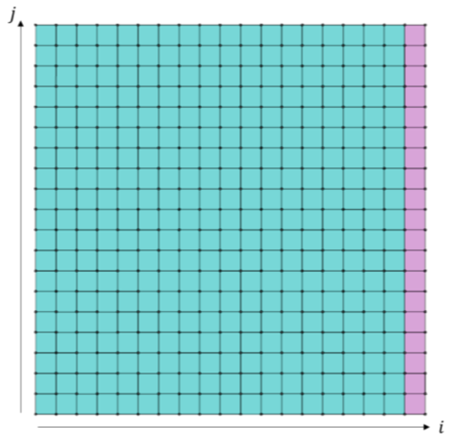


Figure 6.7 UV map of an open surface, cylinder.

$$\begin{aligned} e &= j, \\ f &= nr + j, \\ g &= nr + j + 1, \\ h &= j + 1. \end{aligned} \tag{6.11}$$

As this research is on mapping and forming tessellations without gaps and overlaps, the Cartesian co-ordinate system will not be altered, hence the hexagonal grid will be created within the square grid. Altering a grid system is a case of remeshing a surface, which is a problem that can be dealt with in the future. So to form the hexagonal grid, the following set of equations are applied, equation 6.6. Each hexagon is contained within a 2x3 (pointy top) or 3x2 (flat top) square grid.

For the pointy top hexagonal grid,

$$
\begin{aligned}
a &= \left(\frac{1}{2} + \frac{(-1)^{j+1}}{2}\right)n + 2j + (2i+1)n \\
b &= \left(\frac{1}{2} + \frac{(-1)^{j+1}}{2}\right)n + (2j+1) + (2i+2)n \\
c &= \left(\frac{1}{2} + \frac{(-1)^{j+1}}{2}\right)n + (2j+2) + (2i+2)n \\
d &= \left(\frac{1}{2} + \frac{(-1)^{j+1}}{2}\right)n + (2j+3) + (2i+1)n \\
e &= \left(\frac{1}{2} + \frac{(-1)^{j+1}}{2}\right)n + (2j+2) + (2i)n \\
f &= \left(\frac{1}{2} + \frac{(-1)^{j+1}}{2}\right)n + (2j+1) + (2i)n
\end{aligned}
\tag{6.12}
$$

where $i, j = 0,1,2 \ldots s$ and $s = (n/2) - 1$

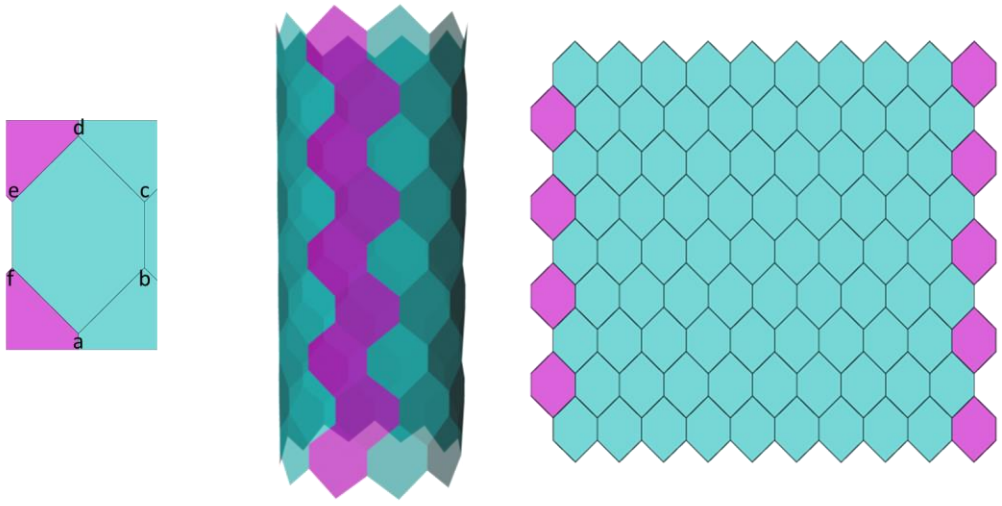


Figure 6.8 An example of the pointy top hexagonal grid.

For the flat top hexagonal grid,

$$a = \left(\frac{1}{2} + \frac{(-1)^{j+1}}{2}\right) + (2j)n + (2i+1)$$

$$b = \left(\frac{1}{2} + \frac{(-1)^{j+1}}{2}\right) + (2j+1)n + (2i+2)$$

$$c = \left(\frac{1}{2} + \frac{(-1)^{j+1}}{2}\right) + (2j+2)n + (2i+2) \tag{6.13}$$

$$d = \left(\frac{1}{2} + \frac{(-1)^{j+1}}{2}\right) + (2j+3)n + (2i+1)$$

$$e = \left(\frac{1}{2} + \frac{(-1)^{j+1}}{2}\right) + (2j+2)n + (2i)$$

$$f = \left(\frac{1}{2} + \frac{(-1)^{j+1}}{2}\right) + (2j+1)n + (2i)$$

Hence, the point cloud data for either of the hexagonal grids is grouped with 12 points.

6.4.4 Mapping transformation

With the lattice points defined and the volumetric index shell formed, it is now time to apply the PSR method. The transformation is the key component of mapping a 3D volume onto a volumetric point cloud. The data points in both source and target point clouds should correspond correctly ideally, forming a one-to-one correspondence, for the motif to map without any deformation or distortion. This can be checked by the formation of the data set and how the points are aligned to each other. For example the 2 data sets data sets below,

$$
\begin{aligned}
P1 &= (0,0,0) \longrightarrow P1' = (2,2,0) \\
P2 &= (1,0,0) \longrightarrow P2' = (3,2,0) \\
P3 &= (1,1,0) \longrightarrow P3' = (3,3,0) \\
P4 &= (0,1,0) \longrightarrow P4' = (2,3,0) \\
P5 &= (0,0,1) \longrightarrow P5' = (2,2,1) \\
P6 &= (1,0,1) \longrightarrow P6' = (3,2,1) \\
P7 &= (1,1,1) \longrightarrow P7' = (3,3,1) \\
P8 &= (0,1,1) \longrightarrow P8' = (2,3,1)
\end{aligned}
$$

This set corresponds correctly, with a smooth transformation of translation (2, 2, 0) in the x- and y-axis. However the ideal situation will not always be as smooth. If the points are not ordered correctly, the motif will be distorted after the transformation will occur.

Figure 6.9 gives an example of two data sets, the data points from the lattice deformer (left) that contains the motif and the data points of the volumetric texture space (right). The points are in the form of,

Source (lattice) = [0, 1, 2, 3, 4, 5, 6, 7]

Target (Volumetric Shell) = [1, 0, 2, 3, 4, 5, 7, 6]

As it can be seen, points 2, 3, 4 and 5 correspond correctly, but points 0, 1, 6 and 7 are not.

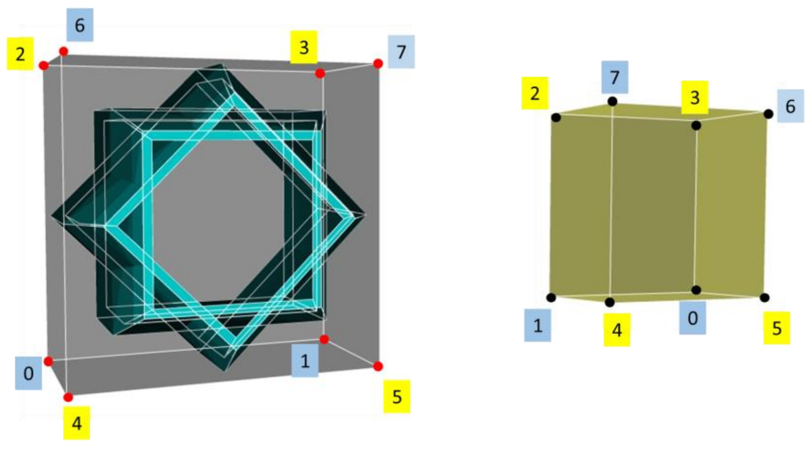


Figure 6.9 Correspondence of points in (left) texture space (right) shell space.

Hence to align the points, the CPD registration will be applied. CPD has both rigid and non-rigid point set registration algorithms. As the research is applying motifs through Euclidean transformations onto non-Euclidean surfaces, the rigid algorithm will be applied to register all the mapping transformations between the volumetric texture and shell space.

To apply the algorithm the Matlab code that is produced by [97] is implemented. The CPD Matlab code will only register the points, the mapping will be formed in Maya with the transformation matrix, which then calculates the result from the code. The transformation matrix will be calculated result after the input of the two datasets X and Y.

The next section describes the implementation of the VSSM algorithm, including both registration and mapping results.

6.5 Implementation

To Implement the VSSM algorithm the mapping occurs between the volumetric texture and the texture space. With the formation of the lattice deformer that encloses the 3D IGM and the parametric equations which creates the texture space, the corresponding data sets of the two spaces are exported from Maya and implemented into the Matlab code. After running the code the following registration results were formed to show the accuracy of the technique.

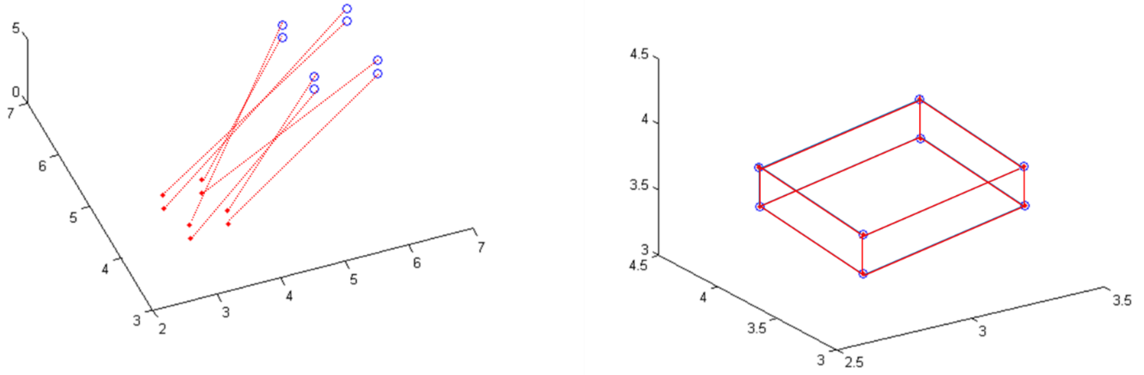


Figure 6.10 Registration of motifs (a) before (b) after in a wall.

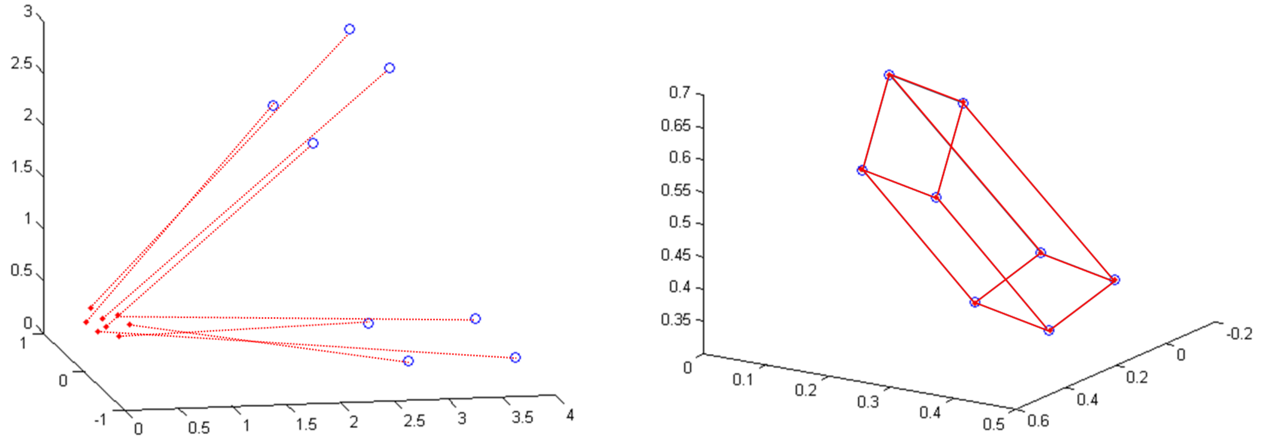


Figure 6.11 Registration of motifs (a) before (b) after in a column.

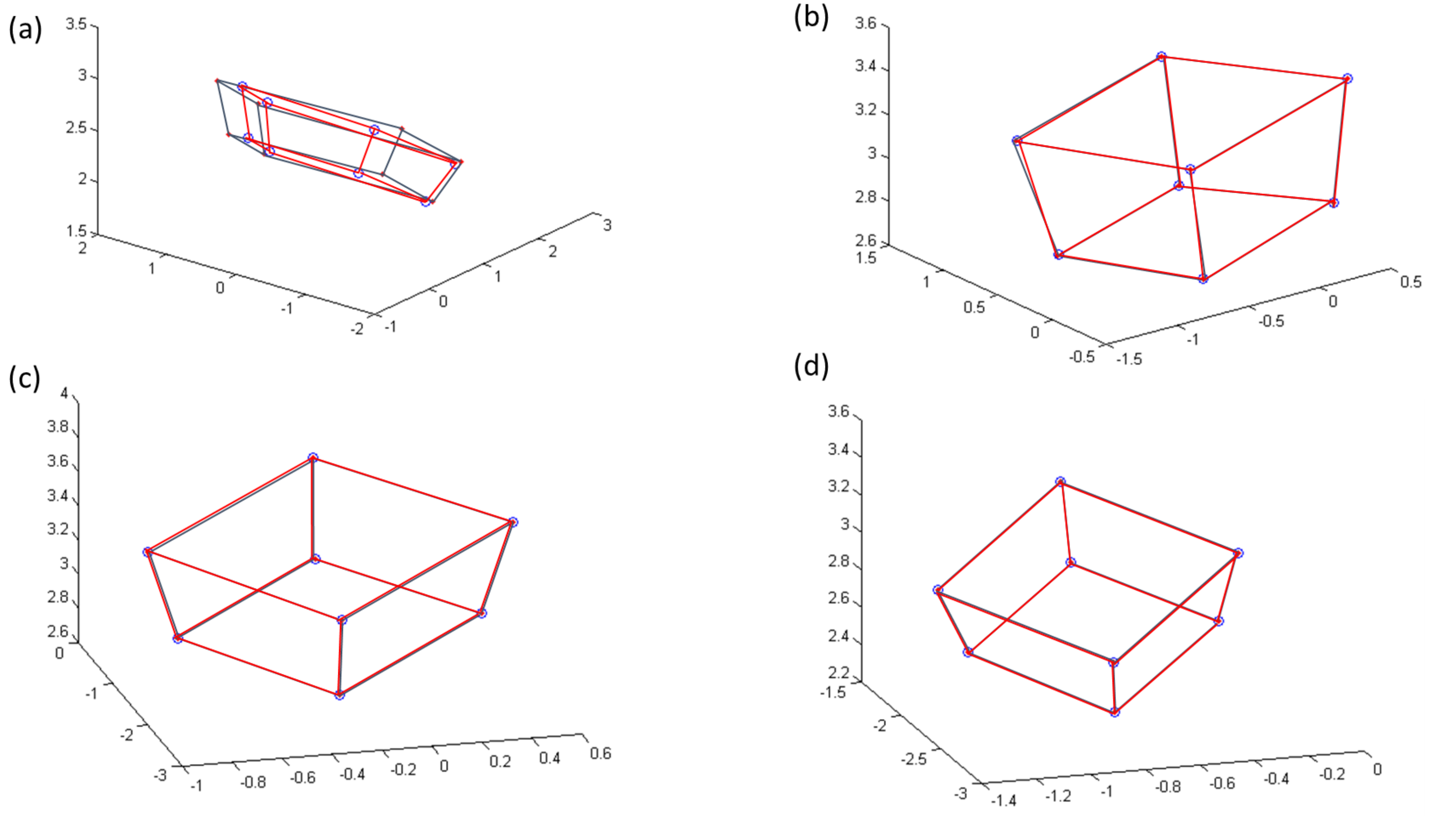


Figure 6.12 Registration of (a) 8 motifs (b) 16 motifs (c) 24 motifs (d) 32 motifs in a dome.

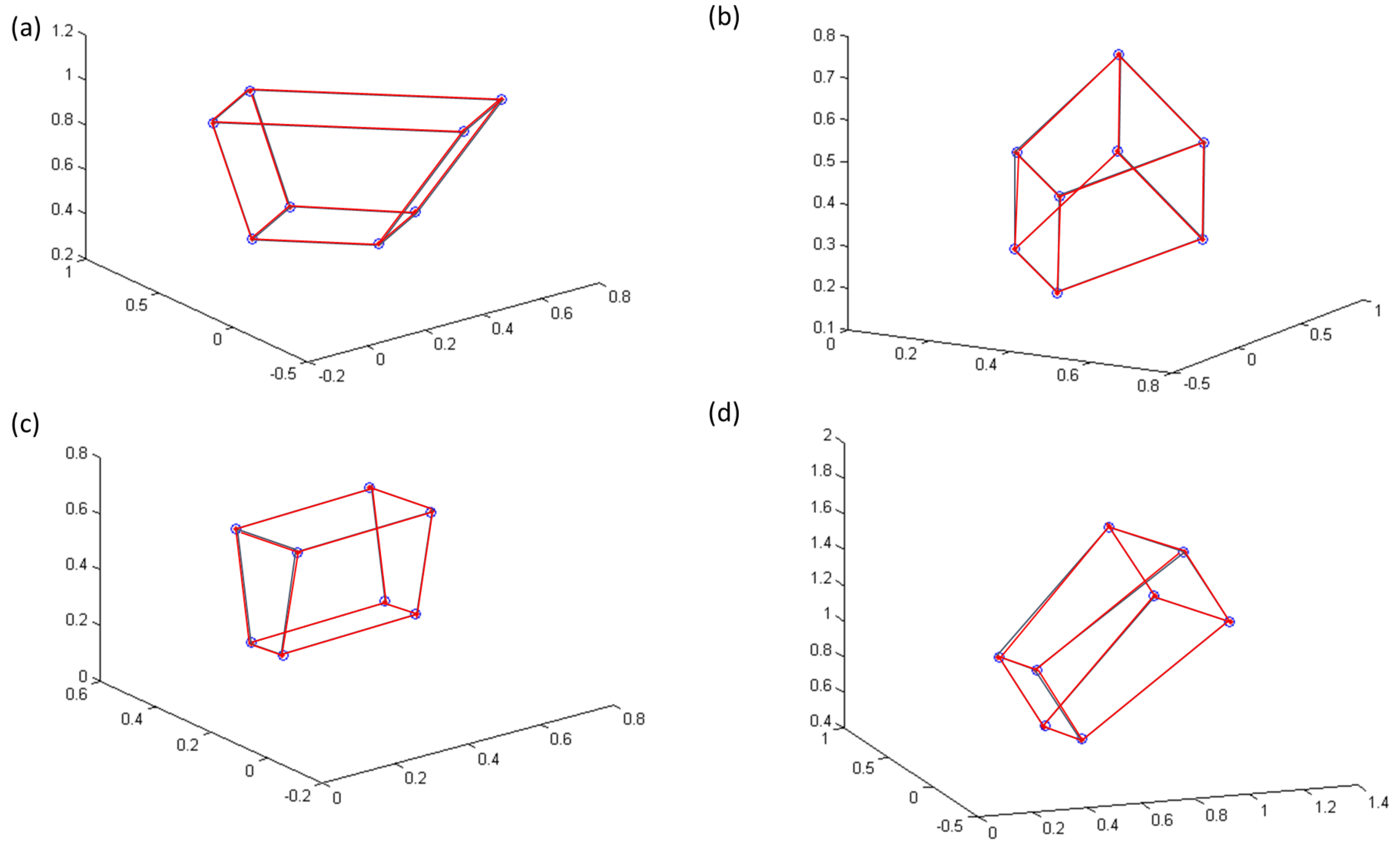


Figure 6.13 Registration of (a) 8 motifs (b) 16 motifs (c) 24 motifs (d) 32 motifs in a self-similarity star.

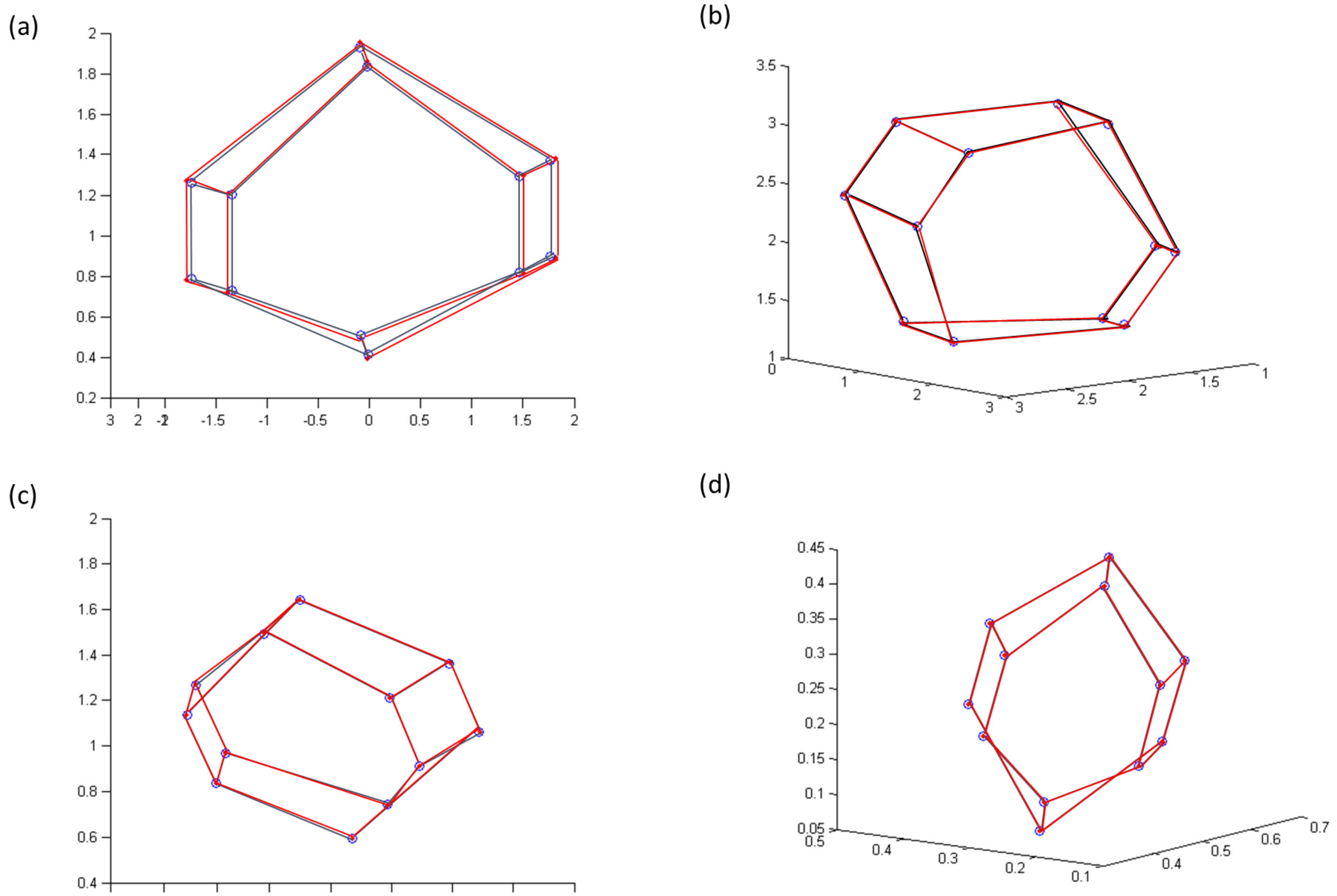


Figure 6.14 Registration of (a) 8 motifs (b) 16 motifs (c) 24 motifs (d) 32 motifs in a column (hexagonal grid).

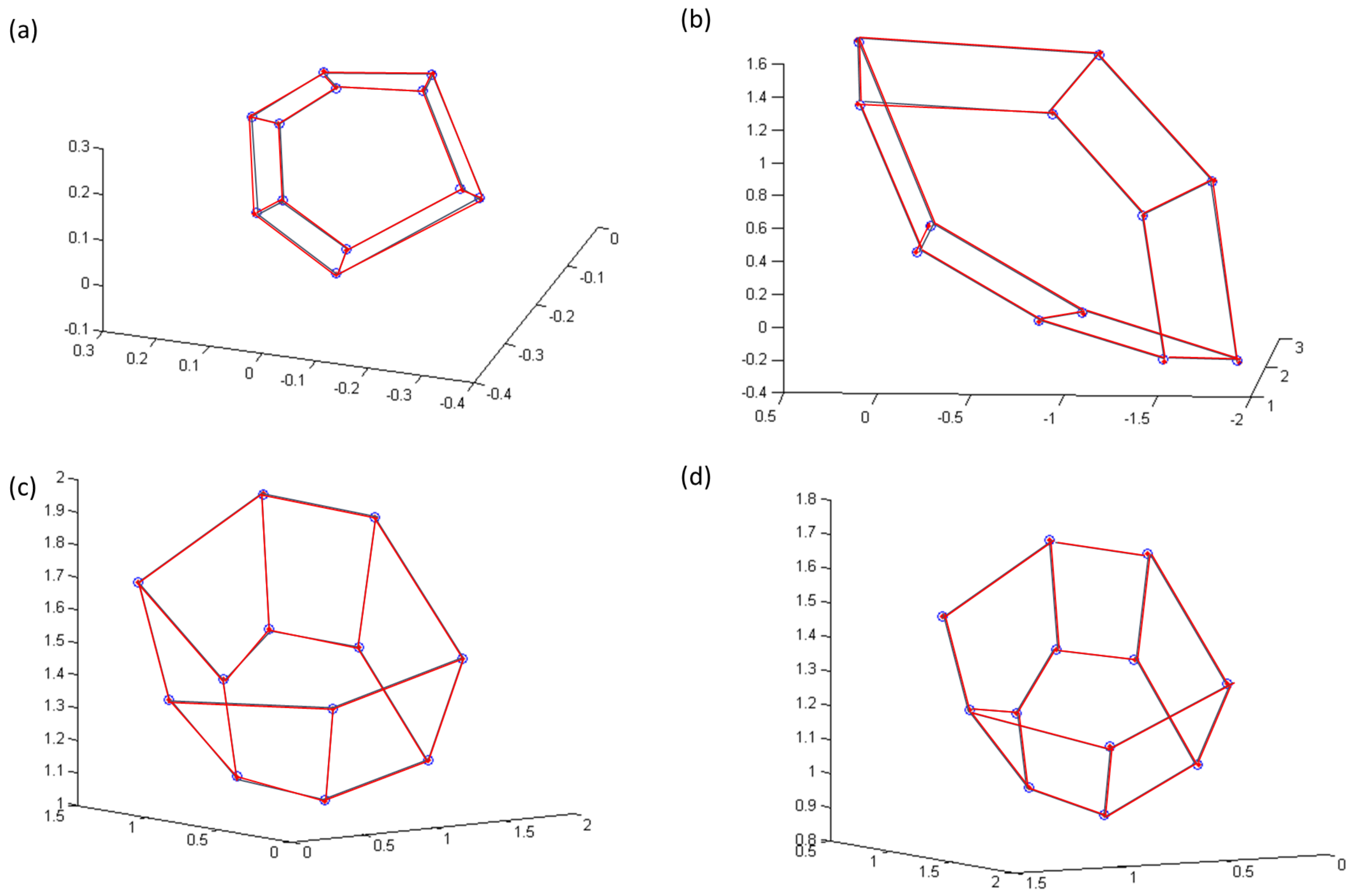


Figure 6.15 Registration of (a) 8 motifs (b) 16 motifs (c) 24 motifs (d) 32 motifs in a dome (hexagonal grid).

Figures 6.10 - 6.13 shows the registration results of a single 3D motif that is mapped to form a wall, a column, a dome and self-similarity star respectively, within a square grid with a set number of motifs. Figure 6.10 (a) - 6.11 (a) illustrates the two datasets, X (blue points, target set) and Y (red points, source set) before registration for a wall and a column. Figures 6.10 (b) – 6.11 (b) shows the correspondence after the registration. Figures 6.12 – 6.13 are registrations of the dome and the self-similarity star. Each figure contains four graphs, where each graph corresponds to a registered data set applying mapping onto a surface created by (a) 8 motifs (b) 16 motifs (c) 24 motifs and (d) 32 motifs. The same registration technique is applied to motifs that are mapped onto the hexagonal grid as well. Figures 6.14 – 6.15 show the registration of datasets of a motif mapped onto a column and a dome. Like previously each figure contains four graphs of whom correspond to the application of (a) 8 motifs (b) 16 motifs (c) 24 motifs and (d) 32 motifs.

Figures 6.16-6.19 shows the complete renders applying the VSSG method. Renders of perspective view of a column (Figures 6.16), dome (Figures 6.17) and the self-similarity star (Figures 6.19). To see the internal view of the dome, a bottom view render of the dome can be seen in Figures 6.18.

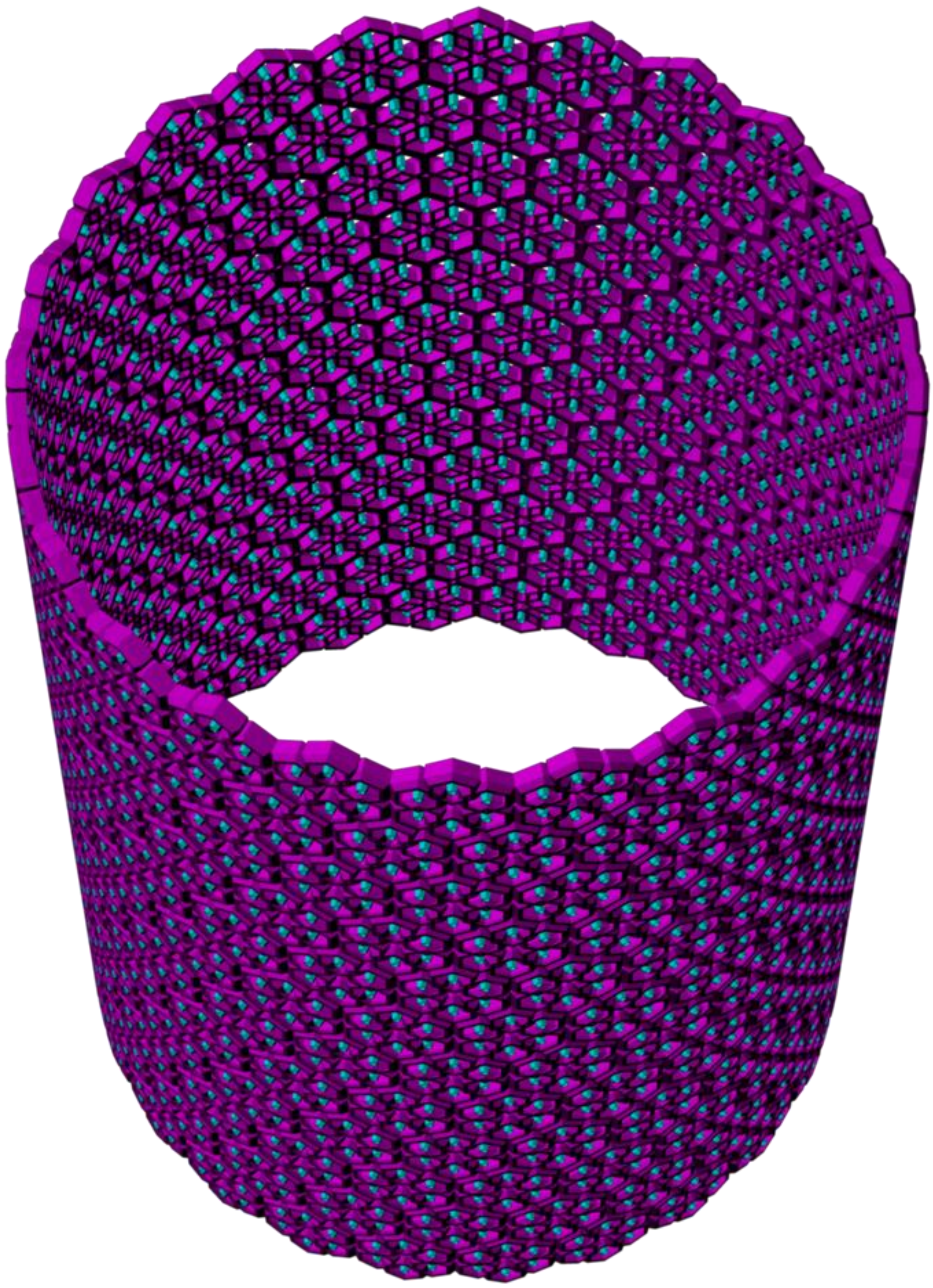


Figure 6.16 A perspective view VSSG render of a column.

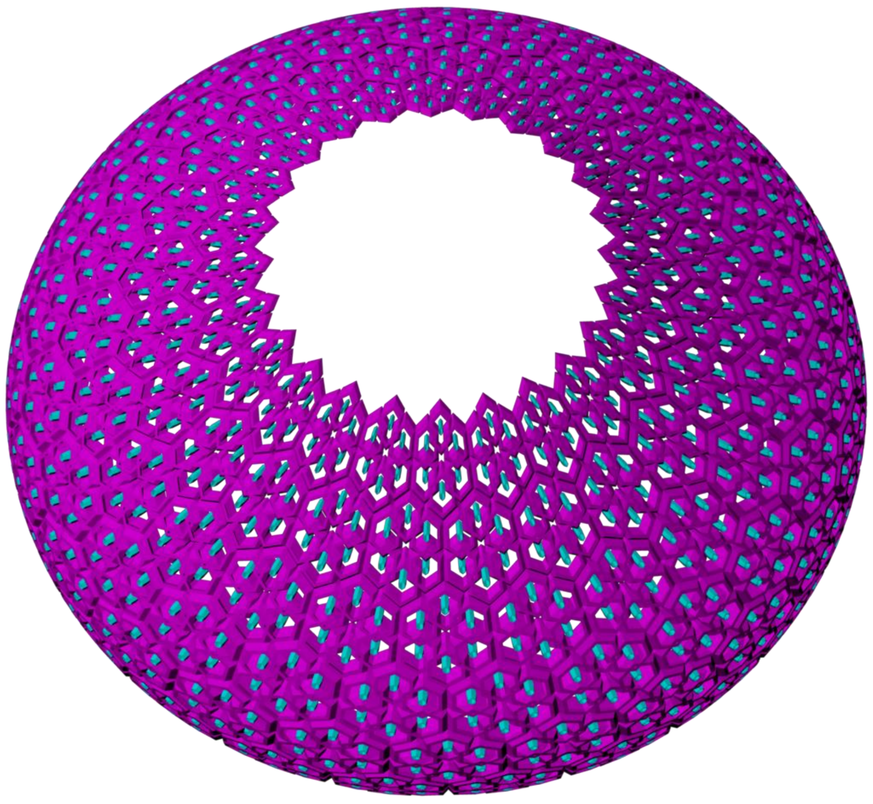


Figure 6.17 A perspective view VSSG render of a dome.

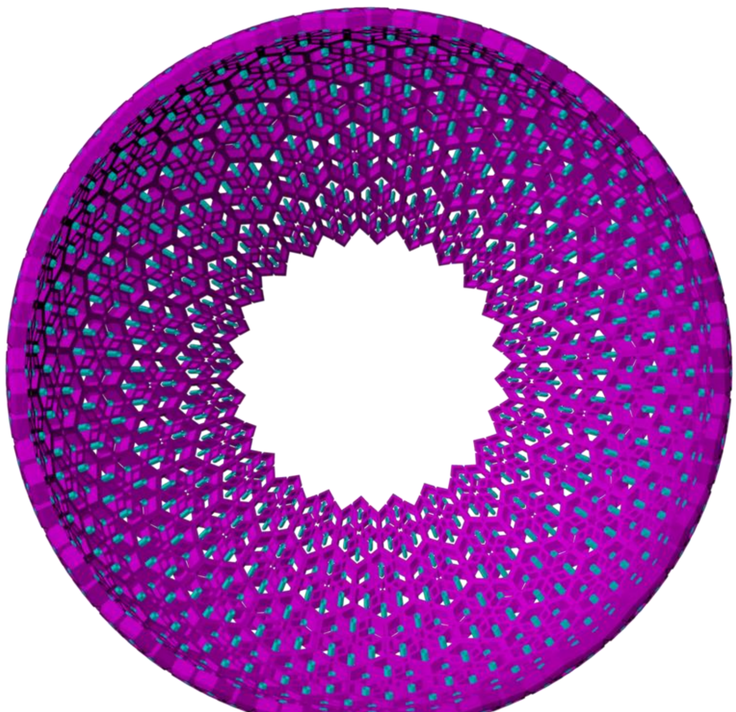


Figure 6.18 A bottom view VSSG render of a dome.

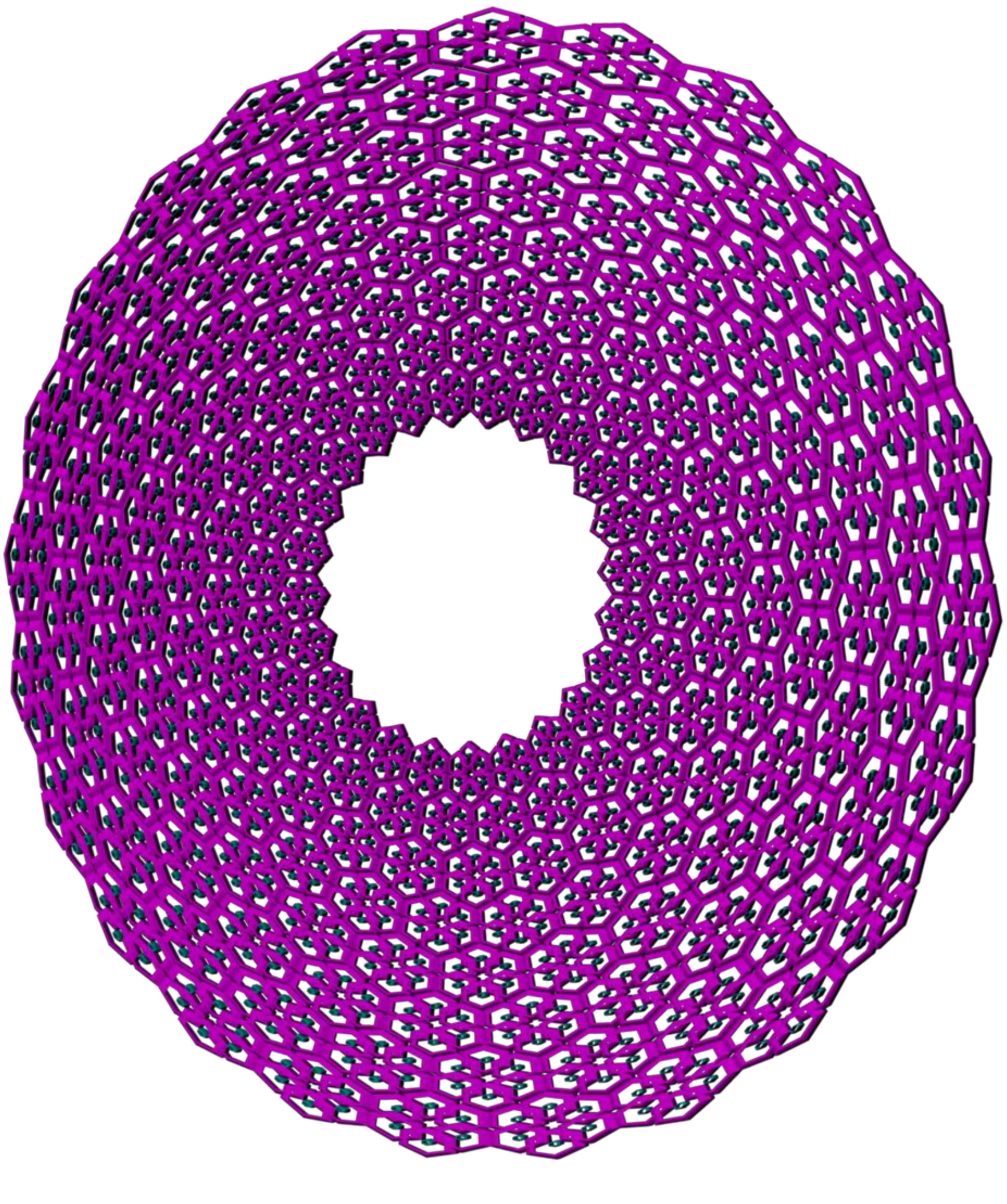


Figure 6.19 A perspective view VSSG render of a self-similarity star.

6.6 Analysis

From the registration graphs and renders of the architectural structures, shown in the previous section, it can be seen that the use of Euclidean transformations on both flat and curved surfaces, has produced better results. Firstly, looking at the motifs mapping on a square grid, the wall and the column resulted in perfect registration correspondence. As in the previous chapter, to form these structures the application of translation and or rotation need to be applied. Going onto the curved surfaces like the dome and the self-similarity star shown in Figures 6.12 – 6.13 the registration resulted with a difference. As the number of motifs applied to form the structure is a vital part in this research, the registration of the datasets between the two spaces was implemented by varying the use of the motifs, starting from 8 motifs and incrementing it with additional of 8 motifs.

From the dome registrations it was seen that the registration was better by increasing the number of motifs. The reason for this can be due to the large polygonal spaces that the motif has to map to. In the dome the less motifs used caused larger spaces, creating a very rigid registration. As the dome is formed of incremental circular surfaces, the more motifs applied creates a more curved surface.

However it was the opposite for the self-similarity star, but with a smaller difference. The motifs tessellate radially in the star, and as the motifs are packed compactly with a larger number of motifs hence the registration of datasets resulted in minor artefacts.

When the registration of datasets occurred between motifs mapping onto the hexagonal grid, the results varied from structure to structure. The correspondence of the wall resulted in a perfect registration, which was predicted as translation is the only transformation applied. The column however did not generate a perfect correspondence compared to the wall. Figure 6.15 shows that increasing the number of motifs produces better registration. As the shape of both the lattice deformer and the shell space is hexagonal, the correspondence is limited to only 12 data points. The lack of points could be a reason, as CPD produces optimal results on larger datasets.

The dome on the other hand resulted in minor difference between the registrations of the datasets, but again the correspondence is optimal with larger number of motifs used to form the structure.

The following figures, Figure 6.20Figure 6.27, shows the renders of the column, dome and the star. Each figures has a render of the structure in the two methods formed, APSG (in purple) and VSSG (in green), to see the comparison between the two. They also contain zoomed parts of each render to analyse the render. Figure 6.20Figure 6.22 shows the renders of the column in the front, perspective and top views respectively. Figure 6.23Figure 6.25 shows the renders of the dome in the top, perspective and bottom views respectively and finally Figure 6.26-Figure 6.27 are the star renders in the front and perspective views respectively.

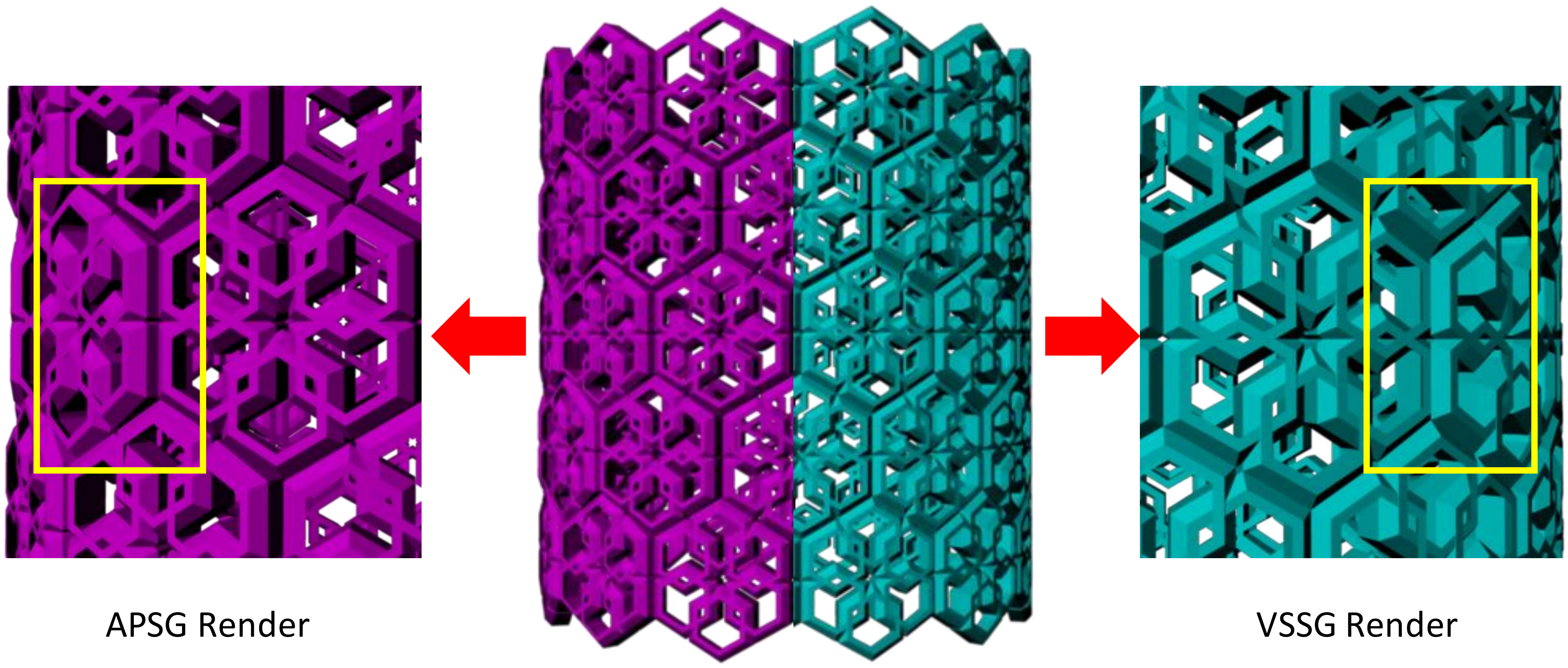


Figure 6.20 Column renders (front view) of (Left) APSG (Right) VSSG.

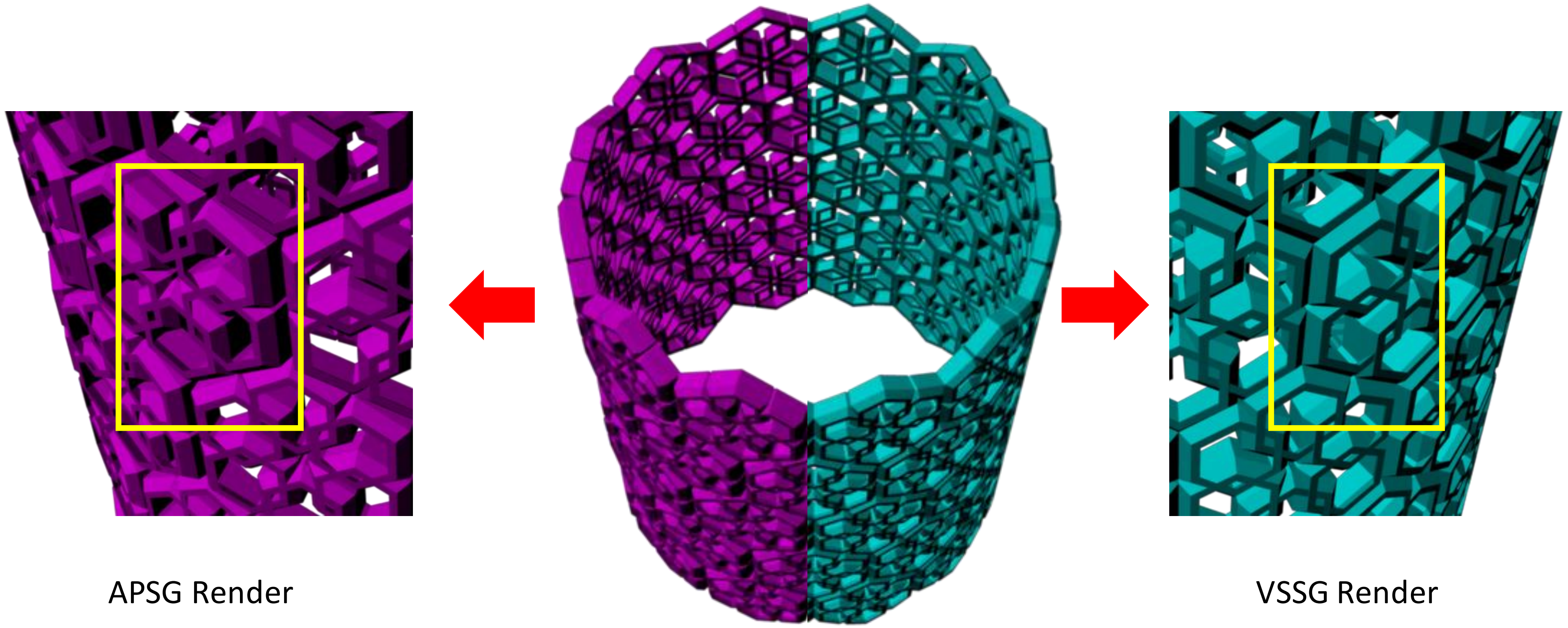


Figure 6.21 Column renders (perspective view) of (Left) APSG (Right) VSSG.

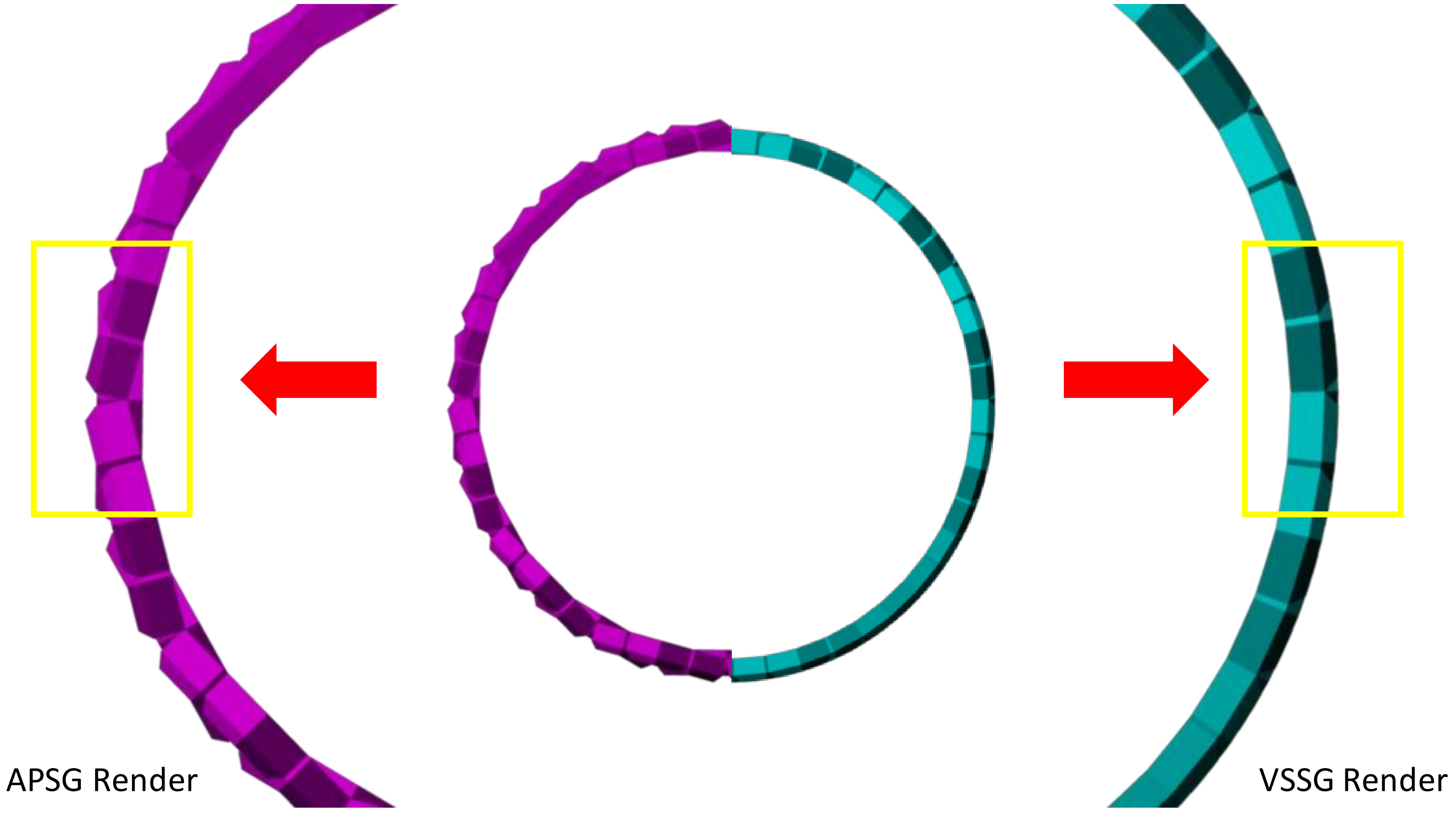


Figure 6.22 Column renders (top view) of (Left) APSG (Right) VSSG.

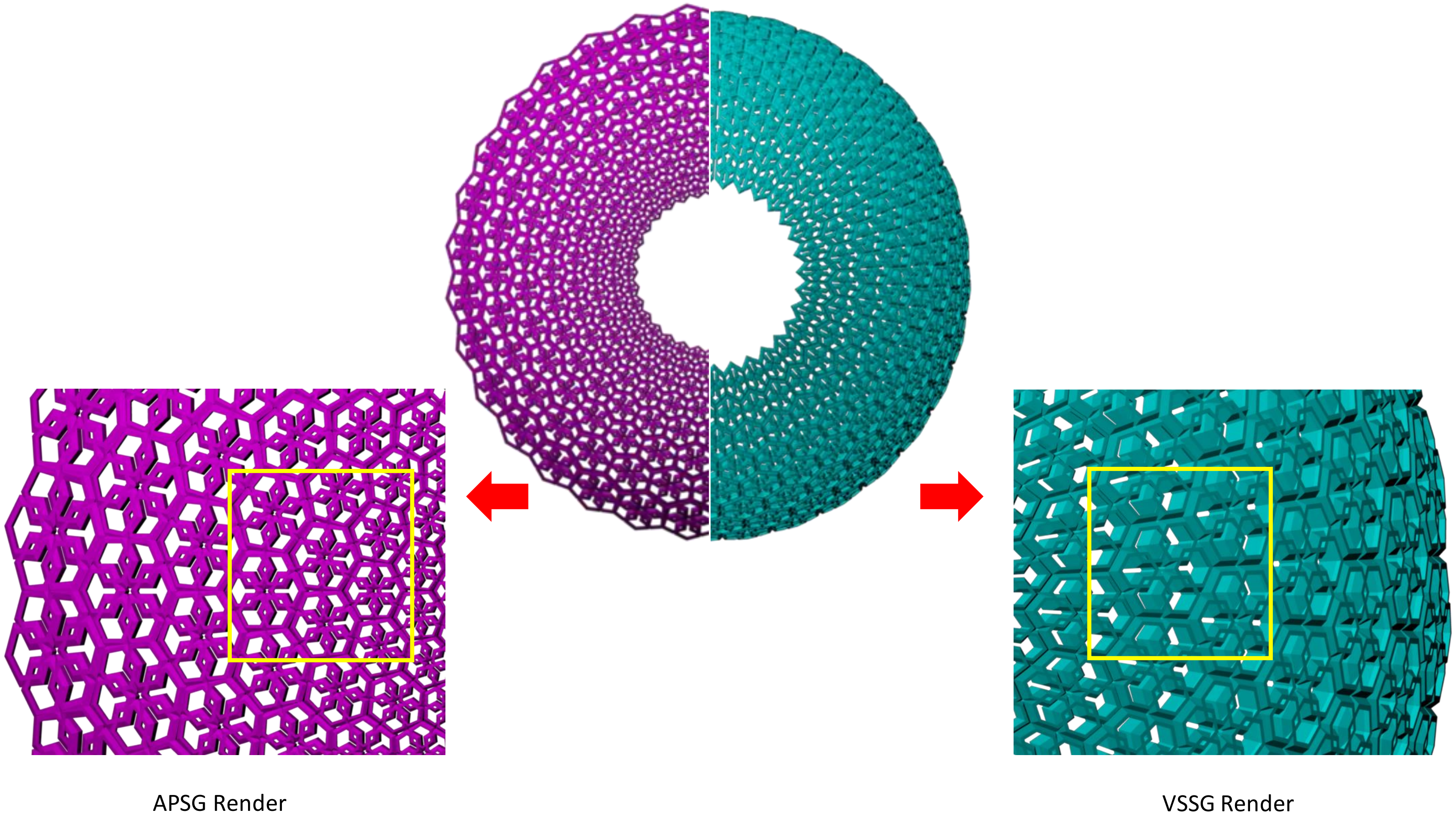


Figure 6.23 Dome renders (top view) of (Left) APSG (Right) VSSG.

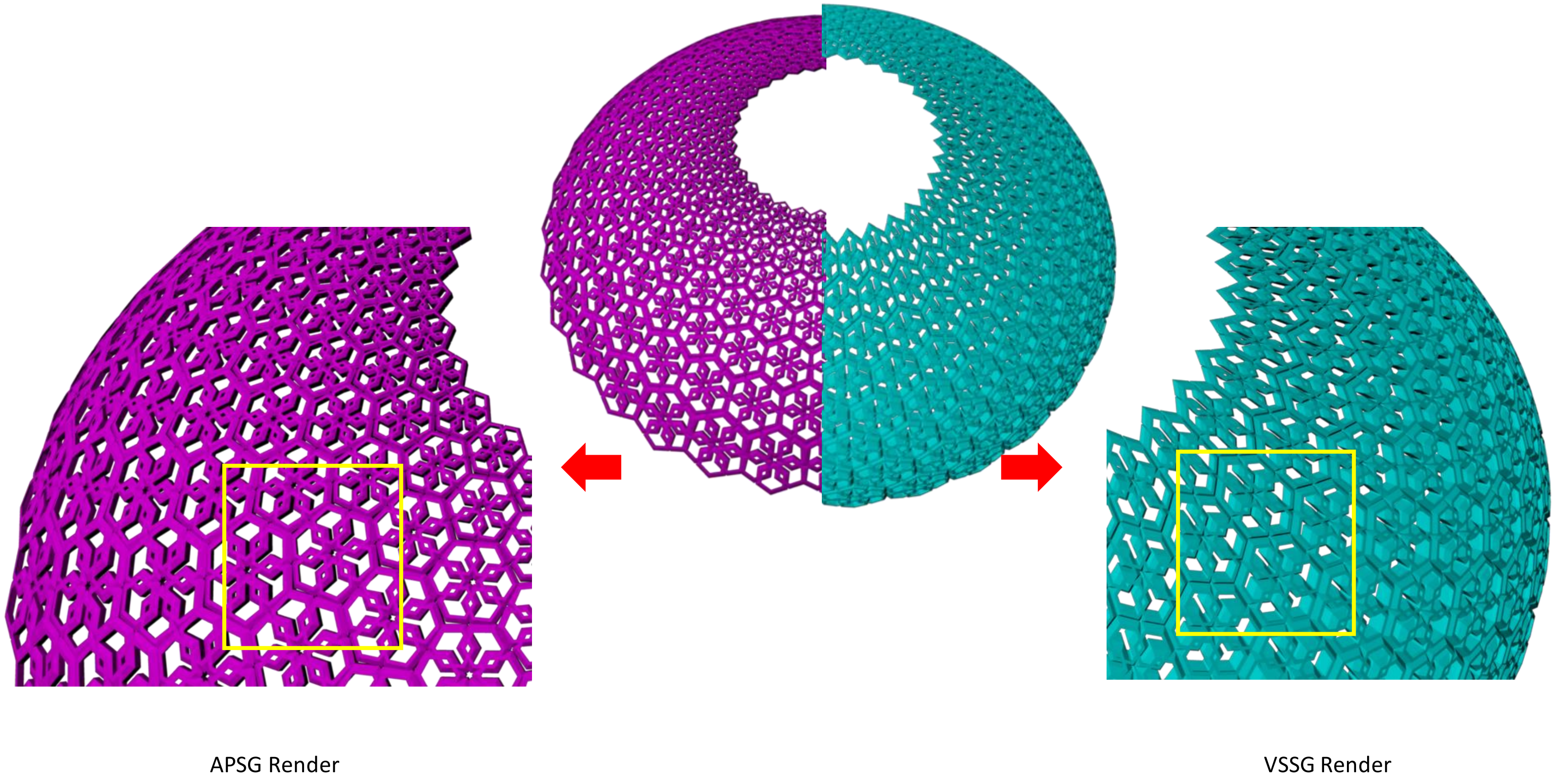


Figure 6.24 Dome renders (perspective view) of (Left) APSG (Right) VSSG.

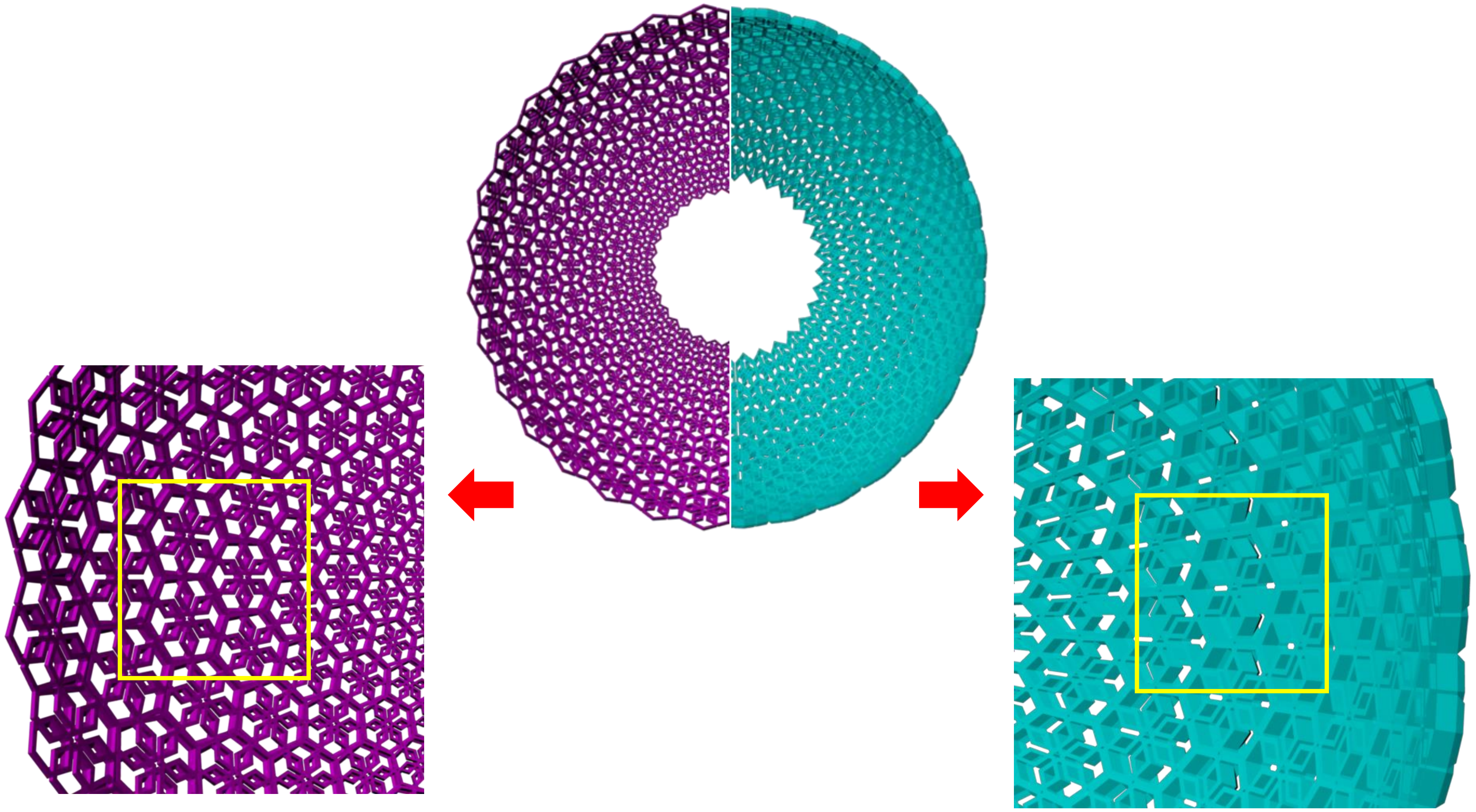


Figure 6.25 Dome renders (bottom view) of (Left) APSG (Right) VSSG.

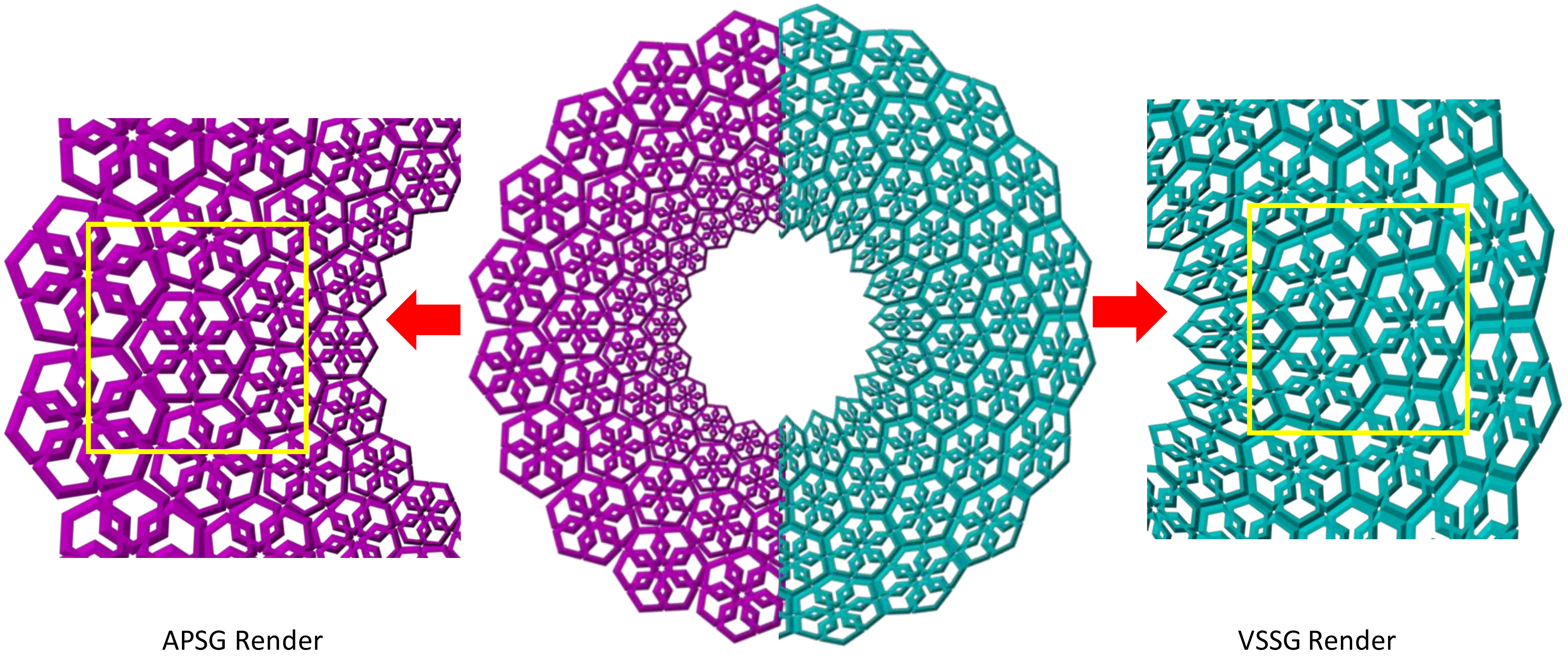


Figure 6.26 Star renders (front view) of (Left) APSG (Right) VSSG.

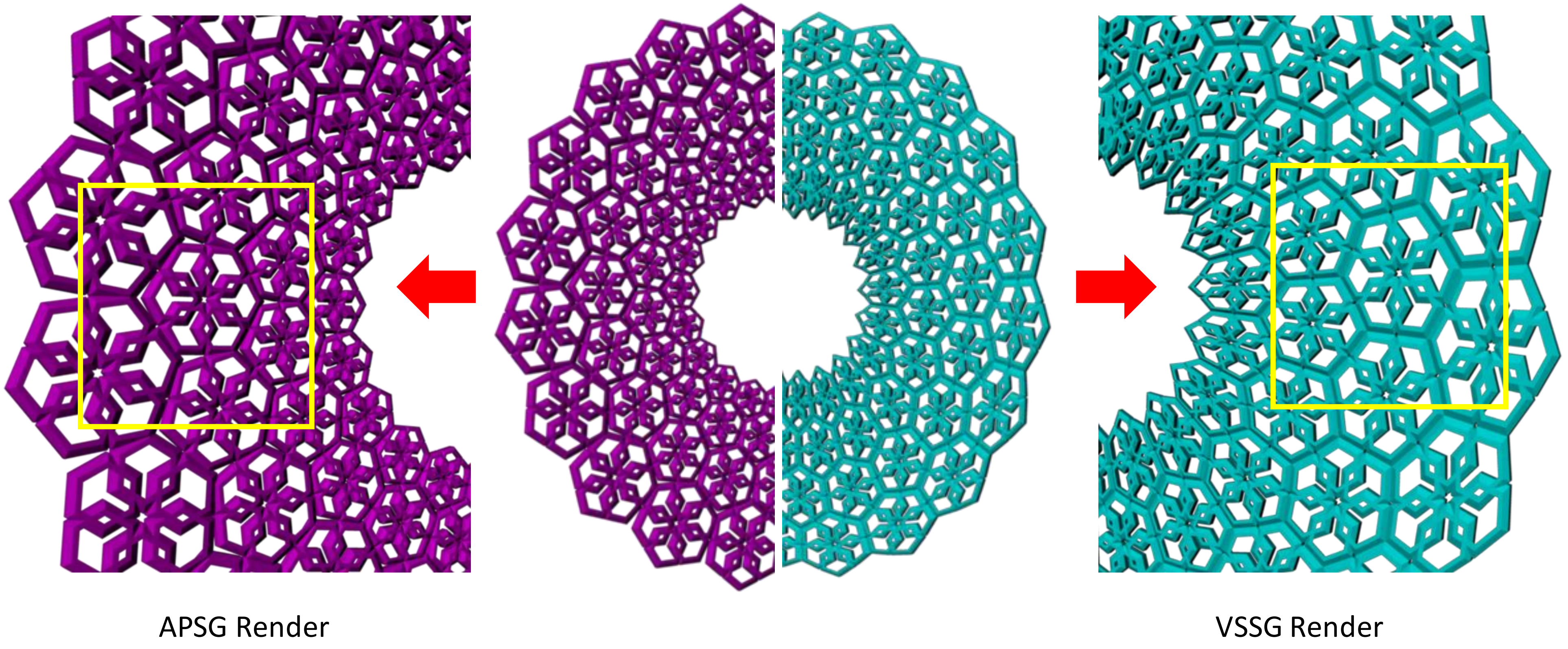


Figure 6.27 Star renders (perspective view) of (Left) APSG (Right) VSSG.

Looking at the renders in Figure 6.20Figure 6.27, firstly it can be seen that integrating the CPD and shell mapping has produced efficient results, as the gaps and overlaps have reduced vastly. The mapping of the motifs has resulted with a smooth mapping on a curved surface. The artefacts have only occurred by the use of the number of motifs. The limitation here is the amount of data points applied for the registration.

6.7 Application on Surfaces

This section describes how one can extend the VSSM algorithm to map the motifs onto any given 3D surface. It only provides a methodology of how it can be adapted.

The VSSM algorithm has been applied to form a wall, a column, a dome and the self-similarity star from point clouds. However by extending the algorithm it can be applicable for a given surface. With the provided 3D surface in form of polygons, applying the geometric technique of extrusion within the shell mapping method, the motifs can be mapped onto the surface.

The algorithm is extended in the initial stage of the texture space as shown in the flowchart in Figure 6.7. For a surface, to create the texture space data set the surface is separated into individual polygons and each polygonal face is extruded out to form the texture space. As the texture space is formed, the coordinates of each point of the texture space is taken as the registration points.

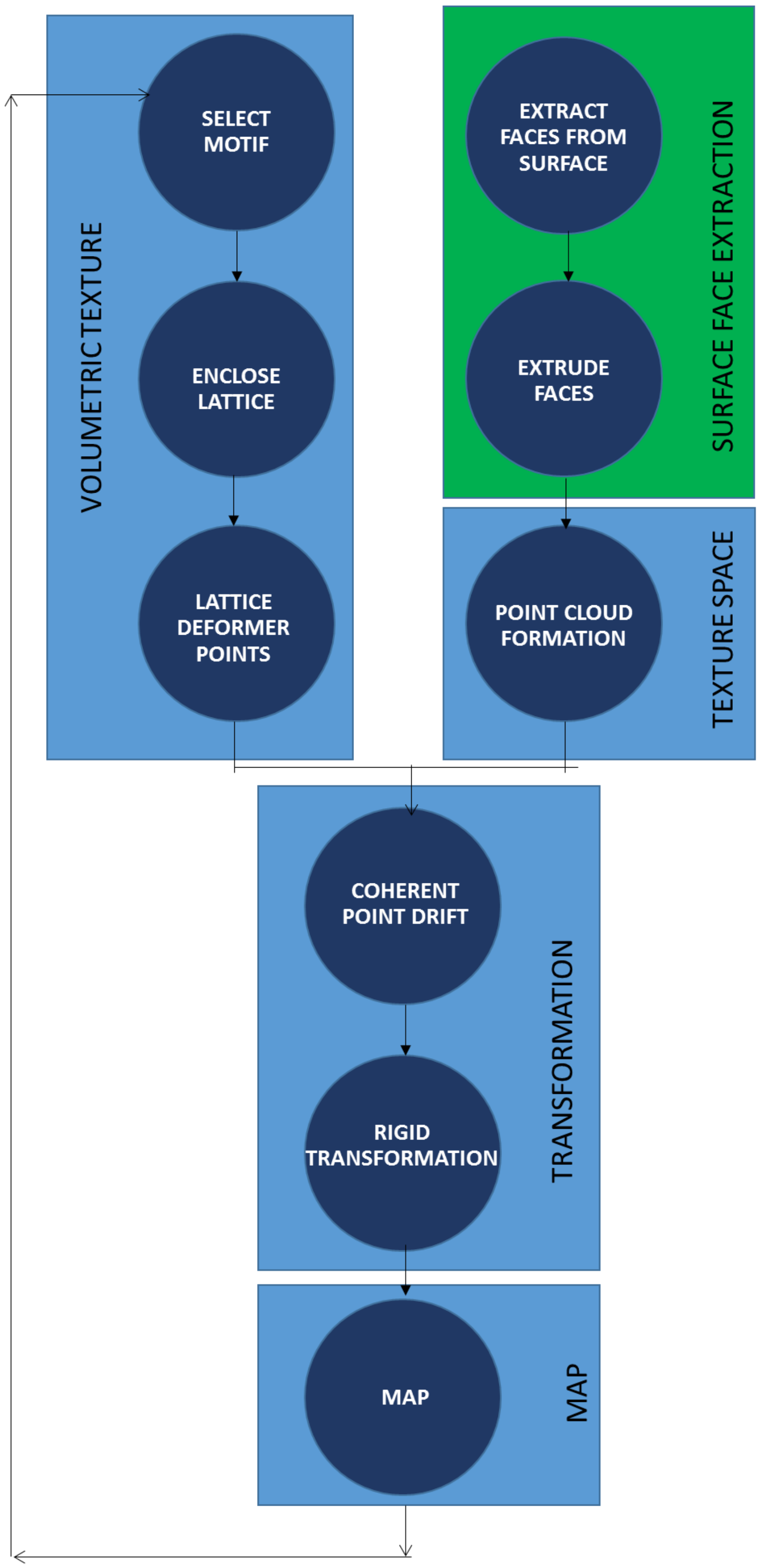


Figure 6.28 Flowchart for mapping onto given surface.

Collecting all the data points from the texture space, the surface can then be removed. The data points collected will form as an invisible point cloud and the process of mapping the motifs can be applied as normal.

6.8 Summary

This chapter has described the technique of integrating geometric modelling and registration methods into shape grammar, and has produced an efficient 3D mapping technique.

7. Conclusion and Future Works

"Education is the most powerful weapon which you can use to change the world."[7]

The aim of this research was to find an efficient 3D mapping technique to map 3D IGM onto any given surface with the use of Euclidean transformations, overcoming the problem of gap formations and overlaps. Furthermore, the method should be twofold, a) create a 3D IGM and b) apply as a mapping technique. From reviewing the construction methods of IGP, both traditional and computerized, it was seen that the shape grammar method had been applied previously to generate IGP. However, the implementations exhibited limitations such as producing a base (to tile) or only allowing a limited range of patterns to be generated. Additionally, it only produced 2D patterns.

The shape grammar method has many advantages:

- It can be parameterized, parameters which allow flexibility in forming shapes or patterns.
- It is not restricted to a given dimension, can be 1D, 2D or 3D.

[7] *Nelson Mandela.*

- Results are shown as the grammar is producing the patterns and therefore the formation of the pattern can be visualised at every step.

Hence, due to these advantages, the shape grammar technique was applied throughout this research from generating the 3D IGM to mapping techniques. To clarify, one can see the steps of the grammar producing the 3D motifs be running the python code in individual steps as shown in section 4.4.1.

The 3D IGM are produced by the PSG, allowing parameterization of the rules within the shape grammar, including Euclidean transformations and emergence rules. The rules were parameterized to create variations within the formation of the motifs. As mentioned previously, the unique feature of IGP is how symmetry plays a huge role; this provides the advantage of applying Euclidean transformations to form an IGM. The emergence rules are applied within the grammar, during the process of generating the patterns, hence both sets of rules coincide with each other. Previous methods that implemented shape grammar formed 2D IGP. Adding an extra dimension to the initial shape by extruding the front faces of the polygonal shape at a given angel, which is the initial pattern formation step, produced 3D IGM with a carved effect, eliminating the shape to look like a visualized extruded block. The PSG is an efficient grammar as it can produce more than 50 3D IGM of different categories from 4-6 fold 3D IGM. This is attributed to the changing of the initial shape, a method which has not been adopted previously.

IGP are adorned on architectural structures in a repetitive manner, with self-similarity; methods to generate such structures were analysed. A number of algorithms were formed to create simple structures like a wall, column, dome and a self-similarity star, using the shape grammar approach. The automated parameterized shape grammar (APSG) forms architectural structures by manipulating and mapping 3D IGM. The algorithm to form each structure was applied through the use of Euclidean transformations, initiating from translation and adding rotation and scale step by step. As the wall is a flat surface, the motifs translated in either the x- or y-axis. Adding the rotation with the translation formed the column, and finally the dome, which is a combination of translation, rotation and scale. An algorithm for a self-similarity star was also created, as tessellating motifs radially is a traditional characteristic in IGP. The APSG has shown that the shape grammar can also be applied as a mapping technique in 3D and not just for forming patterns in a restrictive manner.

The results showed how the APSG formed gaps and overlaps after generating curved structures like the dome. To improve upon this, geometric modelling and point registration methods were integrated into the shape grammar. The geometric modelling technique of shell mapping, enabled the formation of a shell space that can correspond with the texture space. This correspondence forms a direct transformation between the two spaces hence allowing a motif to be mapped into its correct position. The integration of CPD, a PSR technique, facilitated the application of the transformation matrix to the texture space. Working with only Euclidean transformations, the rigid CPD was applied to register the points between the two spaces. The VSSG, which is formed of the two integrated methods with shape grammar, produced efficient results through forming a curved structural surface, with very little gaps or overlaps.

7.1 Limitations

Although the grammars PSG, APSG and VSSG successfully generate motifs and motif based-structures, there are however some limitations.

Limitation 1

In generating the 3D motifs, a user is limited to creating a complete built motif as it is built motif from the GUI provided as it is built from the code produced in chapter 4. This allows one to visualize the generated result. However as mentioned previously, running the individual lines of the python code, one can see the pattern generation step by step. To overcome this limitation forming a GUI that will allow a user to visualize the steps taken to create an IGM, will show the simplicity behind forming complex islamic patterns.

Limitation 2

As the research was on forming individual 3D motifs and overcoming the problem of ‘overlaps’ when tessellating them on curved surfaces. There are however many IGP that are formed of motifs that are actually overlapped at a given distance. Overlapping motifs is a special feature in IGP as it forms motifs that cannot be distinguished at first glance, creating an illusion for the viewer. As the research applied the mathematical tessellation rule of ‘no gaps’ or overlaps, the overlapping of the motifs was not considered. The overlapping at a given distance of translation, rotation or scale, can add an extra complexity within a generated

3D IGM, resulting in a larger variety of tessellations on the given surface. Hence this limitation limits in the amount of tessellated patterns formed. Figure 7.1 shows a few examples of how the individual motif (in purple) is overlapped to form the pattern.

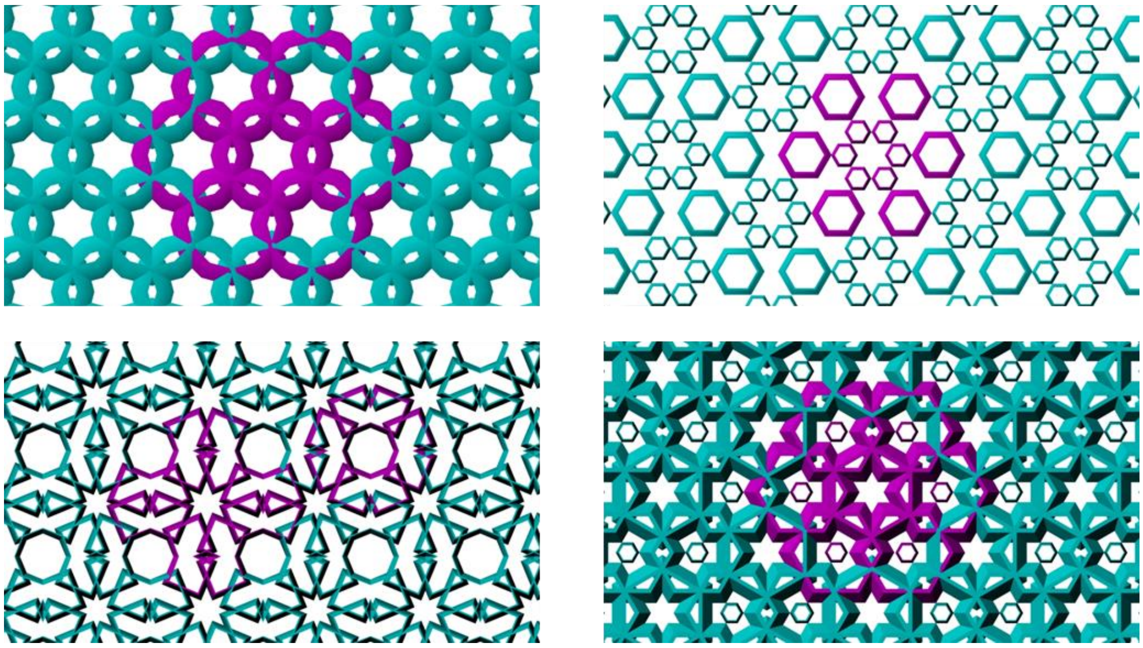


Figure 7.1 Overlapping of motifs.

Limitation 3

Although VSSM produced efficient mapping results, it still formed a few artefacts from the use of lower number of motifs on curved surfaces. For example from the registration results of the motifs in Figure. 6.2 (a) for an 8x8 square grid tessellation there was a larger gap in registering the point sets. The registration of the points was very rigid compared to a higher number of tessellated motifs. If the number of data points were increased, on both the lattice that encloses the motif and the provided point cloud, it may have given better results due to the flexibility and less rigidity.

7.2 Future Work

Although the results presented have demonstrated the effectiveness of the integration of geometric modelling and registration techniques into shape grammar, the research can be further developed by improving the existing research methodology and generating an improved 3D generalized procedural mapping technique. The research methodology can be enhanced in the following ways:

- To generate a methodology which would allow constructing a structure formed by more than one 3D motif. The beauty of Islamic geometric patterns is the complex interwoven patterns of a number of different motifs, where the interlocking of each motif is a visual illusion for the viewer.
- Creating extended patterns that do not restrict one to forming a central 3D motif.
- To allow one to map in a variety of polygonal grids, without the restriction of only the square and hexagonal grid.
- Expand the mapping to other types of genric geometric patterns, to form generalized mapping for any type of pattern.

Nonetheless , this research has taken full advantage of the shape grammar technique by forming grammars to generate both motifs and architectural structures. Thereby, producing a large variety of 3D IGM and a generalized mapping technique that can also be applied onto any given surface to map 3D IGM.

8. References

[1] S. R. Canby, *Islamic art in detail*. London: British Museum, 2005.

[2] J. F. Bonner, “The Historical Significance of the Geometric Designs in the Northeast Dome Chamber of the Friday Mosque at Isfahan,” *Nexus Netw. J.*, vol. 18, no. 1, pp. 55–103, 2016.

[3] S. Cenani and G. Cagdas, “A Shape Grammar Study: Form Generation with Geometric Islamic Patterns,” in *Proceedings of the 10th Generative Art Conference (GA2007)*, 2007, pp. 1–7.

[4] E. Ulu and S. M. Sener, “A Shape Grammar Model To Generate Islamic Geometric Pattern,” in *Proceedings of the 12th Generative Art Conference (GA2009)*, 2009, pp. 269–282.

[5] I. Jowers, M. Prats, H. Eissa, and J.-H. Lee, “A Study of Emergence in the Generation of Islamic geometric Patterns’,” in *New Frontiers: Proceedings of the 15th International Conference on Computer-Aided Architectural Design Research in Asia*, 2010, pp. 39–48.

[6] S. J. Abas and A. S. Salman, *Symmetries of Islamic geometrical patterns*. London: World Scientific, 1994.

[7] D. Sutton, *Islamic Design: A Genius for Geometry*. USA: Bloomsbury Publishing, 2007.

[8] C. F. Baker, *Qur’an Manuscripts : Calligraphy, Illumination and Design*. London: British Library, 2007.

[9] A. M. Aljamali, “Classification and Design of Islamic Geometric Patterns Using Computer Graphics,” in *Second International Conference in Visualisation (VIS’09)*, 2009, pp. 253–258.

[10] S. J. Abas and A. S. Salman, “Geometric and Group-theoretic Methods for Computer Graphic Studies of Islamic Symmetric Patterns,” *Comput. Graph. Forum*, vol. 11, no. 1, pp. 43–53, 1992.

[11] C. . Kaplan, “Islamic star patterns from polygons in contact,” in *Proceedings of Graphics Interface (GI’05)*, 2005, pp. 177–185.

[12] N. Inception, “Islamic Design Services for Architects, Interior Designers and Developers,” 2013. [Online]. Available: http://www.nomadinception.com/.

[13] R. Tennant, “Medieval Islamic Architecture, Quasicrystals, and Penrose and Girih Tiles: Questions from the Classroom,” in *Bridges Proceedings 2008: Mathematical Connections in Art, Music and Science*, 2008, pp. 297–304.

[14] M. M. Pickthall, *The Glorious Qur’an: Text and Explanatory Translation*, 2nd ed. London: Tahrike Tarsile Qur’an, 1996.

[15] J. S. Abas, “Islamic Geometrical Patterns for the Teaching of Mathematics of Symmetry,” *Symmetry Cult. Sci.*, vol. 12, no. 1, pp. 53–65, 2001.

[16] G. J. Dalu; Michell, *The Arts of Islam - Exhibition catalogue Hayward Gallery*. Arts Council of Great Britain, 1976.

[17] P. R. Cromwell, “Islamic Geometric Designs from the Topkapı Scroll II: A Modular Design System,” *J. Math. Arts*, vol. 4, no. 3, pp. 119–136, 2010.

[18] G. Necipoğlu, *The Topkapı Scroll : Geometry and Ornament in Islamic Architecture*. Getty Research Institute, 1996.

[19] C. Bier, “Geometric Patterns and The Interpretation of Meaning: Two Monuments in Iran,” in *Bridges: Mathematical Connections in Art, Music and Science*, 2002, pp. 67–78.

[20] A. Özdural, “Mathematics and Arts: Connections between Theory and Practice in the Medieval Islamic World,” *Hist. Math.*, vol. 27, no. 2, pp. 171–201, May 2000.

[21] B. L. Bodner, “A Nine- and Twelve-Pointed Star Polygon Design of the Tashkent Scrolls,” in *Proceedings of Bridges 2011: Mathematics, Music, Art, Architecture, Culture*, 2011, pp. 147–154.

[22] B. L. Bodner, “From Sultaniyeh to Tashkent Scrolls: Euclidean Constructions of Two Nine- and Twelve-Pointed Interlocking Star Polygon Designs,” *Nexus Netw. J.*, vol. 14, no. 2, pp. 307–332, 2012.

[23] P. R. Cromwell, “A Modular Design System Based on the Star and Cross Pattern,” *J. Math. Arts*, vol. 6, no. 1, pp. 29–42, 2012.

[24] E. Broug, *Islamic Geometric Design*. London: Thames and Hudson, 2013.

[25] D. Wade, “Patterns in Islamic Art,” 2017. [Online]. Available: http://patterninislamicart.com/.

[26] P. Lu and P. Steinhardt, “Decagonal and Quasi-Crystalline Tilings in Medieval Islamic Architecture,” *Science (80-.).*, vol. 315, no. 5815, pp. 1106–1110, 2007.

[27] Y. Aboufadil, A. Thalal, and M. A. E. I. Raghni, “Symmetry groups of Moroccan Geometric Woodwork Patterns,” *J. Appl. Crystallogr.*, vol. 46, no. 6, pp. 1834–1841, 2013.

[28] G. Riether and D. Baerlecken, “Digital Girih, A Digital Interpretation Of Islamic Architecture,” *Int. J. Archit. Comput.*, vol. 10, no. 1, pp. 1–12, 2012.

[29] P. R. Cromwell, “The Search for Quasi-Periodicity in Islamic 5-fold Ornament,” *Math. Intell.*, vol. 31, no. 1, pp. 36–56, 2009.

[30] J. Bonner, “Three Traditions of Self-Similarity in Fourteenth and Fifteenth Century Islamic Geometric Ornament,” in *ISAMA-BRIDGES Conference*, 2003, pp. 1–12.

[31] H. Lalvani, “Coding and generating complex periodic patterns,” *Vis. Comput.*, vol. 5, no. 4, pp. 180–202, Jul. 1989.

[32] A. Aljamali and E. Banissi, “Normalised grid/motif based patterns-Islamic geometric patterns,” *Inf. Vis. 1997. …*, 1997.

[33] G. Stiny and J. Gips, “Shape Grammars and the Generative Specification of Painting and Sculpture,” *Int. Fed. Inf. Process. Congr.*, vol. 2, no. 3, pp. 125–135, 1971.

[34] M. Gross, “Emergence in a Recognition Based Drawing Interface,” *Vis. Spat. Reason. II*, pp. 51–65, 2001.

[35] N. F. Ismail, F. A. Ishak, J. Tumin, N. Yusup, and E. E. Rupiwin, “Geometric Shapes Generation in Songket Designs Using Shape Grammar,” in *Proceedings of the 12th International Conference on Applied Computer and Applied Computational Science (ACACOS ’13)*, 2013, pp. 153–157.

[36] J. Duarte, “Towards the mass customization of housing: the grammar of Siza’s houses at Malagueira,” *Environ. Plan. B Plan. Des.*, vol. 32, no. 1, pp. 347–380, 2005.

[37] H. H. Chau, X. Chen, A. McKay, and A. de Pennigton, “Evaluation of a 3D shape grammar implementation,” in *Design Computing and Cognition ’04*, 2004, pp. 357–376.

[38] G. Stiny and W. Mitchell, “The Palladian Grammar,” *Environ. Plan. B*, vol. 5, no. 1, pp. 5–15, 1978.

[39] G. Stiny and W. Mitchell, “The Grammar of Paradise: on the Generation of Mughul Gardens,” *Environ. Plan. B*, vol. 7, no. 2, pp. 209–226, 1980.

[40] H. Koning and J. Eizenberg, “The Language of the Prairie: Frank Lloyd Wright’s Prairie Houses,” *Environ. Plan. B*, vol. 8, no. 1, pp. 295–323, 1981.

[41] U. Flemming, “More than the sum of parts: the grammar of Queen Anne houses,” *Environ. Planin. B Plan. Des.*, vol. 14, no. 1, pp. 323–350, 1987.

[42] J. Noghani, E. Anderson, and F. Liarokapis, “Towards a Vitruvian Shape Grammar for Procedurally Generating Classical Roman Architecture,” in *The 13th International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST)*, 2012, pp. 41–44.

[43] S. Cenani and G. Cagdas, “Shape Grammar of Geometric Islamic Ornaments,” in *Comunicating Space(s): 24th eCAADe Conference*, 2006, pp. 290–297.

[44] P. S. Heckbert, “Fundamentals of Texture Mapping and Image Warping,” University of California, 1989.

[45] P. S. Heckbert, “Survey of Texture Mapping,” *Comput. Graph. Appl.*, vol. 6, no. 11, pp. 56–67, 1986.

[46] C. Koniaris, D. Cosker, and X. Yang, “Survey of Texture Mapping Techniques for Representing and Rendering Volumetric Mesostructure,” *J. Comput. Graph. Tech.*, vol. 3, no. 2, pp. 18–60, 2014.

[47] L. Szirmay-Kalos and T. Umenhoffer, “Displacement Mapping on the GPU — State of the Art,” *Comput. Graph. Forum*, vol. 27, no. 6, pp. 1567–1592, 2008.

[48] S. H. Westin, J. R. Arvo, K. E. Torrance, S. H. Westin, J. R. Arvo, and K. E. Torrance, “Predicting Reflectance Functions from Complex Surfaces,” *ACM SIGGRAPH Comput. Graph.*, vol. 26, no. 2, pp. 255–264, 1992.

[49] N. Badler, R. Bajcsy, N. Badler, and R. Bajcsy, “Three-dimensional Representations for Computer Graphics and Computer Vision,” *ACM SIGGRAPH Comput. Graph.*, vol. 12, no. 3, pp. 153–160, 1978.

[50] C. Schlick, “A Survey of Shading and Reflectance Models,” *Comput. Graph. Forum*, vol. 13, no. 2, pp. 121–131, 1994.

[51] J.-F. Dufort, L. Leblanc, and P. Poulin, “Interactive Rendering of Meso-Structure Surface Details using Semi-Transparent 3D Textures,” in *Proceedings of Vision, Modeling, and Visualization*, 2005, pp. 399–406.

[52] Catmull and E. Earl, “A subdivision algorithm for computer display of curved surfaces.” The University of Utah, 1974.

[53] J. F. Blinn, “Simulation of Wrinkled Surfaces,” *ACM SIGGRAPH Comput. Graph.*, vol. 12, no. 3, pp. 286–292, 1978.

[54] G. Turk, “Generating Textures on Arbitrary Surfaces using Reaction-Diffusion,” *ACM SIGGRAPH Comput. Graph.*, vol. 25, no. 4, pp. 289–298, 1991.

[55] J. Cohen, M. Olano, and D. Manocha, “Appearance-Preserving Simplification,” in *25th annual conference on Computer Graphics and Interactive Techniques*, 1998, pp. 115–122.

[56] R. Cook, “Shade trees,” *ACM Siggraph Comput. Graph.*, vol. 18, no. 3, pp. 223–231, 1984.

[57] L. Wang, X. Wang, X. Tong, S. Lin, S. Hu, B. Guo, and H.-Y. Shum, “View-Dependent Displacement Mapping,” *ACM Trans. Graph.*, vol. 22, no. 3, pp. 334–339, 2003.

[58] T. Kaneko, T. Takahei, M. Inami, and N. Kawakami, “Detailed shape representation with parallax mapping,” in *ICAT*, 2001, pp. 1–4.

[59] A. Halli, A. Saaidi, K. Satori, and H. Tairi, “Per-Pixel Displacement Mapping Using Cone Tracing,” *Softw. (I. Re. Co. S)*, 2008.

[60] M. McGuire and M. McGuire, “Steep parallax mapping,” in *Computational Graphics*, 2005, pp. 1–2.

[61] M. Premecz, “Iterative Parallax Mapping with Slope Information.”

[62] M. M. Oliveira, G. Bishop, and D. McAllister, “Relief texture mapping,” in *27th Annual Conference on Computer Graphics and Interactive Techniques*, 2002, pp. 359–368.

[63] A. Depeursinge, A. Foncubierta-Rodriguez, D. Van De Villle, and H. Muller, “Three-Dimensional Solid Texture Analysis in Biomedical Imaging: Review and Opportunities,” *Med. imaging Anal.*, vol. 18, no. 1, pp. 176–196, 2014.

[64] J. T. Kajiya and T. L. Kay, “Rendering Fur with Three Dimensional Textures,” *ACM SIGGRAPH Comput. Graph.*, vol. 23, no. 3, pp. 271–280, 1989.

[65] A. Meyer and F. Neyret, “Interactive volumetric textures,” in *Rendering Techniques*, 1998, pp. 157–168.

[66] S. D. Porumbescu, B. Budge, L. Feng, and K. I. Joy, “Shell Maps,” *ACM Trans. Graph.*, vol. 24, no. 3, pp. 626–633, 2005.

[67] R. C. Nelson and R. Polana, “Qualitative Recognition of Motion using Temporal Texture,” *CVGIP Image Underst.*, vol. 56, no. 1, pp. 78–89, 1992.

[68] S. Dubois, R. Péteri, and M. Ménard, “Characterization and Recognition of Dynamic Textures based on 2D+T Curvelet Transform,” *Signal, Image Video Process.*, vol. 9, no. 4, pp. 819–830, 2015.

[69] F. Policarpo and M. M. Oliveira, “Relief Mapping of Non-Height-Field Durface Details,” in *Proceedings of the 2006 symposium on Interactive 3D graphics and games (SI3D ’06)*, 2006, pp. 55–62.

[70] M. Pharr and R. Fernando, *GPU Gems 2 : Programming Techniques for High-Performance Graphics and General-Purpose Computation*. Addison-Wesley, 2005.

[71] A. Hornung, K. Wurm, M. Bennewitz, and C. Stachniss, “OctoMap: An Efficient Probabilistic 3D Mapping Framework based on Octrees,” *Auton. Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[72] K. J. Dana, B. van Ginneken, S. K. Nayar, and J. J. Koenderink, “Reflectance and Texture of Real-World Surfaces,” *ACM Trans. Graph.*, vol. 18, no. 1, pp. 1–34, 1999.

[73] K. Perlin, “An Image Synthesizer,” *ACM SIGGRAPH Comput. Graph.*, vol. 19, no. 3, pp. 287–296, 1985.

[74] K. Perlin, E. M. Hoffert, K. Perlin, and E. M. Hoffert, “Hypertexture,” in *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques*, 1989, vol. 23, no. 3, pp. 253–262.

[75] M. Knuth, J. Bender, M. Goesele, and A. Kuijper, “Deferred Warping,” *IEEE Comput. Graph. Appl.*, vol. PP, no. 99, pp. 1–9, 2016.

[76] K. Erleben and H. Dohlmann, “The Thin Shell Tetrahedral Mesh,” in *Proceedings of DSAGM*, 2004, pp. 94–102.

[77] L. Guibas and J. Stolfi, “Primitives for the Manipulation of General Subdivisions and the Computation of Voronoi,” *ACM Trans. Graph.*, vol. 4, no. 2, pp. 74–123, 1985.

[78] J. Peng, D. Kristjansson, D. Zorin, J. Peng, D. Kristjansson, and D. Zorin, “Interactive Modeling of Topologically Complex Geometric Detail,” *ACM Trans. Graph.*, vol. 23, no. 3, pp. 635–643, 2004.

[79] X. W. Tsinghua, X. Tong, S. Lin, S. Hu, B. Guo, and H.-Y. Shum, “Generalized Displacement Maps.,” in *Proceedings of the Fifteenth Eurographics conference on Rendering Techniques (EGSR’04)*, 2004, pp. 227–233.

[80] J. Hirche, A. Ehlert, S. Guthe, and M. Doggett, “Hardware Accelerated Per-Pixel Displacement Mapping,” in *Proceedings of Graphics Interface*, 2004, pp. 153–158.

[81] S. Jeschke, S. Mantler, and M. Wimmer, “Interactive Smooth and Curved Shell Mapping,” in *Proceedings of the 18th Eurographics Coference on Rendering and Techniqes*, 2007, pp. 351–360.

[82] N. Ritsche, “Real-time shell space rendering of volumetric geometry,” in *Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia (GRAPHITE ’06)*, 2006, pp. 265–274.

[83] P. Decaudin and F. Neyret, “Volumetric Billboards,” *Comput. Graph. Forum*, vol. 28, no. 8, pp. 2079–2089, 2009.

[84] A. Brodersen, K. Museth, S. Porumbescu, and B. Budge, “Geometric Texturing Using Level Sets,” *IEEE Trans. Vis. Comput. Graph.*, vol. 14, no. 2, pp. 277–288, 2008.

[85] P. Webster, “Fractal Islamic Geometric Patterns Based on Arrangements of {n/2} Stars,” in *Proceedings of Bridges 2013: Mathematics, Music, Art, Architecture, Culture*, 2013, pp. 87–94.

[86] G. Stiny, *Shape: talking about seeing and doing*. MIT Press, 2008.

[87] K. Critchlow, *Islamic patterns*. Thames and Hudson, 1976.

[88] J. Bourgoin, *Arabic geometrical pattern and design.* Dover Publications, 1973.

[89] J. Duarte and J. Rocha, “Unveiling the structure of the Marrakech Medina: A shape grammar and an interpreter for generating urban form,” *Artif. Intell. Eng. Des. Anal. Manuf.*, vol. 21, no. 4, pp. 317–349, 2007.

[90] B. Grünbaum and G. Shephard, *Tilings and patterns*, 2nd ed. Dover Publications, 1987.

[91] H. S. M. Coxeter and W. O. J. Moser, *Generators and Relations for Discrete Groups*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1980.

[92] B. Cipriani and W. Lau, “Construction Techniques in Medieval Cairo: the Domes of Mamluk Mausolea (1250 AD-1517A. D.),” in *Second International Congress on construction history*, 2006, pp. 695–716.

[93] P. J. Besl and N. D. McKay, “Registration of 3-D Shapes,” *Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, 1992.

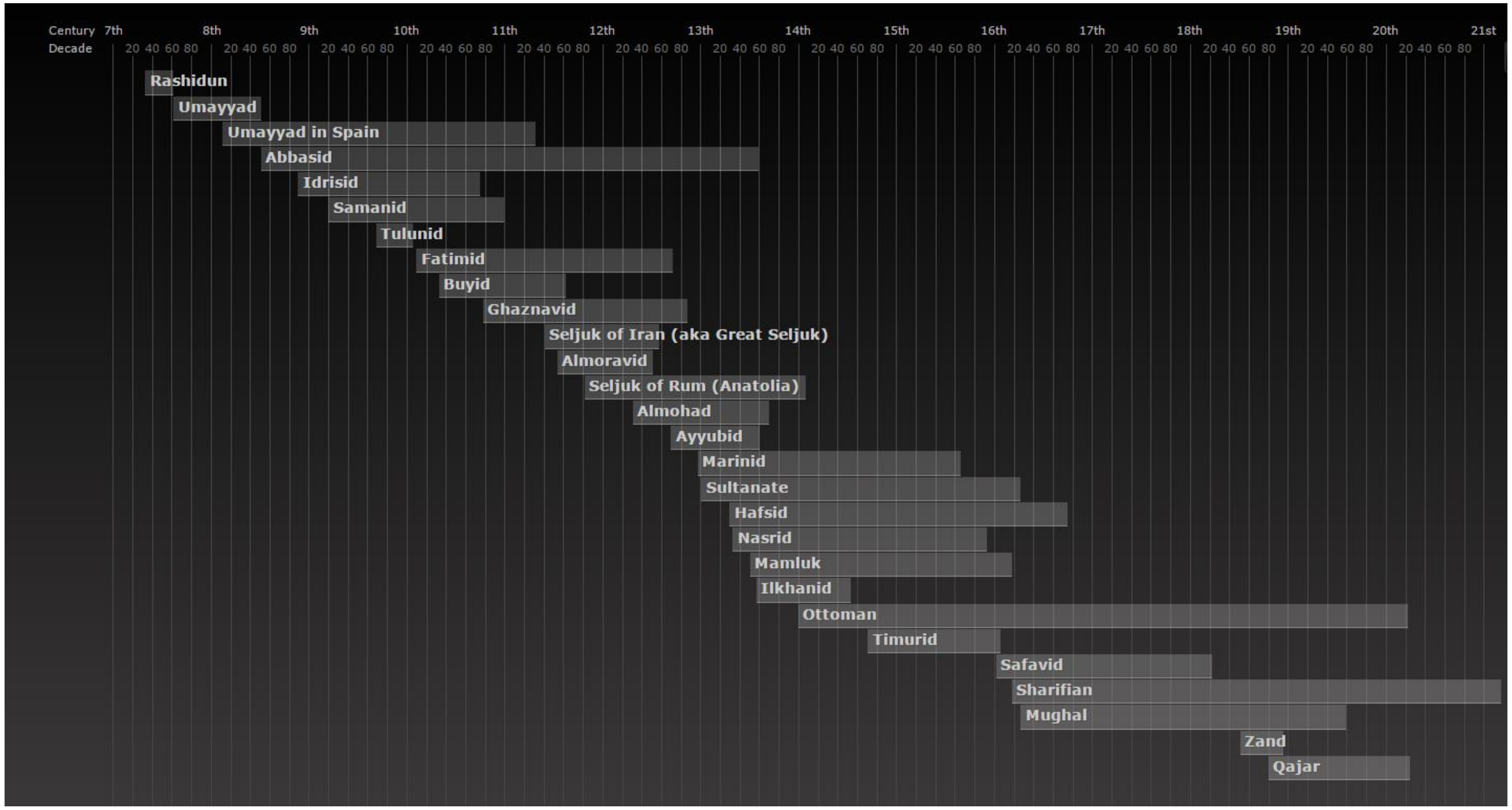
[94] A. Rangarajan, H. Chui, E. Mjolsness, S. Pappu, L. Davachi, P. Goldman-Rakic, and J. Duncan, “A Robust Point-Aatching Algorithm for Autoradiograph Alignment,” *Med. Image Anal.*, vol. 1, no. 4, pp. 379–398, 1997.

[95] Bin Luo and E. R. Hancock, “Structural Graph Matching using the EM Algorithm and Singular Value Decomposition,” *Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 10, pp. 1120–1136, 2001.

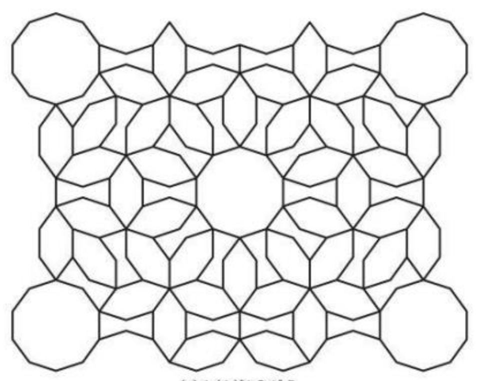
[96] Yefeng Zheng and D. Doermann, “Robust Point Matching for Nonrigid Shapes by Preserving Local Neighborhood Structures,” *Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 643–649, 2006.

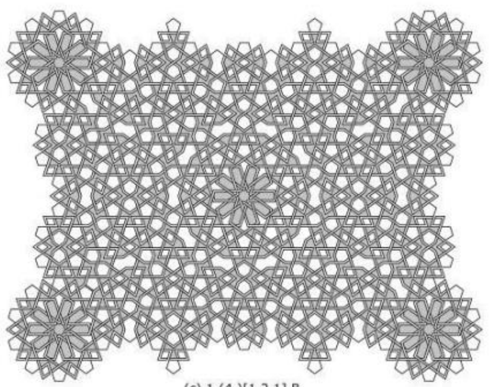
[97] A. Myronenko and Xubo Song, “Point Set Registration: Coherent Point Drift,” *Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 12, pp. 2262–2275, 2010.

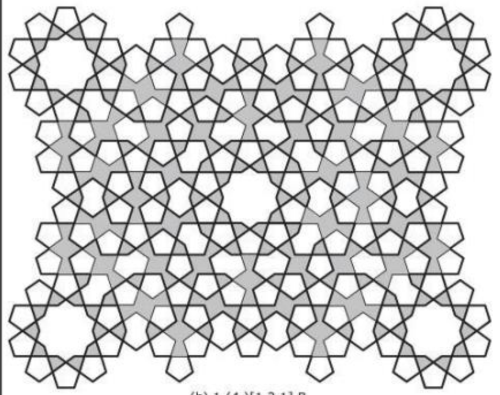
Appendix A – Timeline of the Dynasties

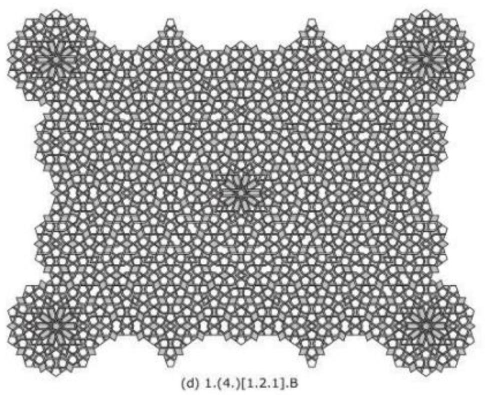


Appendix B – Template Covered by 3 Groups of Covering Tiles (Derivative).

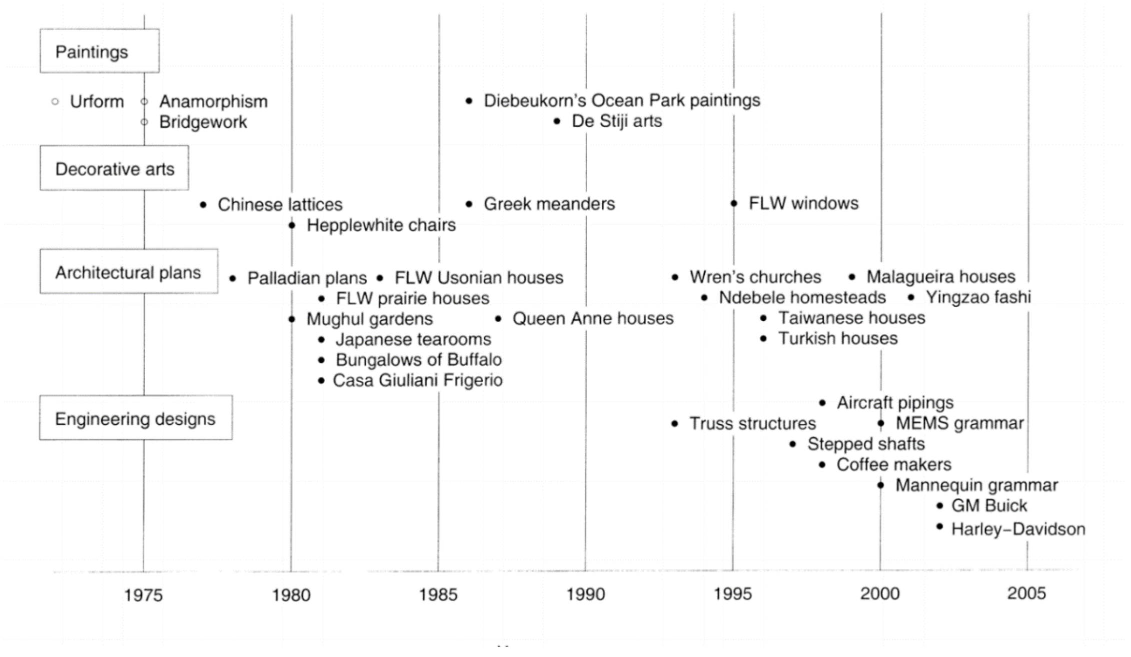

(a) 1.(4.)[1.2.1].B


(c) 1.(4.)[1.2.1].B


(b) 1.(4.)[1.2.1].B


(d) 1.(4.)[1.2.1].B

Appendix C – Shape Grammar Implementation


Paintings
Urform
Anamorphism
Bridgework
Diebeukorn's Ocean Park paintings
De Stiji arts
Decorative arts
Chinese lattices
Hepplewhite chairs
Greek meanders
FLW windows
Architectural plans
Palladian plans
FLW Usonian houses
FLW prairie houses
Mughul gardens
Japanese tearooms
Bungalows of Buffalo
Casa Giuliani Frigerio
Queen Anne houses
Wren's churches
Ndebele homesteads
Taiwanese houses
Turkish houses
Malagueira houses
Yingzao fashi
Engineering designs
Truss structures
Aircraft pipings
MEMS grammar
Stepped shafts
Coffee makers
Mannequin grammar
GM Buick
Harley-Davidson