Post-disaster Temporary Housing System based on Generative Design Method

Filipe G. Afonso and Jun C. Lu
School of Architecture, Guangzhou University, Guangzhou, China
Email: filipe.afonso@gzhu.edu.cn, 173395753@qq.com

***Abstract*—In the current context of frequent natural disasters in China, this paper chooses to combine computer generative design methods with post-disaster modular temporary housing as a research topic, and studies the architectural generative design method and the temporary resettlement of disaster victims from a new perspective. Architecture is being nowadays radically rethought in the design processes it employs, moving away from the traditional approach of individual 'signature' buildings or from a simply functional design, to the logics of nonlinearity, in which evolutionary intelligence plays an important role. This is a way of creating architectural structures as a result of a programmed system. In particular, the paper aims to develop and explore computer programming processes to quickly generate a modular housing combination method to provide an efficient and scientific post-disaster modular temporary housing design proposal. This research will provide new possibilities for temporary housing settlements for disaster victims in China and a new approach in computational architectural design under this context.**

***Index Terms*—computational design, temporary housing, modularity, generative design, architecture**

I. Introduction

This research aims to develop a computational methodology to generate a non-standard modular temporary housing system and a diversity of planning settlement combinations to provide an efficient post-disaster modular temporary housing design proposal. Such study is a new approach to the design of temporary settlements for disaster victims in China, by proposing a computationally-oriented architecture. The computational methodology conceived, will allow to test an efficient and fast design process that can be adaptable to different site conditions. It means that the abstract computational algorithms (or processes) can be programmed to match local geographic, topographic or sociological conditions, and to the specific needs of the end users.

The paper is organized in three parts: firstly, overview of the relevant contents of modular temporary housing after the disaster and analysis of some of the existing generative design methods; Then, the definition of a generative design method, by developing a computer program for Graphic Applications with a specific output related to modular housing systems. The computational method will generate a modular housing system and will allow to combine and aggregate different residential units. This computational method contributes to the discussion about designing with architectural parameters, responding to functional needs of populations, but at the same time allowing a great freedom in the combination of the architectural forms. Along the paper the medium of programming allowed to model and control complex residential settlements and to refine a computational design process that simulates real local conditions in post-disaster environment; Finally, the computational method will allow the generation of endless variations in the organization of the housing units and urban settlements. Such emergent results will be critically analyzed and conclusions will be drawn.

Through the use of programming language for architectural design, this study aims to combine digital technology with the actual situation of disaster relief in China, and greatly improve the design quality of the resettlement layout of temporary housing.

II. Contextualizing Post-disaster Modular Systems

The current studies on modular design of buildings made in China include the work of the academician Peng Yigang [1] from the perspective of space combination and the restrictions of modular building structure on space combination. Bao Jiasheng in the 1970s [2] explained the functional flexibility of the modular space. Tong Shizhong [3] in 2000 introduced the modular theory in detail, and summarized the modularization system as part of a design theory.

Since the Wenchuan Earthquake in 2008, Chinese scholars have conducted a deep analysis on the problems of post-disaster reconstruction in affected areas from the perspective of sociology, economics, architecture, and planning. For example, Qiu Baoxing [4] introduced post-earthquake emergency measures, including the construction of temporary resettlement houses in various regions in China. Guan Pingying [5] analysed the design of the reconstruction of Jintai Village after the Wenchuan disaster, including in his study the complexities of rural construction, and presented Jintai Village project as a well succeeded prototype of a community. The research

Manuscript received December 7, 2020; revised April 11, 2021; accepted May 8, 2021.

doi: 10.18178/ijscer.10.2.80-84

work of Wei Peng [6] focus on the design of post-disaster shelters, including housing planning, structural design, construction methods. In terms of post-disaster reconstruction, the outcome from the above-mentioned authors, except the Jintai Village project by John Lin and Joshua Bolchover, and do not concern the architectural space composition.

III. Contextualizing Generative Systems

Combinatorial theory and the configuration of modular components as generative techniques have been applied extensively in biological systems, as is the example of the work of Lindenmayer [7] and Flake [8]. But only recently such strategies are being used in architectural and urban design. Maeda [9] has been a pioneer in this matter with the introduction of computation as an increasingly easy-to-use tool applied to visual graphics applications. The work of Stiny [10], merging computational procedures, combinatorial optimization and design inspired the new generations of architects making use of the computational power of modern desktop computers. In particular the work of Herbert Simon demonstrated to be inspirational for the development of this research project. Simon presents several strategies in applying combinatory and optimization logics to find alternatives in the design of real-world problems. As Afonso and Magalhães [11], these experimental processes are unlocking architects from the exclusively anthropological point of view in producing architecture. In a time of perceptible technological change, and through the application of the computational force, design is becoming a more experimental practice.

Generative design and genetic algorithms are being applied in China to different research fields. For example, Zhang Peihong et al. [12] applied the improved adaptive ant colony algorithm to optimize the fire protection evacuation path in large public buildings. There are few studies on the application of genetic algorithms in building design in China, mainly by Li Biao [13] in 2006, with projects like "Happy Lattices" and "Cube1001".

IV. Generation of a Computational Design Method

The success of the housing models was achieved by the intricate manipulations made from the departure point of the generator elements, using bottom-up techniques. Within bottom-up strategy, the information processing gives rise to complex systems, by piecing together single elements, “functional units”. These generative elements (or functional compartments of the house) were linked to shape larger structures until a desirable top level is achieved. This strategy resembles the behavior of genetic code; in fact, each different compartment of the house functions as a genotype.

This section explains how to generate housing structures by combining usable functional spaces, based on information such as the number of functions, connection types, and connection rules predetermined by the designer or the end user. In real life, the connection relationship between rooms is usually not random. For example, the living room can be connected to both the bedroom and the kitchen, and the bedroom and kitchen should not be directly connected. To ensure that the generated functional topology relationship is logical, an effective mechanism is needed to describe this unequal connection relationship. This experiment proposes a description mechanism: the "topological relationship matrix”, assuming N types of functions, the allowed connections between them is represented by an N × N matrix. The functional units of the temporary housing are divided into four parts: an activity room (second bedroom), a master bedroom, a kitchen/ or toilet, and a connecting unit living room. The functional topology relationship stipulates that the activity room can be connected to the master bedroom, kitchen toilet, and living room, while the bedroom cannot be directly connected to the kitchen toilet.

A. *Computational Housing System*

The computational housing system uses four two-dimensional arrays to generate the residential units and its constituent space compartments. Compartments, or functional spaces, can be divided into the following two attributes: ① building physical compartments, such as functional rooms are represented by 1; ② non-built compartments are represented by 0; ③ constant spaces, such as the designer's established road accesses or site properties that cannot be changed, are represented by -1. The program must avoid above-mentioned constant variables during the running process. When formulating the rules, lighting, ventilation of the building and the changes in the internal and external space forms are as follows:

- **Rule a1**: the site is divided into a nine-cell grid based on the central cell, and each cell of the site is initialized. The unit is divided into physical space (attribute 1) and non-physical space (attribute 0) Two conditions. The neighbor cells are located at the top, bottom, left, and right of the center cell, which indicates that attribute 1 can be the physical space, and the neighbor cells that are located at the upper left, lower left, upper right, and lower right indicate that the attribute 0 is non-physical space.
- **Rule a2**: pre-designed functional spaces corresponding to the activity room, master bedroom, kitchen - toilet, and living room are placed into the grid in order according to the established connectivity rules. It means, the activity room can be connected with the master bedroom and kitchen. The toilet and the living room are connected. The master bedroom can be connected with the activity room and the living room. The kitchen - toilet can be connected with the activity room and the living room. The living room can be connected with any functional unit.

- **Rule a3**: On the basis of the functional topological relationship that complies with rule a2, it is also necessary to meet the requirements of lighting and ventilation of the activity room, the master bedroom and the living room, and check whether each unit that needs to face South meets at least one lighting surface, which can be non-built space attribute or a constant spaces (see Fig. 1).

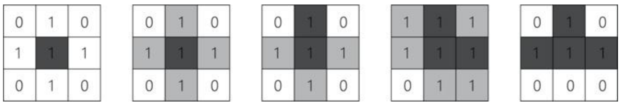

Figure 1. Configuration of a rule system based on space functions.

B. *Code, Process and Design*

From the previous relationship matrix, the computer script generates 720 results that meet the functional topological requirements, and then further filters to remove duplicates, obtaining 320 non-repeating functional topological relationship results. Based on preset rule a3, the results that meet the south-facing rule condition for the activity room, master bedroom, and living room need to be screened. The final program generates 28 results that meet all the preset rules (Fig. 2).

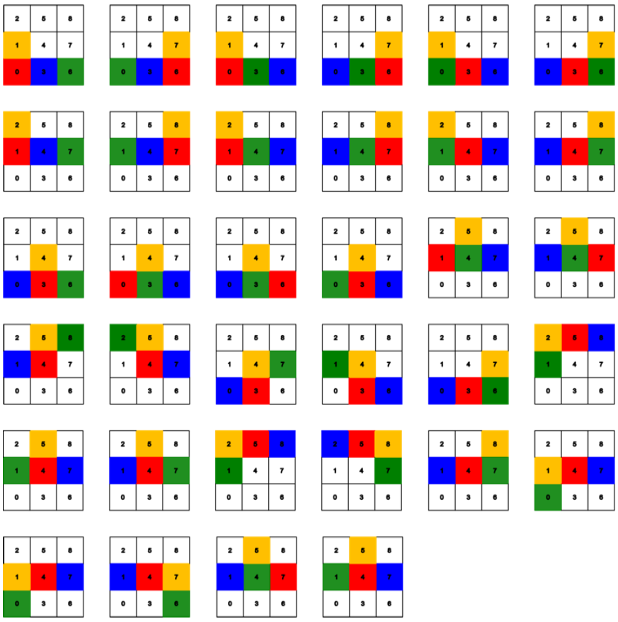

Figure 2. Results of 28 functional topological relationships generated by the program.

The forms can be roughly divided into "L ", "T" and "Z" type residences. All the results satisfy the pre-defined relation matrix and the lighting/ventilation requirements. From the perspective of the post-disaster temporary housing design, the computational system finds an optimal solution adjusted to the end-user's family conditions or personal living habits, allowing a real-time or addition or subtraction of functional spatial compartments.

C. *Experiments on the Generation of a Community Agreggation*

Based on the experiments of the generation of single housing units, the method allows the simulation of endless combinations for the housing plan. In this paper, we choose three types of housing units "L", "T", and "Z" (Fig. 3).

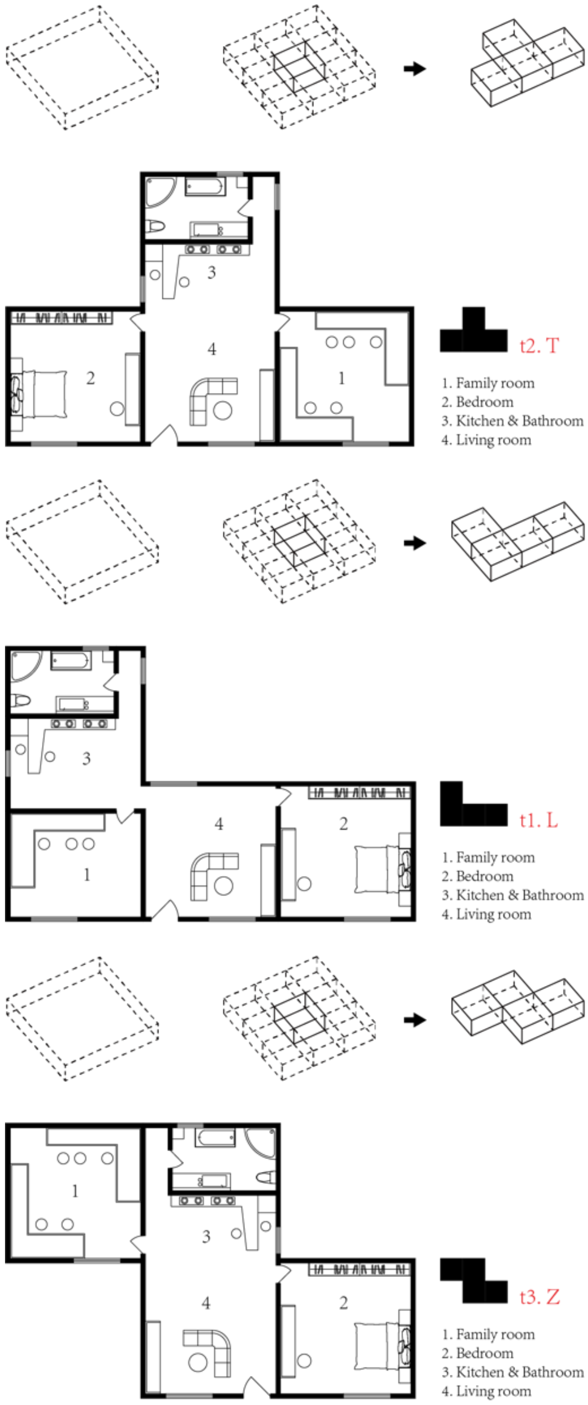

Figure 3. Three forms of functional plans.

Those typologies are combined in different arrangements to create a community (or housing aggregates). The size of this community can also be adjusted by adjusting the number of "spatial cells" in the program according to the real situation of disaster scenario.

A 9 × 9 modular grid was used to simulate the aggregation. Three attributes, “residential”, “service” and “transport” have been clearly identified. The design algorithm produces 28 types of plans generated into a community layout according to the rules of community growth (Fig. 4).

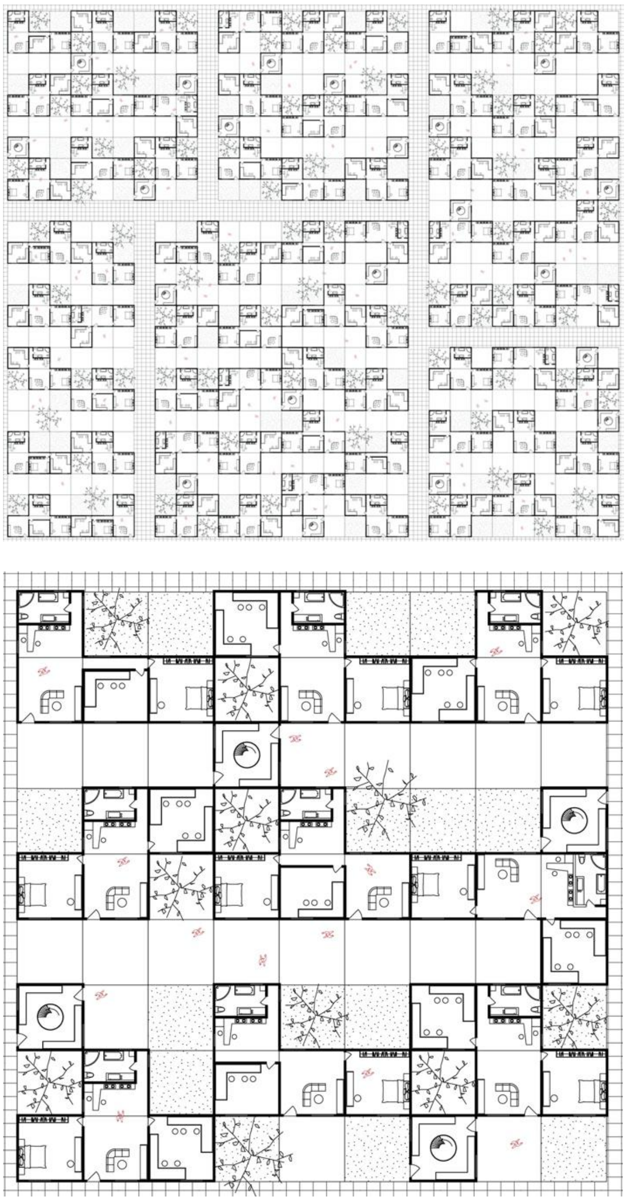

Figure 4. Community plan.

On a wider design scale, the entire community plan is reasonably adjusted to form a sequence of courtyard and shared spaces adjacent to each housing unit, contributing to a certain idea of urbanity.

V. CONCLUSION, FROM COMPUTER TO REAL SOCIAL CONTEXT

The ability of this system to achieve endless combinations by the active intervention of the designer or the users, the resettled families, is contributing to the discussion about crossing social aspects with digital and computational design. The use of these kinds of techniques represents a new way of thinking architecture in terms of computational process, but at the same time shouldn’t forget the main focus of designing for people, in a sensitive scenario of post-disaster relief. Using digital-computational resources we can bind the social variables into a design process, and use the contribution of post-disaster populations to help generating surprising architectural outcomes.

APPENDIX A MODULAR AGGREGATION SCRIPT USING VISUAL BASIC PROGRAMMING LANGUAGE

```
Function Generate()
        Dim intSize, intNumber

        intSize = Rhino.GetReal("Please enter the size of the site plot", 15, 1,
300)
        intNumber = Rhino.getreal("How many iterations should the Residential
Units run?", 20, 2)

        Dim arrField, Field1, Field2
        ReDim arrField(intSize,intSize,intSize)
        Dim i, j, k, FieldDimension, nCount
        Dim gen

        FieldDimension = Ubound(arrfield)

        For i = 0 To FieldDimension
                For j = 0 To FieldDimension
                        For k = 0 To FieldDimension
                                If rnd < 0.5 Then
                                arrField(i,j,k) = 1
                                Else
                                arrField(i,j,k) = 0
                                End If
                        Next
                Next
        Next

        Call init(arrField)
        Field1 = arrField
        Field2 = Field1
        Call DrawField(Field1, FieldDimension)

        For gen = 0 To intNumber
                Call iter(field1, field2, FieldDimension)
                Call iter(field2, field1, FieldDimension)
        Next

End Function
Function CountUnits (x, y, z, arrField, FieldDimension)
        Dim Count, i, j, k
        Count = 0
        If z = 0 Then
                If x = 0 Then
                        If y = 0 Then
                                ' 0 Ebene
                                If arrfield(FieldDimension,
FieldDimension,         z) = 1 Then Count = Count + 1
                                If arrfield(x+1,  FieldDimension, z) = 1
Then Count = Count + 1
                                If arrfield(x, FieldDimension,z)   = 1
Then Count = Count + 1
                                If arrfield(FieldDimension, y+1,  z)= 1
Then Count = Count + 1
                                If arrfield(x+1, y+1, z)= 1 Then Count
= Count + 1
                                If arrfield(x,  y+1,z)  = 1 Then Count =
Count + 1
                                If arrfield(FieldDimension, y, z)= 1
Then Count = Count + 1
                                If arrfield(x+1, y, z)= 1 Then Count =
Count + 1
                                ' 1 Ebene
                                If arrfield(FieldDimension,
FieldDimension,         z+1) = 1 Then Count = Count + 1
                                If arrfield(x+1, FieldDimension, z+1) =
1 Then Count = Count + 1
                                If arrfield(x, FieldDimension, z+1) = 1
Then Count = Count + 1
                                If arrfield(FieldDimension, y+1, z+1)=
1 Then Count = Count + 1
                                If arrfield(x+1, y+1, z+1)= 1 Then
Count = Count + 1
```

```
                              If arrfield(x, y+1, z+1)= 1 Then Count =
                              If arrfield(FieldDimension, y, z+1)= 1
Then Count = Count + 1
                              If arrfield(x+1, y, z+1)= 1 Then Count
= Count + 1
                              If arrfield(x, y, z+1)= 1 Then Count =
Count + 1
                              '-1 Ebene
                              If arrfield(FieldDimension,
FieldDimension,         FieldDimension) = 1 Then Count = Count +

                        End If
                End If
        End If
        CountUnits = count
End Function
Function Connect(state,N)
        If State = 1 Then
                If N < 4 Then
                        Connect = 0
                        Exit Function
                ElseIf N > 6 Then
                        Connect = 0
                        Exit Function
                Else
                        Connect = 1
                        Exit Function
                End If
        Else
                If N = 2 Then
                        Connect = 1
                        Exit Function
                End If
        End If
End Function

        Call Rhino.EnableRedraw(True)
End Function

Function init( arrField)
        Dim arrobs, pt, i, strOb
        arrobs = rhino.allObjects

        If isarray(arrobs) Then
                For Each strOb In arrobs
                        pt = Rhino.PointCoordinates(strOb)
                        arrField(CInt(pt(0)),CInt(pt(1))) = 1
                Next
        End If
End Function
Function iter( Field1, Field2, FieldDimension)
        Dim i,j,k, liv, N
        For i = 0 To FieldDimension
                For j = 0 To FieldDimension
                        For k = 0 To FieldDimension
                                N =
CountUnits(i,j,k,Field1,FieldDimension)
                                liv = Connect(Field1(i,j,k),N)
                                If liv = 1 Then
                                Field2(i,j,k) = 1
                                Else
                                Field2(i,j,k) = 0
                                End If
                        Next
                Next
        Next
        Call DrawField(Field2, FieldDimension)

End Function
```

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Filipe Afonso conducted the research process: Jun Lu built the computational models and the architectural layouts. Both authors wrote different parts of the paper and approved the final version.

ACKNOWLEDGMENT

The authors wish to thank the Guangzhou University, School of Architecture, the institution where this research was conducted.

REFERENCES

[1] Y. Peng, *Architecture Space Combination Theory*, Beijing: China Building Industry Press, 1998.

[2] J. Bao, *Supporting Housing*, Jiangsu: Jiangsu Science and Technology Press, 1988.

[3] S. Tong, *Modular Principle Design Method and Application*, Beijing: China Standard Press, 2000.

[4] B. Qiu, *Case Analysis of Post-earthquake Reconstruction*, Beijing: China Building Industry Press, 2008.

[5] G. Guan, "From concept to construction: Architecture in China's modern countryside," *Journal of Architecture*, vol. 12, pp. 29-35, 2018.

[6] W. Peng, "Talking about the design of post-disaster shelter", *Chinese and Foreign Architecture*, vol. 2, pp. 73-74, 2013.

[7] A. Lindenmayer, "Mathematical models for cellular interactions in development," *Journal of Theoretical Biology*, vol. 1, p. 18, 1968.

[8] G. W. Flake, *The Computational Beauty of Nature: Computer Explorations of Fractals, Chaos, Complex Systems, and Adaptation*, Massachusetts: MIT Press, 1998.

[9] J. Maeda, *Design by Numbers*, Massachusetts: MIT Press, 2001.

[10] G. Stiny, *Shape: Talking about Seeing and Doing*, Massachusetts: MIT Press, 2006.

[11] F. Afonso and V. Magalhães, "Peripheral State of Mind," in *Arquitectos Anonimos*, C. Sant'Ana and M. Leiria, Ed. Lisbon: Caleidoscopio, 2012.

[12] P. Zhang et al, "Realization and application of adaptive ant colony algorithm for intelligent evacuation path optimization of large public buildings," *Journal of Shenyang Jianzhu University*, vol. 6, pp. 1055-1059, 2008.

[13] B. Li, *Building Generation Design: Research on Computer Generation Method of Building Design Based on Complex System*, Nanjing: Southeast University Press, 2012.

Filipe G. Afonso (Porto, 1975) holds a Master Degree in Digital Architecture from Universitat Internacional de Catalunya, Spain, in 2009.

He has been developing a methodology for his architectural practice and research that cross the areas of advanced computation, fabrication, arts and drawing. He is a Professor at the Guangzhou University. In 2001 he co-founded Arquitectos Anonimos®, an award-winning architecture studio with built work recognized internationally. He has been an invited critic, guest lecturer, and keynote speaker at various institutions in Portugal, Austria, Australia, Netherland, Bulgaria, Slovenia, United States, Singapore, South Korea and China. His current research focus on computational design and materiality in the field of architecture.

Jun C. Lu (Wuhan, 1994) holds a Master Degree in Architecture and Urban Planning from Guangzhou University, China, in 2020.

He has been focusing his studies and research on computational design and digital technologies for architecture since 2017.

He was awarded with several prizes in design competitions in China and abroad, like the "Mind de Cork International Competition - 2018" and the "Tongji University Upro Competition – 2018".