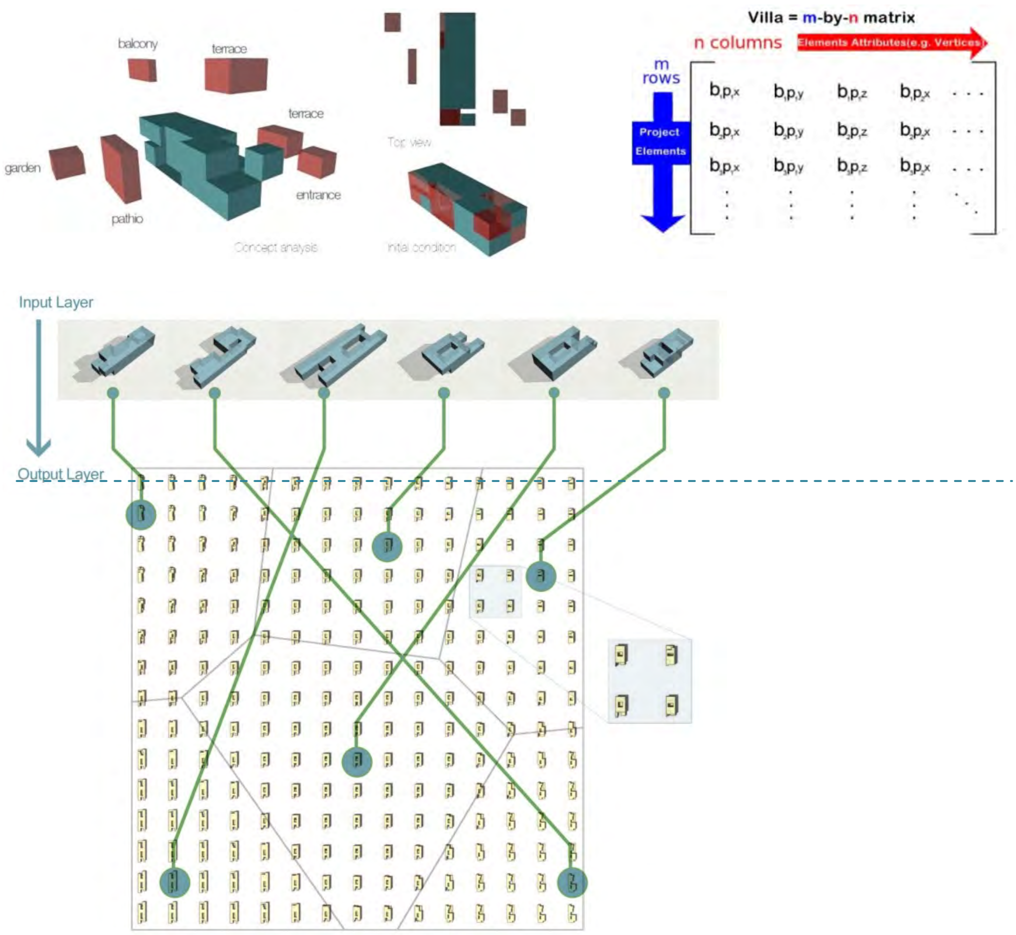
Mohamed Zaghloul

Machine-Learning aided architectural design
Self-Organizing map generate in between design alternatives

Abstract:

The study is a part of an on-going research that focusing on using Self-Organizing Map – SOM – as an unsupervised learning algorithm in order to classify and cluster Design-data inputs and integrating them with the design process. A Villa experiment is presented to target using SOM in the form finding phase through a non-linear morph of geometrical elements and spatial solution varieties. The Villa Design-data has multiple attributes that had been encoded as Matrices. By applying SOM algorithm on 6 initial different design alternatives Matrices, The output is a 2d topological map shows the “close” and “far” of the 6 input variations that in center of 6 Voronoi cells. Moreover, by rendering the weights of SOM neurons which are in-between these initial inputs, that creates a non-linear morphing in between the 3d models. Finally, the paper shows creating a design system – by using SOM – that captures, stores, analyzes, clusters and presents a non-linear morph in between many 3d models at once according to their distances.



SOM input layer: 6 design alternatives of villas. Output layer: a non-linear morphing in between 6 design alternatives

Topic: Architecture Design

Authors:

Mohamed Zaghloul
ETH Zurich, Department of Architecture, Chair for Computer Aided Architectural Design,

Switzerland

www.caad.arch.ethz.ch

Main References:

[1] L. Hovestadt, “*Eigen Architecture*”, Ambra Verlag MMag, Switzerland, 2013

[2] T. Kohonen, “*Self-organizing maps*”, Springer-Verlag Berlin Heidelberg, 2001

zaghloul@arch.ethz.ch

Keywords: Self-Organizing map - SOM - Machine learning Algorithm - non-linear morphing

Machine-Learning aided architectural design

Self-Organizing map generates in-between design alternatives

Mohamed Zaghloul

ETH Zurich, Department of Architecture, CAAD

Chair for Computer Aided Architectural Design

e-mail: zaghloul@arch.ethz.ch

Abstract

– The study is a part of an on-going research that focusing on using Self-Organizing Map – SOM – as an unsupervised learning algorithm in order to classify and cluster architectural design-data inputs and integrating them with the design process. A Villa experiment is presented to target using SOM as a form finding through a non-linear morph of geometrical elements and spatial solution varieties. The Villa design-data has multiple attributes that were encoded as Matrices. By applying SOM algorithm on 6 initial different design alternatives' matrices, the output is a 2d topological map shows the "close" and "far" of the 6 input variations that in the center of Voronoi cells. Moreover, rendering the weights of SOM neurons in-between these initial inputs create a non-linear morphing in between 3d models. Finally, the paper shows creating a design system – by using SOM – that captures, stores, analyzes, clusters and presents a non-linear morph in-between many/any 3d models at once.

– **Keywords:** Self-Organizing Map – SOM – Machine learning – Non-linear Morphing – Architectural Design Process – Unsupervised Learning Algorithm

Introduction: Neural Networks vs. conventional computing

– A dominant mode of using computers in architecture is as merely machines to save time from doing an overwork; notwithstanding that the beauty of computers is that they are not machines; they are abstract machines [1] which enable formulating general concepts by abstracting common properties of instances.

– An argumentation since 1950 about using computers: How are the computers talking to us / how should we talk to computers / can we teach them to interplay with us in the data processing – In brief "*Could machines think*?!"[8] – I will say, "Yes, machines can think as much as they learn." And a clue for that is discussed in this study from architectural point of view. By integrating machine learning algorithm with the design process, e.g. SOM, a process of discovering patterns inside data is easily attained, and that will push the limits of developing the start-up steps of the design process.

– Neural networks – NN – use a different approach to problem solving than conventional computers. NN and conventional algorithmic computers are not in competition but complement each other [6]. The conventional computers use specific steps – Algorithm – to solve a problem and without these steps the problem cannot be solved. On the other side, Neural Networks can learn by feeding them with examples.

– A neural net consists of any number of processing elements called neurons or nodes. Each neuron is connected to other neurons each with an associated weight. Neural nets can be applied to a wide variety of problems such as classifying patterns, performing general mappings from Input patterns to output patterns and grouping similar patterns. [9] NN uses a family of machine learning algorithms, which inspired by – used to model – the biological nervous systems. [5] Among NN models, SOM is commonly used unsupervised learning algorithm.

– The regular morph techniques are between two entities in a linear way, and that uses diverse of mathematical and representation levels for these linear morphing that used conventional morphing algorithms [3]. A different kind of linear morphing technique had been presented in Eigenchair project [2] which used Principal Component Analysis – PCA – algorithm. By integrating NN with the 3d modelling as presented in this paper via using SOM algorithm, a non-linear morphing technique is emerged in-between many and any at the same time.

1. Machine Learning aided architectural design

– “Machine-Learning aided architectural design” is suggested in order to integrate learning from data that produces discovering patterns, understand and manipulate the data entities in a holistic way, with architectural design. The proposed design process goes parallel with an unsupervised learning algorithm – SOM – which will not tell how to end-up solving a problem directly but will tell how to begin preconceiving data and discovering its heuristic rules. It shows you complex data sharp borders and boundaries were morphing anything is possible, and anything may be related to any.

1.1 Learning from Data vs. Design by Data – Theoretical Approach

– By being surrounded by a massive amount of information, integrating a classifying and clustering analysis requires a balanced intelligent environment that able to learn from the information. If design elements of objects are abstracted and coded as multidimensional vectors, they become more effective and manipulative. SOM is suggested to integrate with the architectural design process explicitly with any of design steps. It’s a new way to preconceive our data and dive into the hidden similarities and to discover patterns inside the used data. That data can express geometrical elements - physical attributes – building performances and any.

– SOM is one of the unsupervised learning algorithms that no labels are given to the learning algorithm, leaving it on its own to find structure in its input. It can discover hidden patterns in data [4]. SOM scientifically is used to classify and cluster data, but this paper focuses and pushes the limits of linear morphing to a non-linear morphing in between 3d models by rendering the SOM neurons’ weights.

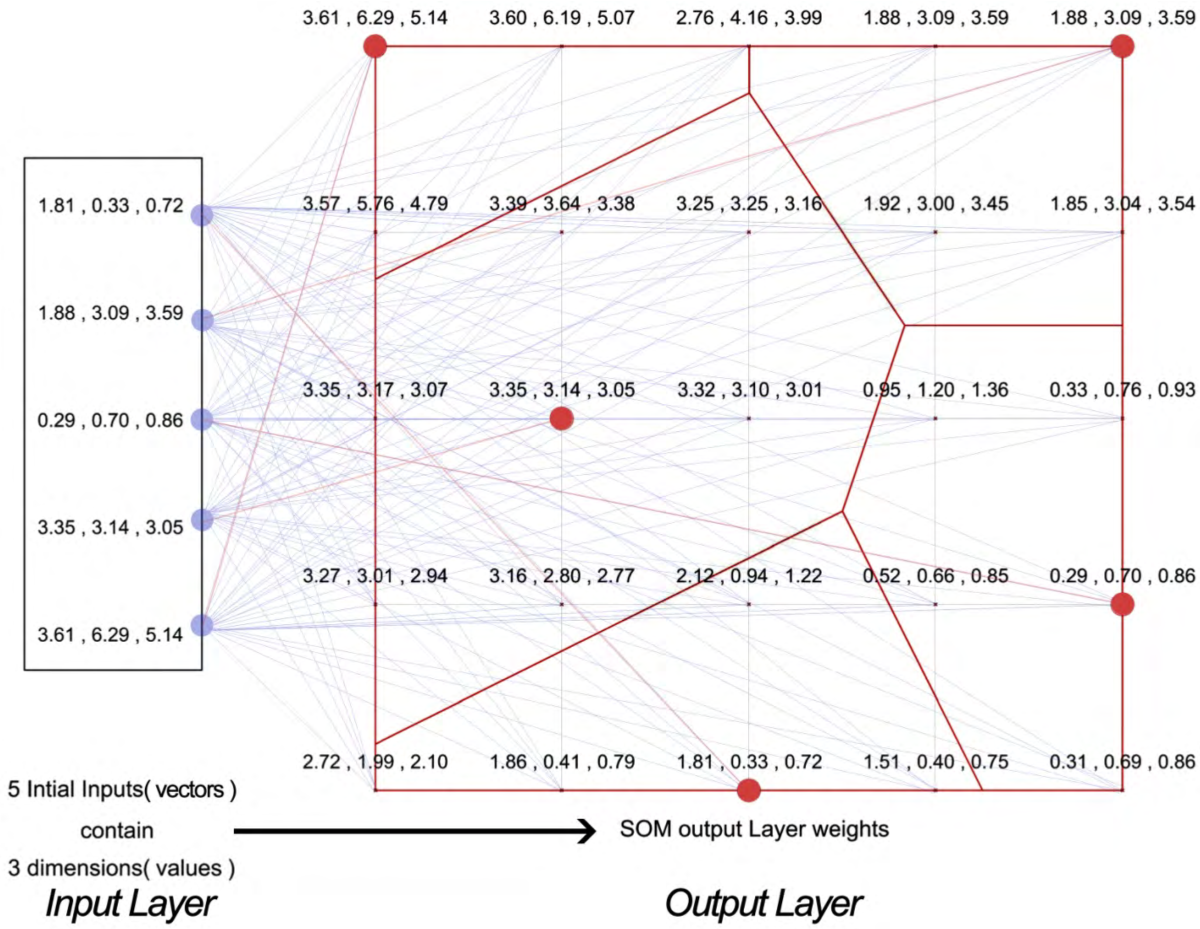
– A lot of data are used within the design process at different levels such as project elements areas, relations, geometrical data, weather data, building performance and Energy consumption…etc. This paper isn’t talking about data optimization, but about understanding and preconceiving data patterns by SOM because of its ability to reduce the data dimensionality and express it in our limited Cartesian space.

– 1.2 Dimensionality Reduction by SOM – Technical Approach

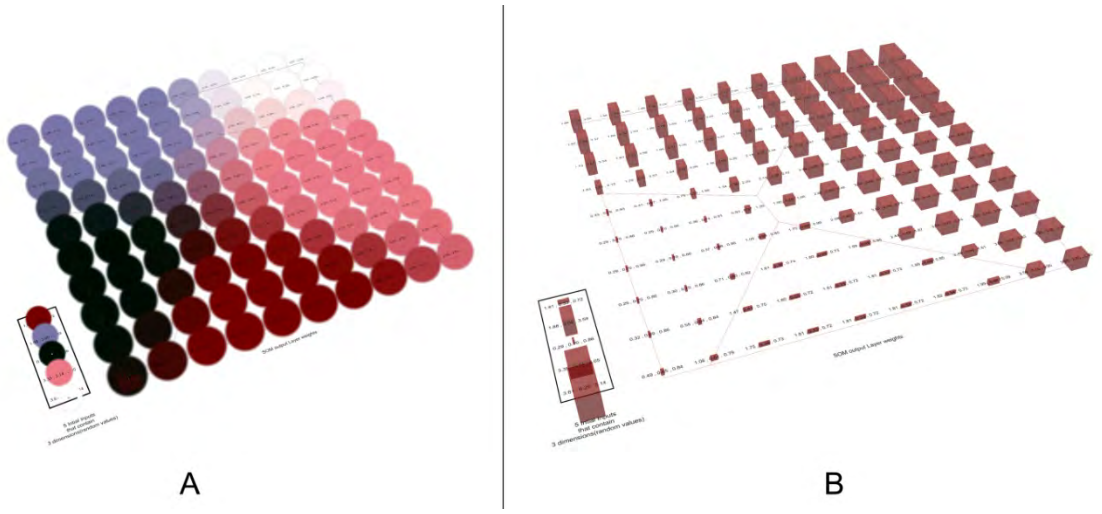
– We use Cartesian logic of modelling in order to formulate our ideas, which limited especially with multi-dimensional models. SOM enables reducing those multi-dimensional values and represents it in lower dimensional spaces (1D, 2D and 3D). Consequently, we are able to extract the hidden similarities in between these data, and visualise the degree of belonging in between the data elements via clustering.

– A detailed algorithm example – Villa experiment – will be described later in section 2.2. This section shows an example – Figure1 – of how simply SOM represents graphically the relations between, on one side, the input layer that contains initial inputs. The inputs are 3 dimensional vectors that can represent any RGB colours or box dimensions (Figure 2).

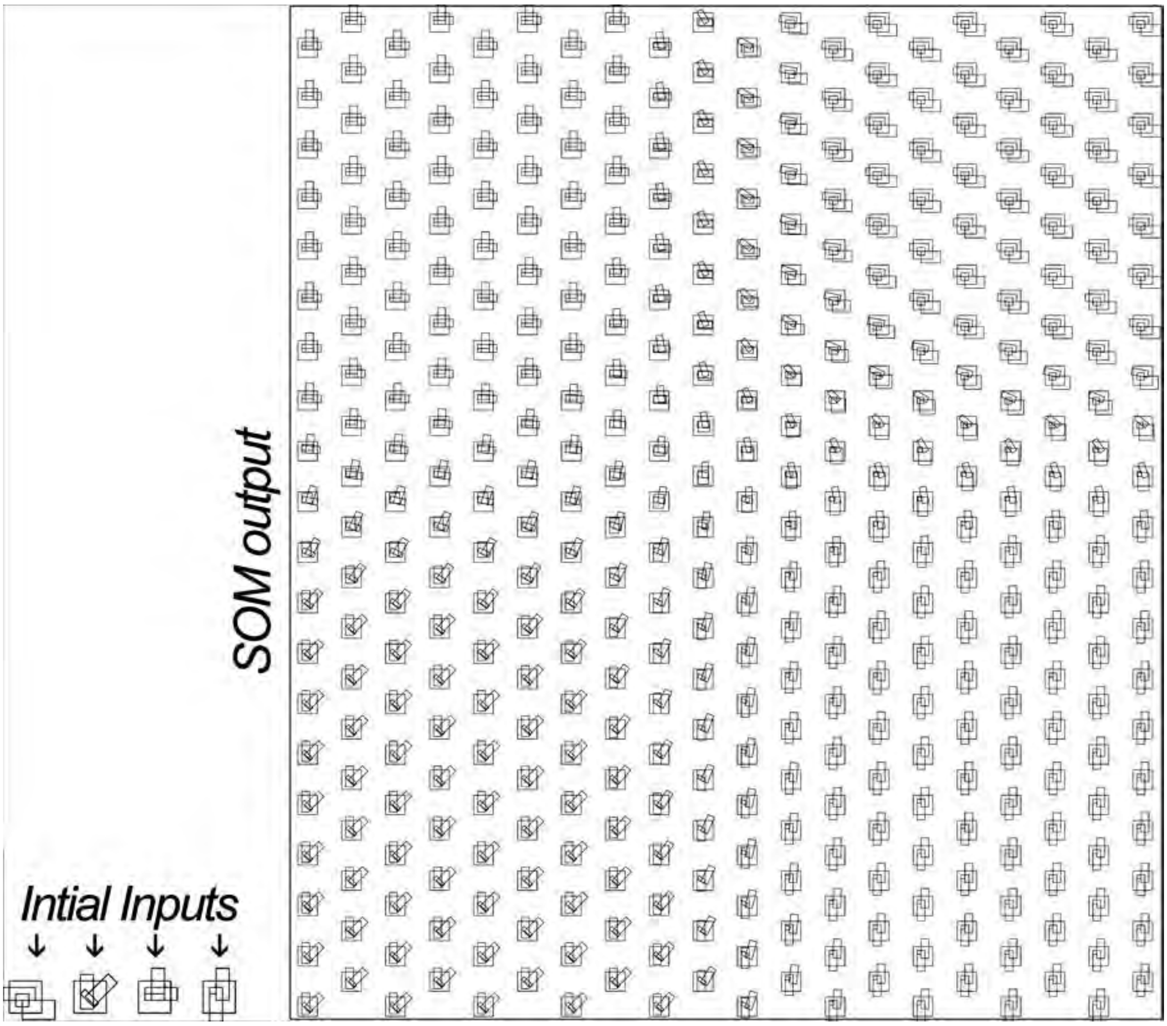
– On the other side, the output layer is a topological grid – rectangular or hexagonal or any type of grids – of neurons/nodes, which projects 3-dimensional data in a 2-dimensional representation. The similarities between inputs can be recognized according to the close/far distances between them. Moreover, by generate in-between the inputs, heuristic non-linear morphing paths between the inputs are emerged.



– Figure 1: Shows the 2D topological output map of SOM that represents the distances between 3D Input vectors.



– *Figure 2: adding more neurons for the output layer and visualising the inputs' values (random) A) as RGB colours of dots – B) as dimensions of boxes, SOM returns a parametric mapping of the in-between input values.*

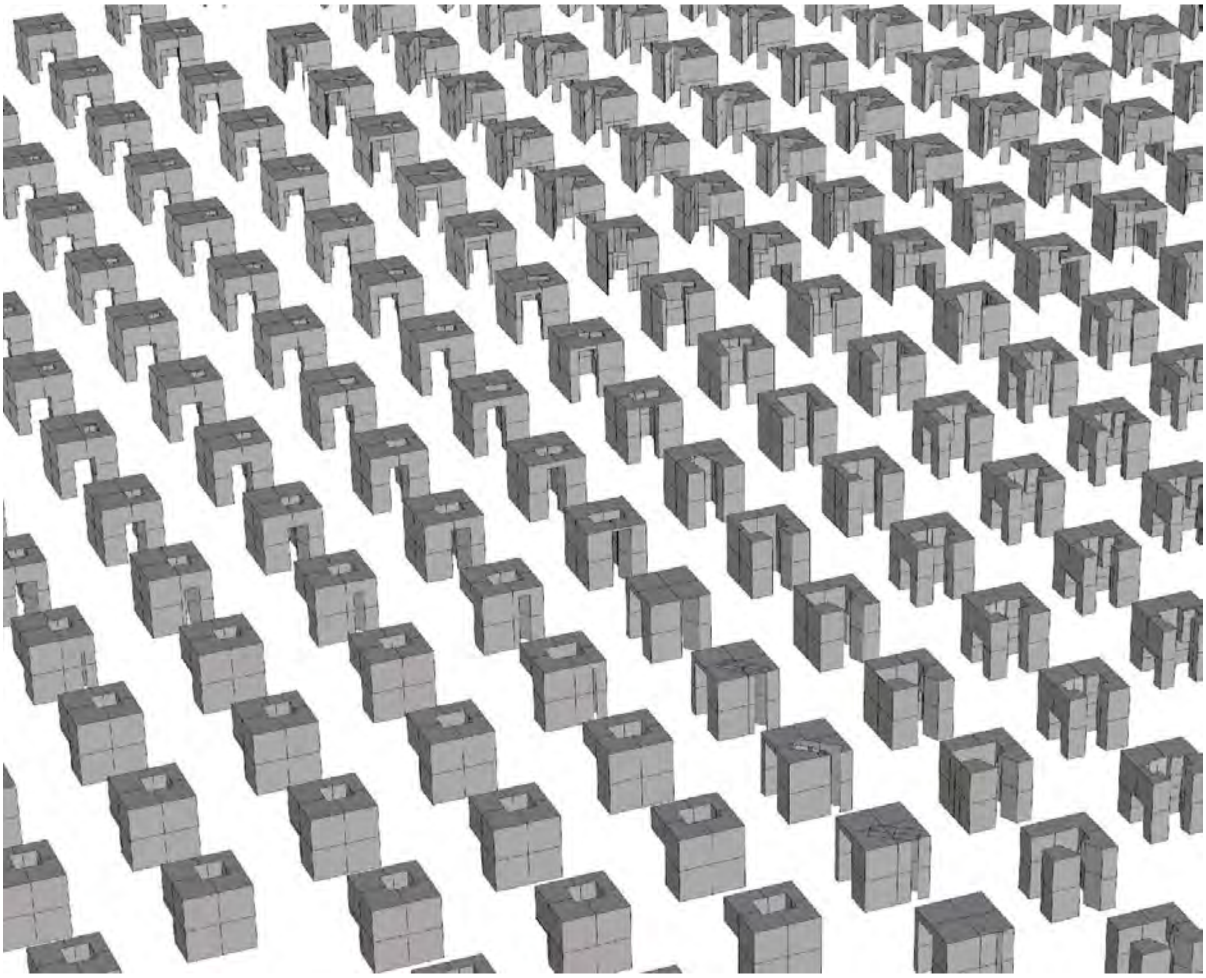


– *Figure 3: SOM defines automatically the behaviour of rotation and scale of the 3d initial objects (Boxes) without pre-parameterization process.*

– In Figure 3, four initial inputs, each consists of three boxes that rotated and scaled inside a bigger cube. The inputs were coded as a sequence of vertices of each box, so each input vector has 72 values. Hierarchal relations are stored between the values that transform the values later to vertices, faces and solids. After training SOM, which learned from the inputs, it returns out morphing in-between the inputs' values.

– Then, the models are subtracted from a box – Figure 4. Imagine that these subtracted boxes express entrances and courts that changing its dimensions and rotations according to the domain of the inputs. Thus eventually, SOM starts with multidimensional vectors that represent entities in an un-parameterize encoding, then it sorts the indices around the input data to produce a nonparametric mapping.

–



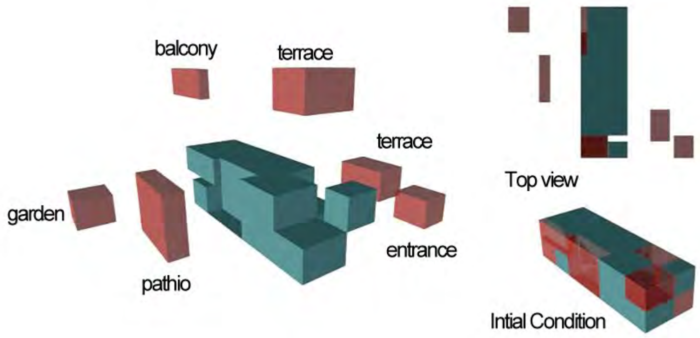
– *Figure 4: representing the subtraction of 3 Boxes (entrances – courts) from a main cube that can express simply a building.*

–

– 2. Villas experiment

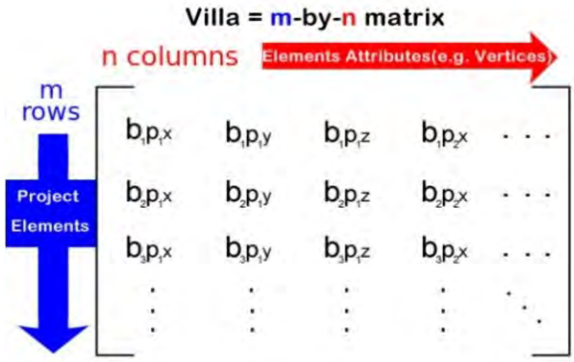
2.1 Encoding Matrices

– Most of the design-data have manifold attributes. These attributes can be articulated as dimensions of multiple vectors. If there is a data set of element X, $X=\{x_1,...x_n\}$ as a set of values that describe the n-dimensional space of the vector X. For instance, a program element – e.g. reception – will be articulated as a vector with multiple attributes. The Reception has the following properties {40 m2, facing a good view, away from direct sun-rays and away from private area, and direct relation with entrance}. This can be written as follows: Reception= {40,1,0,0,1} the measuring units of this vector attributes – dimensions – are different from each other area is in m2 and the other attributes is a binary or rational number between 0 and 1. Another example, a box has eight vertices and each of these point vertices has three values $p_n=\{x_n,y_n,z_n\}$.So we can describe it as a Box $b=\{x_{p1},y_{p1},z_{p1}, x_{p2},y_{p2},z_{p2},\ldots\ldots x_{p8},y_{p8},z_{p8}\}$.



–

– *Figure 5: shows design elements which based on addition or subtraction of the whole building volume according to the design objectives.*



–

– *Figure 6: shows an architectural project as matrix of m elements (vectors) by n attributes (dimensions).*

– If we have many boxes in the Space R – R is denoting a villa design in this Figure (1), then we can describe them as entities of this space $Bn \in R$. While $B1$=Entrance, $B2$= Court, $B3$= Terrace...etc. Some of these Boxes are added, and others are subtracted from the whole volume according to the design objectives.

– A parallel process to encoding the matrix is encoding the hierarchical relations between its values. These hierarchical relations will be used again at the end of the decoding process. Example of those hierarchal relations between the matrix values - Figure (2): each Box b contains face f, and each face contains points that have values x, y and z.

– $b_n=\{f_1\{p_1\{x,y,z\},\ p_2\{x,y,z\},\ p_3\{x,y,z\},\ p_4\{x,y,z\}\},\ f_2\{\ldots\},f_3\{\ldots\},\ldots.f_6\{\ldots\}\}$.

– The last step is defining different design alternatives of villas V.
$Vn=\{b_1\{f_1\{p_1\{x,y,z\},\ldots p_4\{x,y,z\}\},\ldots,f_4\{p_1\{x,y,z\},\ldots p_4\{x,y,z\}\}\},b_2\{\ldots\}\ldots\ldots b_n\{\}\}$.

2.2 Decoding Matrices by SOM

– Here we describe the main steps of the classical SOM algorithm that were applied to 6 different alternatives of designing villas. Each prototype has different positions of zones (entrances-courts-terraces-reception areas…etc.) and each of them added or subtracted later from the whole volume. The equations and algorithmic steps of SOM that were used are referenced by T.Khonen [5], The main steps of this experiment are as follows:

– 1) Normalizing the vectors: If all the attributes are different in measuring units, then the normalization step is important to put all of them within the same domain values according to the weight of each attribute to the whole vector [7]. Nevertheless, in this experiment, all the values have the same units that related to positions of vertices $\{x, y, z\}$, So this step can be skipped in this experiment.

– 2) Specify the Out-layer grid number and Initial random weight for each of the Output layer nodes. The weight vectors have the same dimensionality as any of the input vectors dimensions.

- e.g each zone has eight vertices and each vertex has three values {x,y,z}, That gives 24 values. Here in this experiment, each villa has 10 zones, thus each one is described by 240 values.

- It's important to note that the same sequences of zones for each prototype are the same.

- A rectangular grid with 15 column x 15 rows is used as an output layer for this experiment.

– 3) Compute the Euclidean distance Matrix between out-layer grid nodes – neurons – . That will be used later in updating the weights within the iterations gradually according to a neighbo urhood function. *[5]*

$$DistFromInput^2=\sum_{i=0}^{i=n}(I_i-W_i)^2$$ *The Euclidian distance equation [5]*

– 4) Training SOM – For each Iteration of a time step (t):

- Select randomly any of input vectors I

- Calculating the Best Matching Unit BMU: by iterating the Euclidean distance calculation of all the nodes between each node's weight vector and any input vector. The node with a weight vector closest to the input vector is tagged as the BMU or the winning neuron. This equation is for iterating the Euclidean distance, I is the current input vector, and W is the node's weight vector.

- Updating the weights: $W(t+1)=W(t)+\Theta(t)L(t)(I(t)-W(t))$ .the Greek letter theta Θ, represent the amount of influence a node's distance from the BMU has on its learning. L is for the learning rate. For further details about Θ and L is described in details by T. Kohonen [5].

– 5) The output layer weights have 240 sequential values for each node. Three values are a Cartesian position of point {x, y, z} then rendering each eight vertices as a box that be added or subtracted according to the previous hierarchal encoding of the input layer elements.

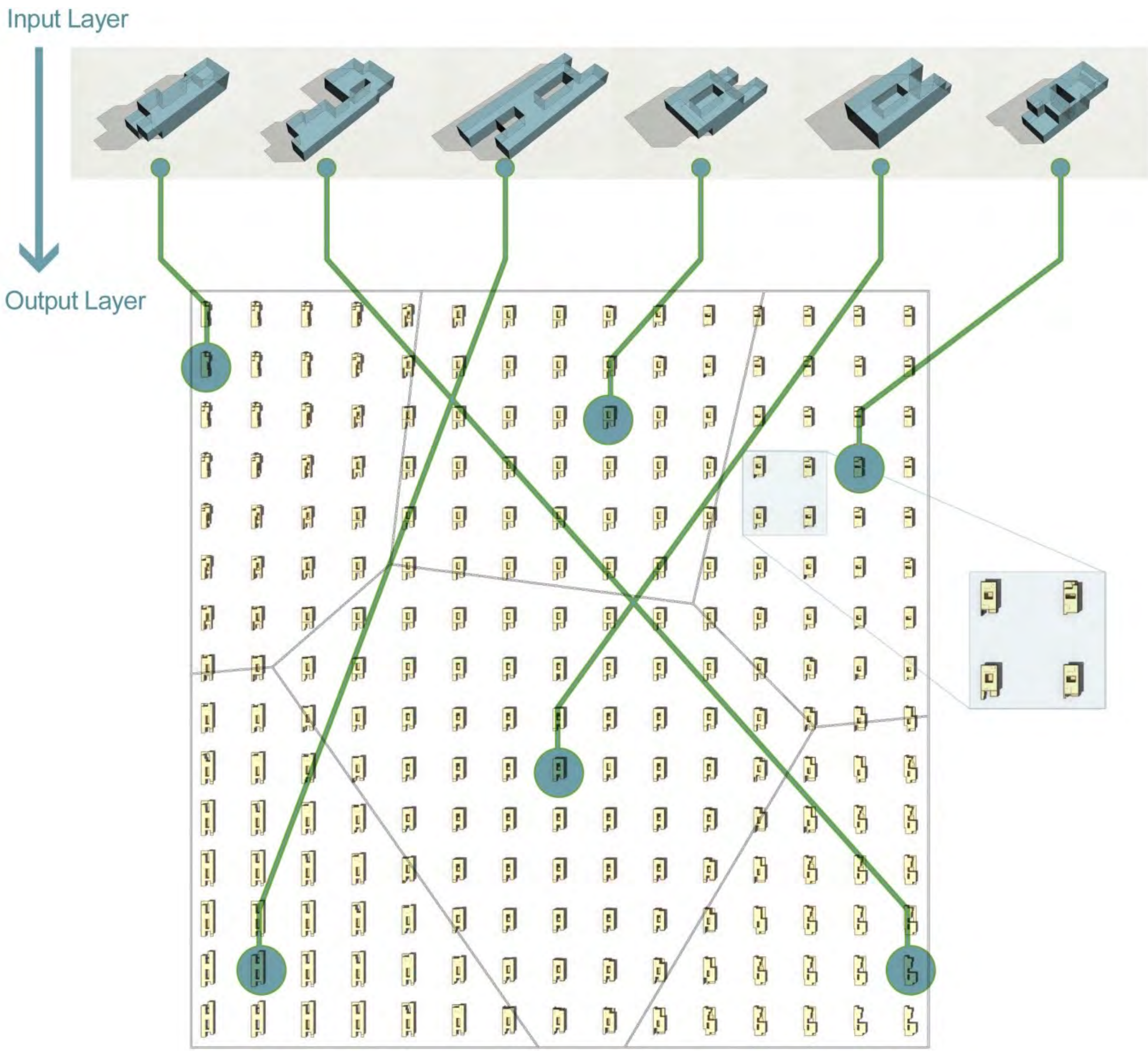


Figure 7: shows SOM input layer: 6 design alternatives of villas. Output layer: a non-linear morphing in between 6 design alternatives.

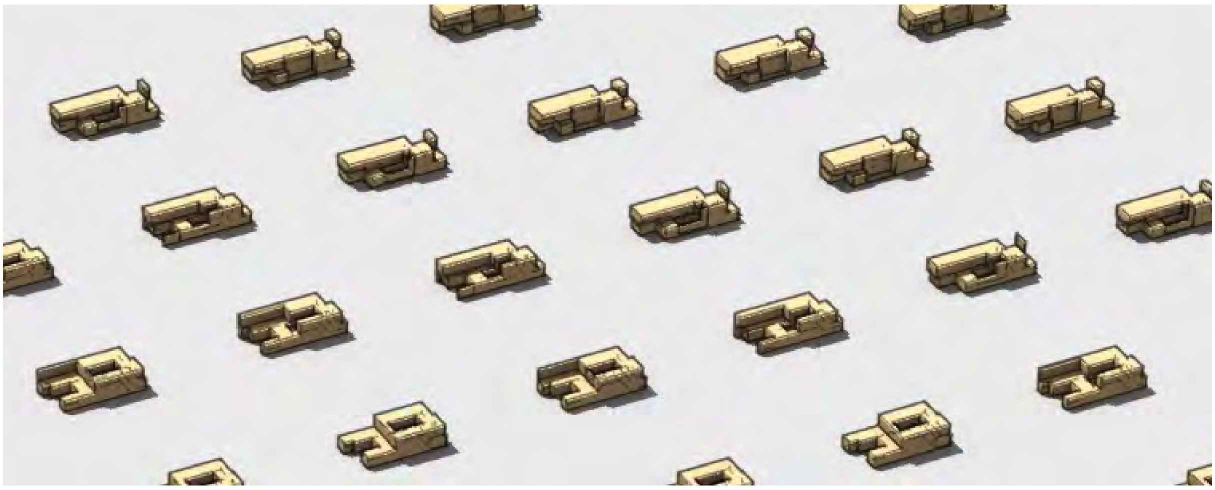


Figure 8: preview for some models of the Output layer: a non-linear morphing in between the 3dmodels.

2.3 Design concept refining - development

– A cyclical process can be established by the designer who decides a trip from any object in this topological map to another passing with the in-between models in a non-linear way. Then, the designer can select from these in-between models to start a new stage of data morphing. It's possible to choose new designs – as shown in Figure (8) – between the emerged outputs to train SOM with the new designs' data. This process can run in parallel to basic building performances' simulations, e.g. shadings – solar radiation – heating/cooling energy consumption for the buildings' envelopes…etc.

3. Application interface

– The SOM code was implemented using Rhino, Grasshopper GH and Mathematica via SOM tool – developed by the author – similar to Mantis.[10] On one side Rhino and Grasshopper were chosen for dealing with constructing and deconstructing the 3d-Modeling elements in an intuitive parametric manipulation. On the other side, Mathematica was chosen for its symbolic computational language that enables dealing with a huge amount of data inputs in fast time and processing intuitively. The Main SOM code was written in Mathematica then be called via an add-on SOM tool in GH.

4. Conclusions

– Finally, the paper shows creating a design system that captures, stores, analyzes, manages, clusters and presents in between the geometrical data that are linked to the designs' alternatives. SOM is integrated with the tool to compute the changes in building geometry and visualizing the hidden relationship between designs in a non-linear analysis method that produce a new non-linear morphing technique.

– SOM can affect strategic decisions in the early conceptual design stages that lead to new optimal alternatives. It also means that ideas can generate alternatives then be pursued, tested and accepted/rejected at early stages of the design process.

– 5. Acknowledgment

– I would like to thank my supervisor Prof. Dr. Ludger Hovestadt, head of the CAAD chair at the ETH Zurich for support and mentoring my PhD research, and thanks to Federal Commission for Scholarships (FCS) for their financial support. In Addition, I would like to thank ENCODE studio teamwork that participated in the villas' proposals that presented to SODIC construction company in Egypt.

6. References

– [1] L. Hovestadt, V. Buhlmann (2013). *Eigen Architecture*. Publisher: Ambra Verlag MMag. Franz Schaffer

– [2] M.Roman (2013). *Four Chairs and All the Others - Eigenchair*. ecaade conference 31 pp 405-414

– [3] J.Gomes. (1997) *Warping and Morphing of Graphical Objects.* SIGGRAPH. http://www.cs.rit.edu/usr/local/pub/ncs/ImageBased/WarpMorph.pdf

– [4] S. Russell, P. Norvig. (2003) *Artificial Intelligence: A Modern Approach* (2nded.). Prentice Hall

– [5] T. Kohonen (2001). *Self-organizing maps*. Springer-Verlag Berlin Heidelberg.

– [6] C. Stergiou and D. Siganos (96 journal- vol4). *Neural Networks*. http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html

– [7] P.Dematines, F. Balayo (1992). *Kohonen's SelfOrganizing maps: Is the normalization necessary?* Complex Systems,6(2):105-123

– [8] K.Devlin (1998). *Goodbye, Descartes: The End of Logic and the Search for a New Cosmology of the Mind*. Wiley: Pages142-162

– [9]L.Fausett (1994). *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*. Pearson Press.

– [10]http://www.grasshopper3d.com/group/mantis