Self-Learning Algorithm as a Tool to Perform Adaptive Behaviour in Unpredictable Changing Environments – A Case Study

Elite Sher, Angelos Chronis, Ruairi Glynn

University College London, Gower Street,
London, UK, WC1E 6BT
elite.sher.11@ucl.ac.uk, achronis@fosterandpartners.com, r@ruairiglynn.co.uk

Keywords: Adaptive architecture, self-learning, Genetic Algorithms, Artificial Neural Network, Structural Optimisation

Abstract

Adaptive architecture is expected to improve buildings' performance and create more efficient building systems. One of the major research areas under this scope is the adaptive behaviour of structural elements according to load distribution. In order to achieve this, current studies develop structures that adapt either by following a database of pre-calculated solutions, or by using massive computation resources for real-time calculations.

This study aims to achieve an adaptive behaviour in real time, affected by load distribution, by implementing learning abilities on a case study. This is done by applying a learning algorithm—Artificial Neural Network (ANN)—on a physical prototype. The ANN was trained by an optimised database of finite solutions, which was created by a Genetic Algorithm (GA). Through this method, complex calculations are conducted 'off-line' and the component operates in a 'decision-making' mode in real-time, adapting to a versatile environment while using minimal computational resources.

Results show that the case study successfully exhibited self-learning, and acquired the ability to adapt to unpredictable changing forces. This method can be applied over different structural elements (façade elements, canopies, structural components, etc.) to achieve adaptation to other parameters with an unpredictable pattern such as human behaviour or weather conditions.

1. INTRODUCTION

Adaptive Architecture is an expanding, multi-disciplinary research field, exploring the possibilities of buildings to change and respond with relation to various parameters (Teuffel, 2010). The need of adaptive architecture arose once the advantages of it were recognised–buildings can fit a wider range of uses, improve their performance and extend their life time (Sobek & Teuffel, 2001).

Kinetic structures–building or building elements that can change their location, move or change their shape (Fox & Yeh, 2004)–are a key aspect in adaptive architecture.

Various researchers have examined adaptive kinetic structures in an attempt to improve buildings` performance (Sterk, 2006, Senatore et. al. 2011). Currently, research relies on massive complex calculations in real-time or on a vast database of pre-calculated solutions. In order to overcome these issues, this paper offers an approach for structural adaptation by developing self-learning abilities in order to acquire adaptive behavior. This is done by examining a case study element (a canopy), which is situated in a dynamic environment that changes the element's load distribution. This simulated a structure that has to adapt to changing forces and loads such as wind, rain, snow, earthquakes etc. In order to improve its stability performance, the case study had to adapt to unpredictable changes. The case study element chosen by this paper has the potential of being duplicated and assembled as part of a bigger surface or double skin façade, as well as a single structural component, for instance.

2. BACKGROUND

Load distribution, one of the major fields of adaptive structures, deals with the manipulation of external loads and internal force distributions over a structural element, changing over time (Sobek &Teuffel, 2001). This can be carried out in two ways: either by controlling the adaptation of the structure`s shape, or by adapting its structural components' properties--stiffness or strength for instance (Sterk, 2006). A combination between the two methods is possible as well. When time is critical to fulfill the adaptation task, real-time adaptation can be carried out, by embedding computer methods in the kinetic element (Fox & Yeh, 2004).

Optimisation is one of the qualities of an adaptation process, in order to increase structures' efficiency (Sterk, 2006). This allows the structural element to exploit its properties for improving performance and also allows the object to respond to its surroundings. Two of the most commonly used methods for optimisation are Evolutionary Algorithms and Artificial Neural Networks (ANN) (Hanna et. al, 2010). Evolutionary methods, such as Genetic Algorithms (GA), attempt to achieve optimisation through the evolvement of 'generations' (each generation consisted of calculated solutions to the given problem). However, learning methods such as the ANN, enable improvement over time based on the experience gained, to reach the optimised solution (Flake, 1998).

2.1. Genetic Algorithm (GA)

GA is especially suitable for solving optimisation problems with a very large number of possible solutions. The algorithm is based on a process that mimics evolution through a population of candidates. A 'population' (creating a generation) consists of number of 'members', each with its own properties—'genes'. These members represent a possible solution or configuration for a given problem. The 'members' are ranked according to their performance in relation to the target ('fitness' criteria). Following some algorithms that mimic biological principles of natural selection (breeding, crossover and mutation), the fittest 'members' have a higher probability to reproduce and generate a 'better' generation; i.e., to create a population with a better configuration to the given problem.

The use of GAs for optimisation in the architectural field has only recently been initiated. Architectural design problems are often too complex to be simulated, and the number of parameters shaping an 'architectural' problem can make the optimisation process chaotic.

2.2. Artificial Neural Network (ANN)

The ANN can map large data sets and detect their patterns. By that, it is able to find a solution to very complex problems. The ANN algorithms are inspired by the way biological neural networks work, based on neurons and perceptrons (a type of pattern classification device based on visual perception principles) in order to achieve adaptive behaviour within machines. The ANN uses a database as a reference to learn from (known as the 'Training Set'). This database stores similar 'problems' and their known 'solutions'. By looking for patterns in these similar cases, the ANN trains to get a 'right answer' to a given problem.

The ANN consists of three layers: input (problem), hidden layer ('calculation' layer) and output (solution). All layers are connected to each other by 'neurons' that propagate the data from one layer to another. All the connections are weighted based on their relevance to the desired output, so that connections with a higher score are closer to the desired output. First, a 'problem' from the stored database is processed and the ANN gives it a solution. By comparing the suggested solution the ANN produced to the 'right known solution,' the learning process is conducted. The error (the difference between the right answer and the given one) is then calculated and the weights adjusted accordingly (Mitchell, 1997). This is done iteratively until a satisfactory training state is achieved. During that process the results converge to the right solution. Further to the training phase, there is a 'testing phase' based on a data set as well. In this testing phase, the pattern the algorithm has found is tested and verified in a similar way.

The use of the ANN algorithm is widespread in the engineering fields, mainly for purposes of pattern recognition, structures and materials behaviour prediction (Mitchell, 1997; Kota et. al., 2003).

2.3. Relevant Work

In recent years, studies in the field of structures' optimisation according to external changing parameters are well known, especially in the engineering and aerospace studies (Kota et. al., 2003). In the architectural context, however, researchers mainly focus on two fields: The first

uses optimisation as a design tool–optimising different aspects of the design during the design process. The second uses optimisation in real-time to increase the performance of a building by controlling an adaptive dynamic element. Current studies are developing kinetic elements that will perform adaptive behaviour in real-time. These methods follow two main approaches:

One approach is based on reacting according to a vast database, created in advance by detailed simulations with high accuracy. The simulations analyse the behaviour of the structure under the influence of certain static and dynamic loads, and calculate the optimised counter movements required (Details Magazine, 2012). In this method the structure can respond only to known, pre-calculated conditions. That way, the structure is strongly context-based (as the database is created according to a specific environment). Moreover, versatile and unpredictable parameters such as wind or snow, for instance, are destructive.

The second approach currently researched by Senatore et. al. (2011) is a conceptual study that examines the possibility of designing an adaptive structure to achieve improved performance and increase its structural efficiency. This method performs real time optimisation calculations and activates a large number of actuators to maximize accuracy. The complexity of many changing parameters, together with the actuation needed, requires great computational power.

3. METHODOLOGY

This research explores the potential of using adaptive structure methods, based on self-learning abilities, in order to achieve adaptation to unpredictable changing environments and prevent the canopy from buckling. By this, it aims to achieve a wider range of operation and improve its ability to survive in versatile environments.

A single-segment canopy was selected as the examined structural component. The dynamic adaptation process is carried out by controlling the components' properties: by changing the length of the canopy's columns, the column's buckling limit is changed. The case study is aimed to be as generic as possible and to have the ability to adapt to various conditions regardless of location, orientation or materials.

3.1. Research Design

In order to implement a self-learning method, the ANN algorithm was chosen, as it is an efficient learning method for sensor-based data in a constantly changing environment (Mitchell, 1997). Since the ANN requires a database of inputs and finite solutions with which it is trained, a digital simulation was created (Figure 1).

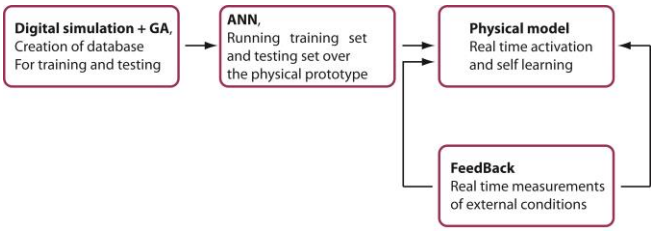

Figure 1. General course of experiment.

First, a digital model of the canopy was created and random forces were applied on it. Aiming to avoid reaching the buckling limit by changing the columns' length, a GA optimisation algorithm was applied on that model, and a database of optimized solutions was constructed accordingly. Later, the ANN algorithm was tested, using the pre-optimised GA database as its initial source of learning.

Lastly, a physical model of the canopy was built and the ANN algorithm was implemented on it. Weight sensors, integrated within its columns, measured the internal loads on each column in order to examine and analyse the acquired adaptive behavior. Linear actuators were embedded in the canopy's columns to enable the change in length of each column. Each 'answer' given by the ANN led to physical adjustments of the prototype. The new configuration was then checked by the weight sensors, in order to ensure the system indeed achieved stability. According to this feedback, the ANN algorithm continued its learning process on the physical model as well as improving its performance. By using this method the optimisation processes was conducted 'off line' (where the complex computation process is carried) and the system operated in real time, in a 'decision making' process, which does not require heavy computing.

3.2. Research Assumptions

In order to examine the feasibility of the method presented above, the system's definitions were based on an abstracted reality and the experiments were conducted under

several simplifications and assumptions (Boris & Srinivasan, 2012):

- The simulation considers the element as isolated. All applied forces and forces acting on the element due to environmental interactions are calculated as a net force: vector sum of all forces.
- Several parameters were defined in a way that allowed tolerance range for fitting the digital simulation to the physical model examined.
- Feedback loops were conducted throughout the process to ensure the goal (defined both to the digital model and the physical one) is achieved.
- Columns are taken as linear and homogeneous.
- Loads on each column are caused by forces applied over the surface. The load on the column is the criteria being examined.
- The critical column load is the buckling limit. The study's buckling threshold is set to a certain percent of the actual buckling limit value, to allow a pre-required tolerance.

The length of a column affects its buckling limit, and therefore the allowed threshold (Figure 2), according to Euler's rule:

$$F = (\pi^2 EI) / (KL)^2$$

where:

F = maximum or critical force (vertical load on column);
E = modulus of elasticity;
I = area moment of inertia;
K = column effective length factor, whose value depends on the conditions of end support of the column; and
L = length of column.

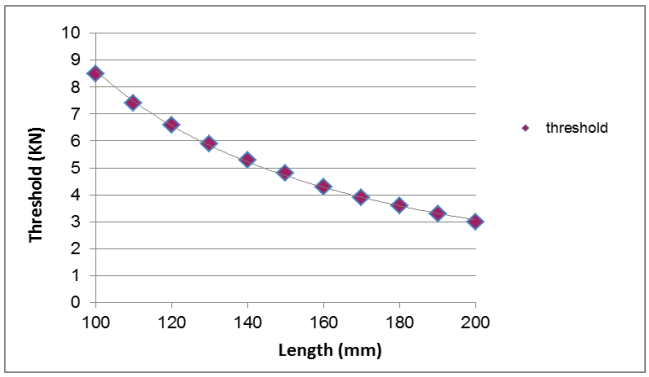

Figure 2. The change of the column's buckling limit in relation to its length.

The success of this test will be measured by measuring the load on each column, and comparing it to the columns' threshold (once adjusted by changing the column's length), so:

Current Threshold – Current Load > 0

Changing the length of one column leads to a change in the angle of the canopy`s surface and therefore a change in the load distribution on each column. There is no single combination of column lengths that avoids the buckling limit of each column; therefore there is no single solution to this problem. By applying a GA, this study got an optimised set of finite solutions.

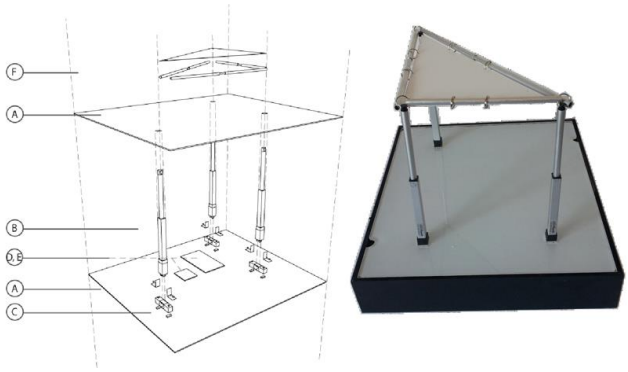

Figure 3. The physical prototype.

3.3. The Physical Prototype

The physical prototype consisted of several components (Figure 3) with inherent elements that can be adjusted and by that change the threshold level. These elements included:

- A wooden 12mm constructed box 400mm*400mm, with upper side made of 6mm Perspex.
- Actuators: 3 Pirgelli linear servos L12-100-100-6-I, with a position feedback, 6V.
- Load cells: 3 micro load cells (0-5 kg) measure the loads applied on each column within the vertical axis only.
- An Arduino board integrated into an electronic circuit operating 3 actuators.
- A PhidgetBridge 4-Input, operating 3 load cells.
- A plastic surface 2mm, attached to telescopic aluminum rods, so it can change its length.

3.4. The Digital Optimisation Process (GA)

Each possible combination of column lengths is called a 'member'. Each 'member' has 3 'genes'—the length of each one of the three columns. Each GA 'generation' is

constituted of 50 members. Each 'generation' picks the members who had the best performance in the previous generation, and uses their 'genes'. Eventually, the GA has 100 generations (as the number of generations increases, the calculated solution is better). In order to overcome the complexity of the problem and to avoid local optima (Hanna et. al. 2007), a 5% degree of randomness was added to the GA when searching for the optimal solution (through the mutation function).

The optimization goals were focused, to avoid chaotic optimisation process (Hanna et. al. 2010). The target was to achieve a stable system, where all columns are stable (load < threshold), while changing the length of each column as minimally as possible and keeping each column as long as possible. This target was set this way for several reasons:

- To avoid generic solutions in which the columns are always in their shortest position, and have the highest threshold value.
- To minimize the energy put into the actuation of the system. (A bigger change requires more power).
- This constraint represents 'Architectural' or 'programmatic' value; and can be further developed.

After setting the new length for each column, feedback loops were conducted to ensure the configuration that was chosen is 'stable' and that the loads on all the columns are indeed below the column's threshold.

In order to create a vast database the simulation ran according to the following method:

- Force applied between the range of 0 to 20N (as this is the maximum load can be carried by the columns of the physical prototype). Each simulation had 1[N] load interval.
- The angle between the surface and the force applied ranges between 0 to 90 degrees (as there is symmetry between the range of 0-90 and 90-180). The simulation ran for every 9 degrees.
- 7 different configurations of force deployments were simulated, each time the combination of the distribution of the force on the columns was changed).

This database included both a training set (1000) and a testing set (400) that were applied on the physical prototype.

3.5. The Learning Algorithm (ANN)

The ANN algorithm was implemented on the physical model using the back propagation method. The canopy reacted to real time inputs (forces applied in real-time) that were detected by the sensors. The input was the current length and the current load for each column (6 parameters in total). If at least one of the columns was above the threshold the ANN produced an output of the new length required for each column in order to avoid the threshold (3 parameters). Since the position of each column affects the other two columns, the output considered the combination of all three columns, rather than the load on a single one.

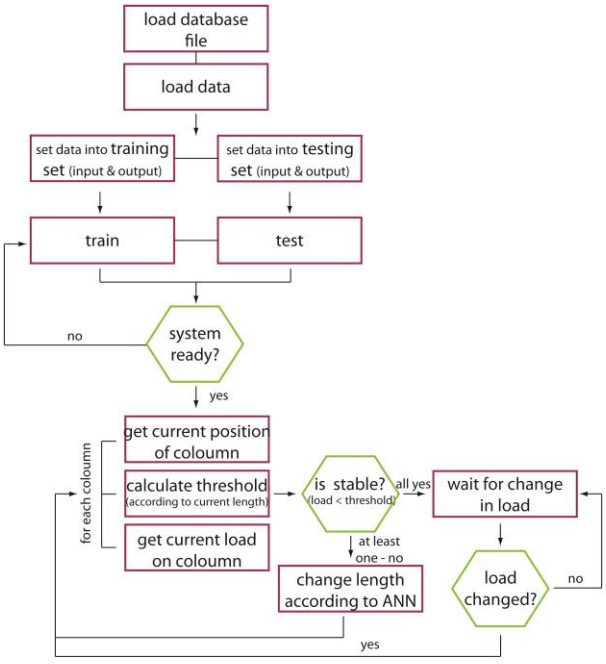

Figure 4. Code flow chart.

3.6. Implementation

All code used in this study, including the communication between Processing and Arduino, Processing and PhidgetBridge, and, the GA and the ANN, have been developed by the author in the Processing programming language.

Once the creation of the database was completed, real-time experiments were conducted in order to calibrate both simulations and model to the same range. The ANN algorithm was embedded inside the final code, including the communication code between the Arduino, Processing and the PhidgetBridge. Both the Arduino and the PhidgetBridge

were used as a feedback tool to get inputs measured in real-time. The Arduino was also used for implementing the results and controlling the physical system. The process consists of the steps as described in Figure 4.

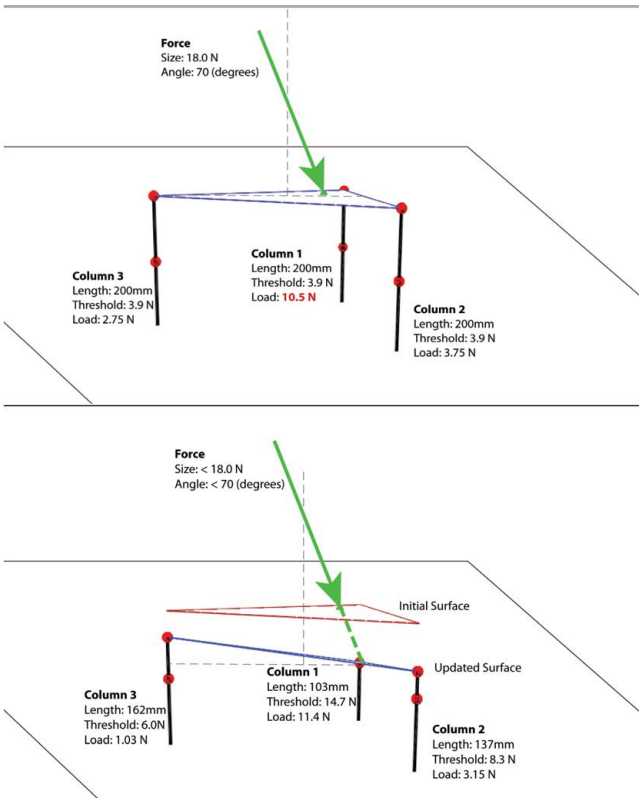

Figure 5. Digital simulation: (Top) Initial configuration – **not stable**. (Bottom) Updated configuration, after optimization – **stable.**

4. RESULTS AND DISCUSSION

4.1. Digital Simulation and the GA

Figure 5 shows an example of a single simulation, and its result. As shown in Figure 5, when the force was first applied over the initialized surface, column 1 was not stable (in 'danger') as its load was supposed to exceed its threshold. By changing the angle of the surface, i.e. by changing the length of the columns, a new setup was created. Within the updated surface position, all columns are stable.

Figure 6 shows the result of 10 (out of 1400) configurations (1 configuration = combination of lengths of the 3 columns), conducted in the simulation and optimised by the GA algorithm. Each chart presents one column. The initial state of the column was the same (length = 200 mm), and random forces were applied over the structure.

As it shows, after optimisation, the load over each column always stays below the updated threshold. The length of the column is changed when at least one of the columns is loaded above its threshold, as can be seen in configurations 1-3 for example.

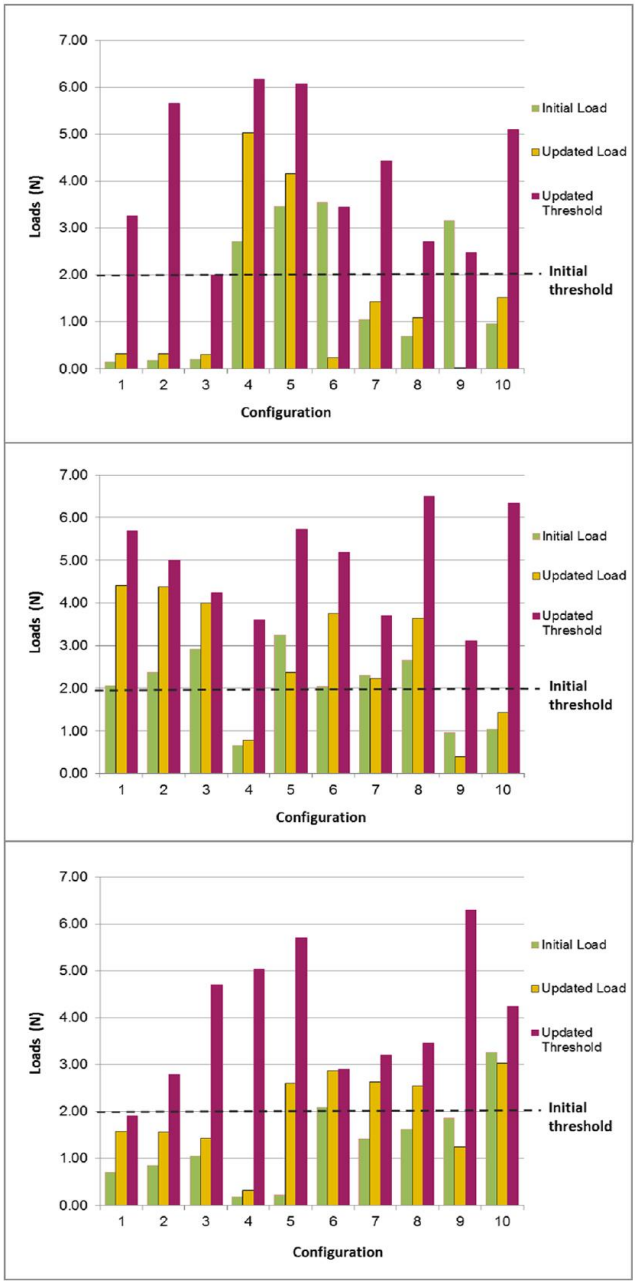

Figure 6. 10 configurations conducted by the simulation: (Top) Column no. 1; (Middle) Column no. 2; (Bottom) Column no. 3.

4.2. ANN

The figures below (Figures 7 to 9) present the results of the testing phase on the digital model. The first testing was conducted with no training before. Then the system was trained and after each training (iteration) it was tested again. 10 training phases were conducted.

Figures 7 and 8 show the results for the first 30 configurations of the testing set without training, after the first training set and after the tenth training set. The 'right answer' is the one given by the GA in the digital simulation to a similar set of inputs.

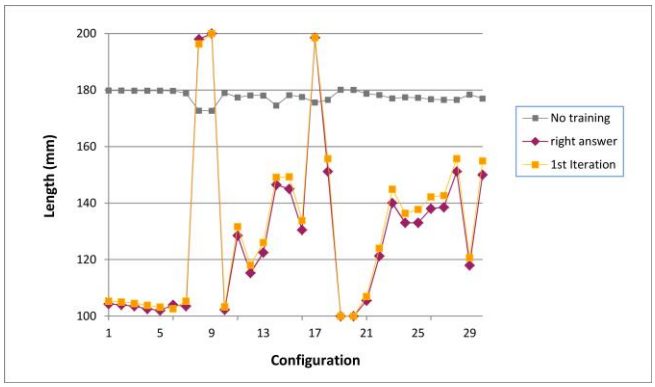

Figure 7. Length change, for a specific configuration, for the first two testing iterations, for a single column.

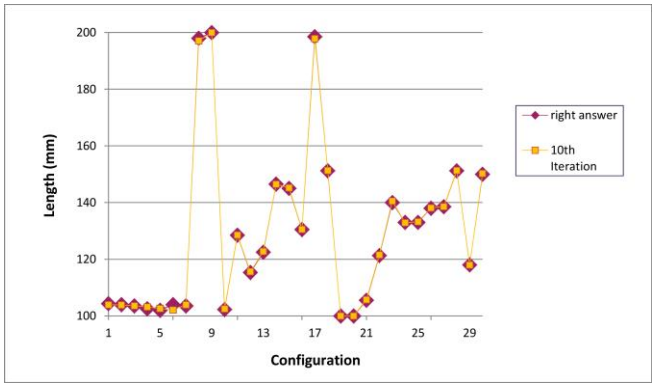

Figure 8. Length change, for a specific configuration, for the 10th testing iteration, for a single column.

Figure 9 shows the convergence of the system to the right answer by presenting the error rate. The error rate was calculated as the difference between the 'desirable length' of each column, as given in the GA database, and the length given by the ANN, for a specific set of inputs.

As can be seen in these charts, the algorithm managed to converge to the 'right solution' and successfully exhibit self-learning ability. The more iterations conducted, the bigger the improvement that is achieved by the algorithm.

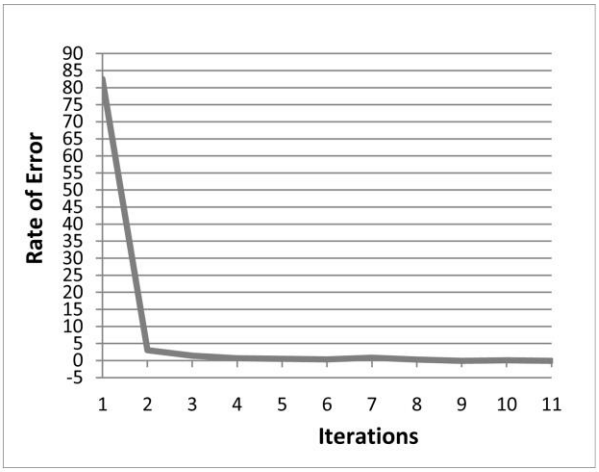

Figure 9. Error rate through the testing iterations

4.3. Physical Model

The last step of the study included an implementation of the ANN on the physical prototype. Figure 10 shows the adaptive behavior acquired by it. All measurements for these charts were in real time, as detected by the load cells and the position feedback of the linear actuator (the column). As Figure 10 shows, in all the examined configurations the system reached its target.

5. CONCLUSION

In this study the potential of a structural component to exhibit self-learning in order to adapt to an unpredictable changing environment was explored and successfully achieved using a case study. An ANN algorithm, trained by a set of optimised finite solutions created by a GA, was applied on a prototype.

The objective of this study focused on the potential for creating an adaptive structural element. The case study successfully presented self-learning abilities: the physical structure reacted to various forces, and managed to learn how to change its position in order to avoid buckling. These structural problems are often too complex to have one absolute solution. Thus, several assumptions and simplifications were made, as described above. These simplifications do not affect the results of these tests, as the main aim of this study was achieving an adaptive behaviour and by that to improve its performance. This goal was achieved and considered as proof of the study's aim. This

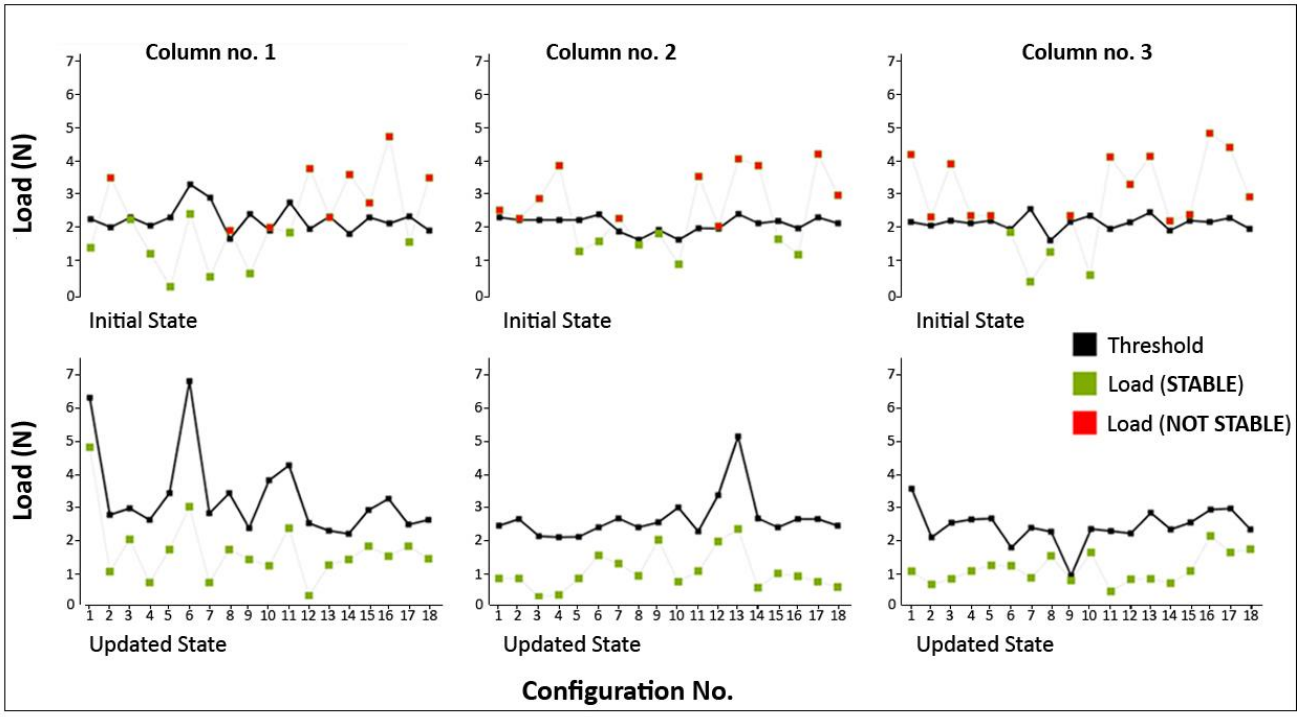

Figure 10. Configurations conducted on the physical prototype. Initial load / Initial threshold – Updated load / Updated threshold

method has a potential for further development and can be applied over different structural elements (façade elements, canopies, and structural components etc.) as well as to adapt to other parameters with an unpredictable pattern (human behaviour, weather conditions, combination of different parameters that will cause unpredicted patterns, etc.).

References

BONVIN, D., SRINIVASAN, B. (2012), 'On the use of models for dynamic real time optimization'. Available in: http://focapo.cheme.cmu.edu/2012/proceedings/data/papers/052.pdf [Accessed 01 September 2012].

DETAIL online-magazine, (2012). Available in: http://www.detail-online.com/architecture/news/maximum-load-carrying-capacity-with-minimum-material-input-018603.html. [Accessed 01 September 2012].

FLAKE, G. W. (1998), The computational beauty of nature: computer explorations of fractals, chaos, complex systems, and adaptation', Cambridge, MIT Press.

FOX, M., YEH, P. (2004), 'Intelligent kinetic system', available in: http://kdg.mit.edu/Pdf/ iksov.pdf [Accessed 01 September 2012].

HANNA, S., (2007). 'Inductive machine learning of optimal modular structures: estimating solutions using support vector machines', Artificial Intelligence for Engineering Design, Analysis and Manufacturing, vol. 21, No. 1, pp. 351-366.

HANNA, S., HESSELGREN, L., GONZALEZ, V. & VARGAS, I. (2010), 'Beyond Simulation: Designing for Uncertainty and Robust Solutions', in Proceedings of the Symposium on Simulation for Architecture and Urban Design at the 2010 Spring Simulation Multiconference, April 2010, Orlando, USA.

KOTA, S., HETRICK, J., RUSSELL, O., PAUL, D., PENDLETON, E., FLICK, P., TILMANN, C. (2003), 'Design and Application of Compliant Mechanisms for Morphing Aircraft Structures, Smart Structures and Materials 2003: Industrial and Commercial Applications of Smart Structures Technologies', Eric H. Anderson, Editor, Proceedings of SPIE Vol. 5054.

SOBEK, W., TEUFFEL, P. (2001). 'Adaptive Structures in Architecture and Structural Engineering'. In the proceedings of the Symposium on Smart Structures and Materials, at March, 2010. Newport Beach, CA, USA.

SENATORE, G., DUFFOUR, P., HANNA, S., LABBE, F., WINSLOW, P. (2011). 'Adaptive structure for whole life energy savings'. Intelligent Environments (IE), 2011 7th International Conference.

STERK, E. de T. (2006). 'Shape control in responsive architectural structures – current reasons and challenges'. In Proceedings of the 4th World Conference on Structural Control and Monitoring, San Diego, CA, USA, July 2006.

TEUFFEL, P. (2010). 'Architectural engineering and beyond'. In the 50th Symposium of the International Association for Shell and Spatial Structures. In the proceedings of 'Evolution and Trends in Design, Analysis and Construction of Shell and Spatial Structures.' Valencia, 2009. Proceedings. Editorial de la Universitat Politécnica deValencia.