P. Janssen, P. Loh, A. Raonic, M. A. Schnabel (eds.), *Protocols, Flows and Glitches, Proceedings of the 22nd International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA) 2017*, 147-157. 

# ESTABLISHING PARAMETRIC RELATIONSHIPS FOR DESIGN OBJECTS THROUGH TANGIBLE INTERACTION

EMAD AL-QATTAN[1], WEI YAN[2] and PHILIP GALANTER[3]
[1,2,3]*Texas A&M University, United States of America*
[1,2,3]*{emadkkqattan|wyan|galanter}@tamu.edu*

**Abstract.** This paper presents a method for translating physical interaction with design objects into parametric relationships. A framework of the method is created to automate the generation of parametric equations as modeling constraints. The prototypes developed for this work link digital models with their physical counterparts to create a hybrid and tangible interface that enables user interaction. The prototypes investigate linear and non-linear types of object relationships for creating parametric models. The results demonstrate a novel approach in architectural design that assists users in creating complex geometric relationships through intuitive interaction.

**Keywords.** Physical Computing; Parametric Design; Building Information Modeling; Tangible Interaction.

## 1. Introduction

Parametric design in a broader sense is an algorithmic approach to establish interdependent geometric relationships for generating architectural forms (Woodbury 2010). The process is inherently mathematical in nature, and with advances in digital design tools, it has been possible for architects to visualize and explicitly utilize mathematical logic in the design process (Stavric & Marina 2011). Since the appearance of the first digitized CAD system (Ivan Sutherland's Sketchpad), modeling tools have evolved to include a wider range of applications and procedures to setup geometric constraints. Each system - including: *Variants programming*, *history-based programming*, *Variational geometry and design*, *rule-based variants*, and *parametric features-based design* - offers its users distinct interactive and controlling features for parametric modeling (Monedero 2000). The formal practice of establishing such systematized geometric relationships and modeling constraints is achieved through text-based (e.g. Revit family editor, API, C#, Python, etc.) and/or graph-based (Grasshopper, Dynamo, etc.) programming applications. Algorithmic procedures enable the architect to translate design intents

into functional parametric systems using explicit mathematical expressions; nevertheless, the process requires specialized mathematical and programming skills, which often fall outside the architect's domain of knowledge (Stavric & Marina 2011). Research has shown that the lack of such skills results in limiting the architect's creative abilities (Kępczyńska-Walczak 2014), whereas the essential premise of design is the negotiation between the designer and the design object. For those reasons, the research presented in this paper proposes and demonstrates a method that captures and translates architects' intuitive interaction with physical design objects into mathematical equations to be utilized as constraints in digital parametric models. The method is tested and validated through a series of prototypes, with each composed of (1) a hybrid interface that links digital and physical models together, (2) a regression analysis method (linear and non-linear curve fitting functions) to automate the generation of object relationships in mathematical form, and (3) an architectural application utilizing the generated equations to create a parametric model.

## 2. Research Questions

Establishing object relationships in parametric models requires explicit use of mathematical logic to communicate design intents. The manual implementation of parametric equations in digital models is a challenging process, especially in the case of creating complex and non-standard architectural forms. The proposed research aims to address this limitation to improve the digital design workflow by answering the following questions:

- *What is the type of user-interface that can capture and translate design intents into object relationships to create a parametric model?* A common practice of establishing and controlling constraints in a digital model is done through text-based or graph-based applications. However, earlier research suggests that a hybrid interface provides a more flexible form of physical interaction to communicate a design intent and to manipulate a parametric model in the digital environment (Al-Qattan et al. 2016). Therefore, the proposed research will develop and utilize a hybrid user-interface to generate and setup constraints in digital models.
- *What type of parametric relationships can a hybrid user-interface represent physically and generate digitally?* Object relationships could be portrayed as linear and non-linear in nature. Through prototyping, the research will investigate the ability to generate both types of relationships through physical interaction, and illustrate which mathematical equations are of value to the design process when creating a parametric system.

## 3. Previous Work

Existing works utilizing tangible interfaces demonstrate unique opportunities to improve the digital design workflow, and users' perception of space and geometry. They process user and/or environmental data to control and manipulate modeling parameters. However, such examples operate through predefined parametric setups, i.e. architects establish object relationships beforehand. Conversely, the pro-

posed work demonstrates a novel approach in creating design object relationships, by enabling the hybrid system to generate the mathematical information required to construct the parametric model. Reversing form and geometry into their mathematical definition is a common practice in the fields of engineering, mathematics, computer science, statistics etc. Software packages utilizing methods for regression analysis such as Math.Net Numerics (Ruegg et al. 2016) have streamlined the process of generating mathematical equations depicting data correlation. Another example is *Trendline* in Microsoft Excel, a technique used to find the best-fit curve (linear, polynomial, logarithmic, etc.) for a set of data points. Regression analysis for design applications is found in the Grasshopper Plug-in *Mantis*, which operates by linking parametric workflows with Wolfram's Mathematica (Zaghloul 2013). Examples of work utilizing Mantis, and Math.Net Numerics with text-based and graph-based programming (such as C#, Python, etc.) have shown the possibility of extracting mathematical information from specific types of geometry. Therefore, the proposed research finds that incorporating such tools in a digital design workflow will provide new means to discover and extrapolate relational patterns and mathematical logic for establishing complex parametric systems.

## 4. Research Significance

The proposed method enables architects to establish object relationships in digital models through their intuitive interaction with physical objects. The workflow utilizes a data mining process (regression analysis) to generate mathematical equations as the relationships. This approach releases the architect's burden of creating complex mathematical equations from scratch when modeling a parametric system. This study contributes to the future application of machine learning in the context of design. Further development may ultimately create a system capable of interpreting design intents, learning user preferences, and most importantly, enabling collaboration between the architect and the machine in the process of generating design options, facilitating parametric simulation, and design optimization.

## 5. Prototyping

Each prototype investigates a type of: (1) kinematics, (2) object relationships, and (3) regression analysis procedures; and is composed of a digital modeling environment and an *artifact*. The artifact is the tangible part of the interface that consists of an architectural representation (a design object) and a control system (sensors, actuators, and a microcontroller). The process is validated by monitoring objects' geometric response in both environments using the generated equations.

### 5.1. PROTOTYPE 1

Prototype 1 investigates a *linear* relationship of object parameters as an example of design intent. The experiment is conducted into two phases: (1) data logging and generating the mathematical equation, and (2) setting up the equation as a constraint in the digital model. The set of tools utilized for this experiment are categorized into two groups:

- *Software tools* for (1) modeling: Revit (Autodesk BIM authoring tool) and Dynamo (visual programming environment for Revit); (2) programming: IronPython (programming language); (3) linkage: Firefly plugin for Dynamo (Payne & Johnson 2012); and (4) data management: MS Excel.
- *Hardware* tools for creating an integrated circuit of electrical and electromechanical components to establish the users' interactive control system include: a mini servomotor, two rotary potentiometers, and a microcontroller type Arduino UNO.

The prototype consists of a BIM model composed of an array of three Revit family panels, and an artifact composed of a control system and three corresponding physical panels (figure 1).

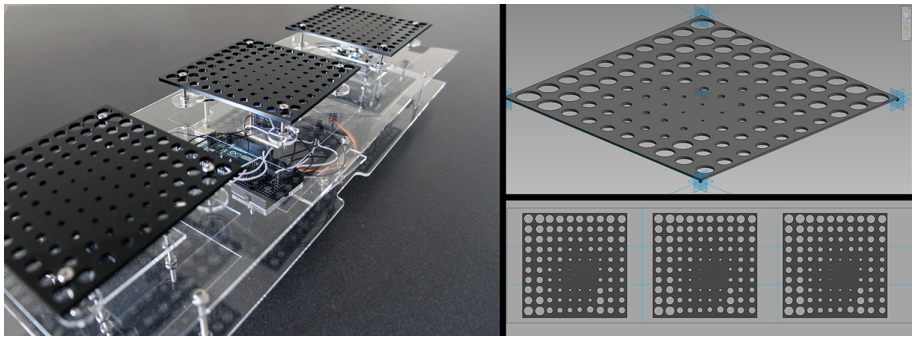

*Figure 1. Artifact with panels (left) and corresponding Revit family panels (right). The panel array in Revit (lower right) is linked to the artifact in this experiment.*

The prototype is operated by having the user physically rotating panels 1 and 3 horizontally (panels at the opposite ends of the artifact). Each of the two panels are attached to a *rotary potentiometer*, a sensor that monitors changes in the panels' angles of rotation. Sensor values will be used in a *linear* regression analysis to generate a mathematical equation depicting the panels' relationship. Following in the process, is parametrizing the equation and setting it up as a constraint in the digital model. The equation will be used to calculate the angles of rotation for Panel 2. In summary, the workflow for Prototype 1 includes the following tasks: (1) linking the BIM model and the artifact through Firefly, (2) performing a linear regression analysis, (3) generating the parametric equation, and (4) calculating the angles of rotation for Panel 2 and sending the values to both the digital model and the artifact (figure 2).

In Phase 1, linear regression analysis requires data samples to find the best fit curve, establish data correlation, and generate the mathematical equation. The rotary potentiometers attached to the panels provide data samples when rotating Panel 1 and 2 manually from 0 to 180 degrees, i.e. Panel 1 is rotated by {X1, X2, X3} degrees and Panel 3 rotated by {Y1, Y2, Y3} degrees, respectively. Once the two panels are rotated, sensor values are recorded and stored in a CSV file (figure 3).

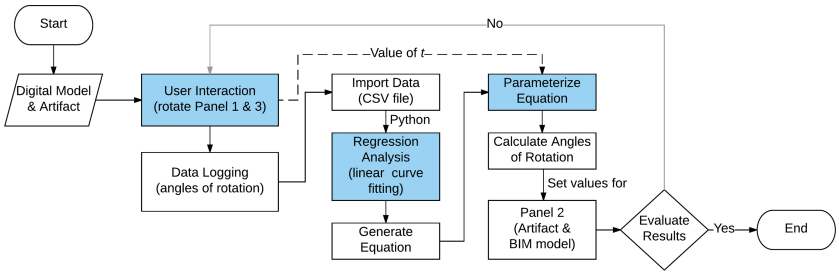


*Figure 2. Workflow for Prototype 1.*

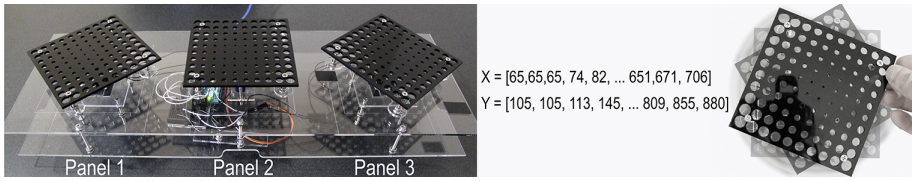


*Figure 3. Panels in their order (left); data logging process for regression analysis (middle); and user manually rotating Panel 1 (clockwise) from 0 to 180 degrees (right).*

In Phase 2, the CSV file is imported in Dynamo for regression analysis. The regression analysis procedure is programmed using IronPython and integrated in the Dynamo workflow. The analysis procedure will calculate two values the *Slope* (equation 1) and *y-Intercept* (equation 2) using the following equations (Yan & Su 2009):

$$m = \frac{\sum[(x_i - \overline{x})(y_i - \overline{y})]}{\sum\left[(x_i - \overline{x})^2\right]} \tag{1}$$

$$b = \overline{y} - (m\overline{x}) \tag{2}$$

The format of the linear equation used for establishing panel relationship is based on the *Point-Slope* from of the line equation ($y = mx + b$), where $m$ is the Slope and $b$ is the y-Intercept. The generated equation at this point describes the relationship between Panel 1 and 3, excluding Panel 2. The angle value required to rotate Panel 2 is obtained by solving $x$ and $y$ using the generated line equation. Nevertheless, the current process does not support parametric functionality that is required to control the panel array. Therefore, an additional procedure in Dynamo was developed to parametrize the equation. The process includes setting $t$ as an independent variable to solve $x$ and $y$ ($x = t$, $y = mt + b$). The *t* value in the equation is substituted with the angles of rotation obtained from the rotary potentiometer attached to Panel 1. The calculated results using the parametric equation provided all the possible interpolations (angles of rotation) for rotating Panel 2 in between the other two panels. At this stage, the parametric system is setup, and further changes in the angles of rotation for Panel 1 and 3 will automatically update Panel 2's rotation in both the BIM model and the artifact. In the artifact, Panel 2 will rotate using the mini servomotor, and in the BIM model by using an instant *Angle* parameter (figure 4).

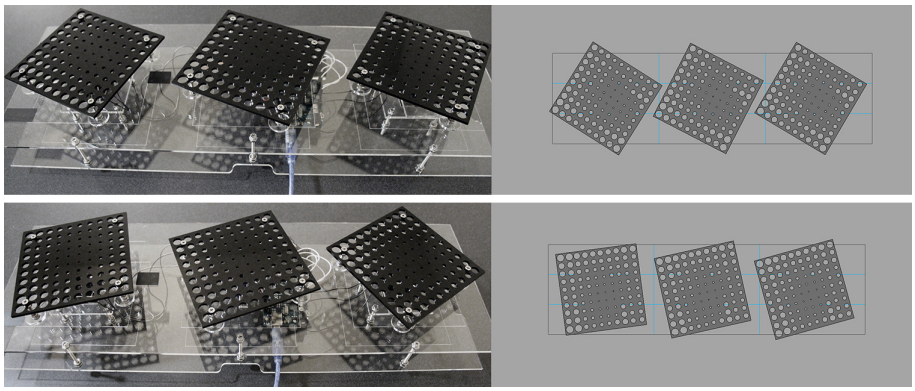

*Figure 4. Physical interaction with Panel 1 and 3 will automatically rotate Panel 2 using the generated parametric equation.*

Prototype 1 shows that utilizing linear regression analysis in this framework helped in mathematically deducing the relationship between physical design objects, and setting up a parametric relationship between the panels in both the digital and physical models.

The purpose of the prototype is not to simply determine the rotation angles of Panel 2, but to determine the linear function and its coefficients representing the parametric relationship of the array of panels. Once the relationship is setup in the digital model, the architect can make changes to the model and the design intent will remain intact.

### 5.2. PROTOTYPE 2

The second experiment investigates *non-linear* object relationships as a more complex example of design intent. The work involves a two-phase process similar to the previous experiment to generate the mathematical equation and establish the parametric relationship between design objects. The set of tools chosen for this experiment are:

- *Software tools* for (1) modeling: Rhino (3D modeling environment) and Grasshopper (visual programming environment); (2) programming: C# (programming language); (3) and linkage: Firefly plugin for Grasshopper.
- *Hardware tools* for creating the control system include: eight linear soft potentiometers (ribbon sensors) and a microcontroller type Arduino MEGA.

The prototype consists of a Rhino model of an architectural space with its roof made of an array of louvers, and an artifact composed of a control system and a physical representation of both the space and louvers (figure 5). In the artifact, each louver is connected to a ribbon sensor, which is located on the model's side. The sensors measure the vertical distance - in inches - between each louver and the model's floor, and send the values to Grasshopper through Firefly. In Rhino, each vertical distance value is represented as a point on the XZ plane to indicate the location of the physical louvers.

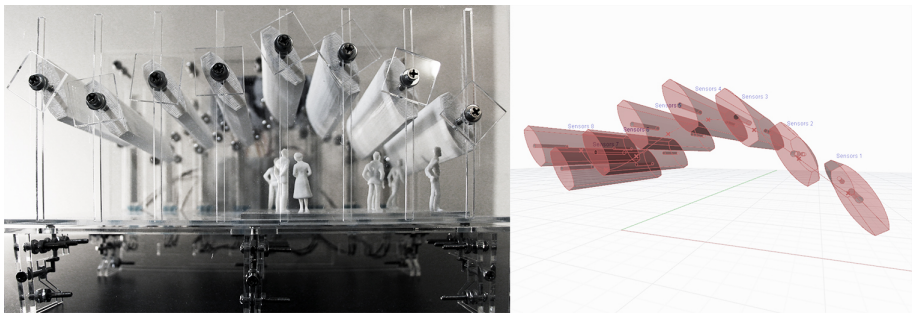

*Figure 5. Hybrid interface developed for Prototype 2.*

The points generated by the sensors are used for performing the non-linear regression analysis. The procedure utilizes a polynomial curve fitting function to generate the mathematical equation depicting the louvers' layout, which will be used later to setup the parametric model (figure 6). Regression analysis in this experiment is programmed using C# and Math.Net Numerics, and similar to the previous experiment, the procedure is integrated with the visual programming workflow. The prototype in this experiment has a unidirectional link, i.e. architects provide the system with analog data by moving the physical louvers to generate responses in the digital model; whereas Prototype 1, has a bidirectional link that enables digital and physical interaction and response.

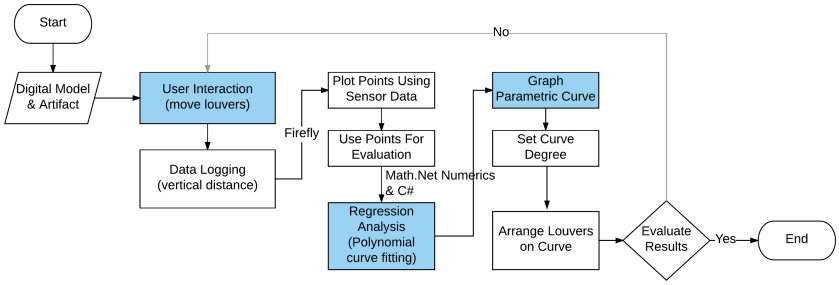


*Figure 6. Workflow for Prototype 2.*

Non-linear regression analysis in Prototype 2 will assist in generating the mathematical equation that depicts complex object relationships. The equation used for this work is assumed to be a polynomial with a single variable $x$ (Tan 2011) as shown in equation 3:

$$a_n x^n + a_{n-1} x^{n-1} + \ldots + a_0 = 0 \qquad (3)$$

In Phase 1, the vertical distance values are plotted in the Rhino scene as points shown as crosses, and each point is labeled with its corresponding ribbon sensor (figure 7, graph 1). A total of eight points are generated for the eight physical louvers. The programming workflow using Math.Net Numerics will find a polynomial curve that best fits the sensor points (figure 7, graph 2). In Phase 2, the polynomial function generated by the regression analysis is used to layout and move the louvers in Rhino parametrically (figure 7, graph 3).

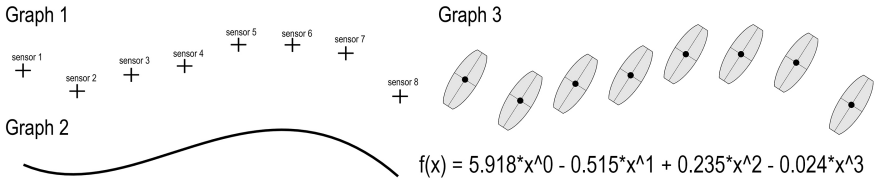


*Figure 7. Regression analysis process and implementation of the parametric polynomial curve.*

The parametric polynomial curve is tested in this workflow by letting the architect reconfigure the louvers' layout in the artifact (figure 8). The results show that the louvers' layout in Rhino updates instantaneously with the architect's real-time interaction with the physical objects. The louvers' reconfiguration shown in figure 8 generates a polynomial curve of degree 4 ($f(x) = 7.057x^0 + 0.122x^1 - 0.540x^2 + 0.123x^3 - 0.007x^4$)

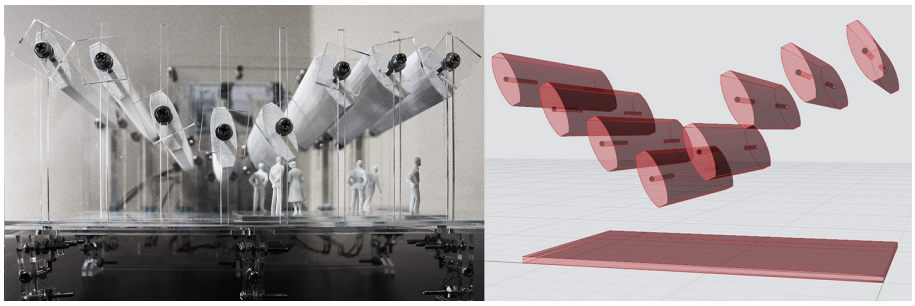

*Figure 8. Testing the workflow, the generated equation in this example is a polynomial of degree 4.*

The workflow in Grasshopper enables the architect to fine-tune the louvers' layout in Rhino to match their physical counterpart by increasing or decreasing the degree of the generated polynomial equation. Figure 9 shows the process of fine-tuning the polynomial equation from degree 4 to 8 using the louvers' layout previously shown in figure 8. The results demonstrate that a degree 8 polynomial is the best fit, yet high degree order may not be desirable in a design context due to its complexity for construction and representational quality.

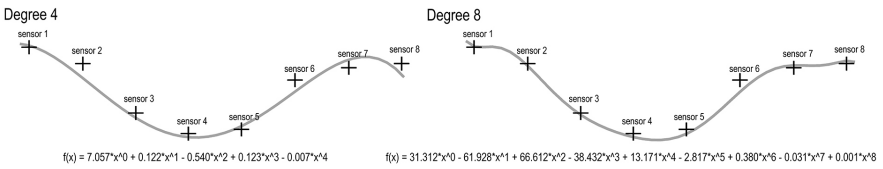


*Figure 9. Fine-tuning of the polynomial curve by increasing the polynomial from degree 4 to 8.*

The integration of the parametric polynomial curve within the developed programming workflow assists the architect in making further changes to the digital model while maintaining the design intent. For example, if the number of louvers is to be changed, the updated array will be automatically re-located along the polynomial curve. If the coefficients of the polynomial are used as parameters, we can fine-tune the curve by adjusting some of the coefficients to obtain different design configurations.

Note that this method of creating curves for parametric design is different from the widely used NURBS curve generation method, where the user's input points are used as control points for creating low degree Bézier curves that are pieced together to form a complex curve. The purpose of this experiment is to demonstrate that physical interaction between designers and the artifact can assist in generating complex relationships to create parametric models (a parametric curve in this example).

## 6. Discussion

The two prototypes demonstrate the possibility to automate the generation of mathematical equations for two types of object relationships, linear and non-linear, by translating architects' intuitive interaction with physical design objects, and without the need to possess advanced mathematical skills.

The two experiments test the method to create both simple and complex (linear and non-linear) geometric relationships in mathematical equation forms. The approach is applicable in the conceptual design phase where architects seek to explore dynamic, and non-standard forms and geometric configurations. The prototypes can not only translate physical transformation information into digital models as existing systems can do, but also capture and translate the design constraints or relationships into digital models to make them parametric and embedded with design intents. The method potentially provides great value to BIM and parametric modeling tools (Revit/Dynamo, Rhino/Grasshopper, Digital Project, CATIA, etc.) to enhance their parametric modeling capability. The benefit of the proposed work lays in its ability to overcome the challenges associated with manual implementation of parametric constraints, and focusing on inviting architects' haptic and visual senses in the digital design process.

## 7. Conclusion

The proposed work considers physical interaction as an alternative method for establishing parametric models, in addition to computer programming, which is of significant difficulty for architects. The prototypes demonstrate a method that is capable of (1) determining the type of object relationships, (2) generating parametric equations, and most importantly (3) its application in the context of design.

The focus of this work is on translating tangible interaction into parametric constraints for digital modeling. Once a parametric model is constructed further studies using optimization algorithms will be done to address certain architectural conditions related to aesthetics and performance. It is expected that the proposed method can assist architects in reducing the extensive effort associated with encod-

ing mathematical procedures in the programming workflow. The use of regression analysis with a simple data set has shown the possibility of constructing parametric models. Further development of the method will provide users with a system capable of implementing more complex design intents. Additionally, the work will further investigate the translation of deign intent between the digital and physical environments. As seen in the experiments, some discrepancies occurred, which were caused by human users' inaccuracy of spatial perception and motor skills.

**Acknowledgment**

We would like to thank our colleagues at Texas A&M University for their help and support, Dr. Saied Zahrammer for his assistance in Python programming and Assistant Professor Dr. Negar Kalantar Mehrjardi for providing us with the digital fabrication machinery.

## References

Al-Qattan, E., Yan, W. and Galanter, P.: 2016, Developing a Tangible User Interface for Parametric and BIM Applications Using Physical Computing Systems, *Proceedings of eCAADe 34*, Oulu, Finland, 621-630.

Kępczyńska-Walczak, A.: 2014, The Act of Design - Beyond the Digital?, *Architecturae et Artibus*, **6**(1), 24-28.

Monedero, J.: 2000, Parametric Design: A Review and Some Experiences, *Automation in Construction*, **9**(4), 369-377.

Payne, A. and Johnson, J.K.: 2012, "Firefly" . Available from <http://www.fireflyexperiments.com>.

Ruegg, C., Cuda, M. and Van Gael, J.: 2016, "Math.Net Numerics" . Available from Math Net< http://numerics.mathdotnet.com/>.

Stavric, M. and Marina, O.: 2011, Parametric Modeling for Advanced Architecture, *International Journal of Applied Mathematics and Informatics*, **5**(1), 9-16.

Tan, S.T.: 2011, *Applied Calculus for The Managerial, Life, and Social Sciences: A Brief Approach*, Cengage Learning, Belmont, CA, USA.

Woodbury, R.: 2010, *Elements of Parametric Design*, Routledge, USA and Canada.

Yan, X. and Su, X.: 2009, *Linear Regression Analysis: Theory and Computing*, World Scientific publishing Co. Pte. Ltd. , Hackensack, NJ, USA.

Zaghloul, M.: 2013, "Mantis" . Available from <http://www.grasshopper3d.com/group/mantis>.