3 Information Processor - Digital Form with Computational Means

"Give me a gun and I will make all buildings move."

Bruno Latour, Albena Yaneva

§ 3.0 Introduction

How computational technology start to take place and gradually become being heavily involved/implemented in the design process of architectural design.

In the architecture domain, not only the proportion of the assistance from computational techniques has been increasing exponentially, but also, the role they play has been gradually shifting from a supporting one to a generative one. No longer limited to being a complex mathematics calculator, computers, have become a ubiquitous necessity in our daily life and even influence the way we live. This, is especially true for the young generation who were born in this digital world, mainly referred to as the "Generation Z"[26]. Business Insider, a fast-growing business media website, mentioned that "Gen Z-ers are digitally over-connected. They multitask across at least five screens daily and spend 41% of their time outside of school with computers or mobile devices, compared to 22% 10 years ago, according to the

26 Generation Z are the cohort of people born after the Millennials. The generation is generally defined with birth years ranging from the late 1990s through the 2010s or from the early 2000s to around 2025. Please see the details through: https://en.wikipedia.org/wiki/Generation_Z

Sparks & Honey report[27]." When Alan Turing first invented the room-sized "Turing Machine" to decipher Nazi codes, he couldn't have expected that this giant machine could one day be put into one's pocket and efficiently compute a million times more data. As compared to the era of tools, such as paper and pen, the computer, in today's context has been heavily utilized and relied upon as a powerful instrument. This change is remarkable, considering the relatively short period of time, especially after 1981 when the first IBM personal computer was released (Mitchell, 1990). Architecture Design cannot be excluded from this inevitable technological tendency. Even the most conservative architecture firms are now required to deliver digital technical drawings to communicate amongst designers, clients, and construction firms in the present scenario. Incorporating computer technology in today's context also provides young designers the opportunity to experiment with creating relatively complex geometry based architectural space. But before applying this powerful technology in architectural design, the crucial knowledge behind it that architects had to understand and realize was the manner and procedure of "Processing of Information". Without information, the computer would be just lying on one's desk as a useless cube, like a vehicle without a driver, or a body without a soul. The shifting roles of computer technology in architectural design are obviously defined by the manner of how designers interpret, digest and operate/process the streams of information flow.

However, dealing with information is not new to architectural design, which already thrived on multi-stakeholder based information exchange long before computers arrived. In order to preserve the measurements underlying his design ideas, Brunelleschi, as an architect in the early Renaissance, investigated means of making projective geometric drawings in order to capture 3-Dimensional information, which subsequently led to the development of parametric perspective space for the first time (Lorenzo-Eiroa, Form:In:Form on the relationship between Digital Signifiers and Formal Autonomy, 2013). During the 14th and 15th century, Girard Desargues developed the concept of "the point at infinity" to create an alternative way of constructing Euclidean geometry in perspective drawings by using vanishing points as references (Lorenzo-Eiroa, Form:In:Form on the relationship between Digital Signifiers and Formal Autonomy, 2013). Not to mention the great influential invention of the Cartesian coordinate system by René Descartes, who, set up the fundamental principles of spatial collaboration both in 2D and 3D graphics. Implausible, in the early 15th century, when paper began to replace parchment as a drawing medium, Italian architects had well understood the concept of graphic projection as communication document shared amongst people dedicated to the

27 Please see the full report and statistics through the website referred: http://www.businessinsider.com/generation-z-spending-habits-2014-6?IR=T

construction process (Weisberg, 2008). The communication documents here refer to the so-called technical/engineering drawings as a medium where the projects are represented in a proper scale, with precise measurements and understandable geometric visualization. To use these technical/engineering drawings not only in terms of translating, preserving, creating but also communicating information of their spatial ideas, architects have shown remarkable abilities to confront information communication as a necessity in the design process, and also reveal the intensive and intimate relationships between information and form since the Renaissance. In other words, architectural design can be seen as an on-going process coupled with streams of information in order to seek/generate a relatively rational form as a specific resulting outcome (with/without computational techniques).

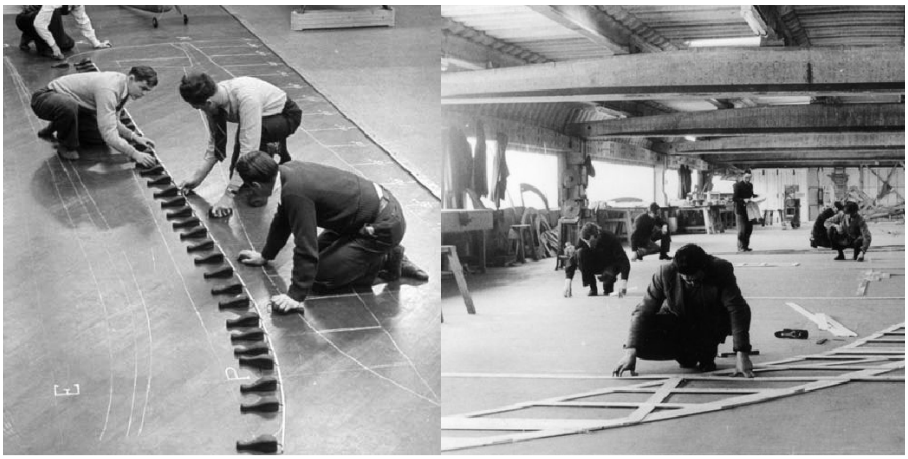

FIGURE 3.1 Left: Course in Airplane Lofting, Burgard High School, Buffalo, NY, USA, January I, 1941. Right: Picture of People working on Airplain Lofting(source: http://cornelljournalofarchitecture.cornell.edu/read.html?id=74, https://i.pinimg.com/736x/0e/79/bb/0e79bbaa467027c649fd7452afb0cfe3.jpg).

With the original intention of "Technical/Engineering Drawings", designers (not only architects) basically created two fundamental methods of dealing with information pragmatically for a long time: 1. To store precise references for fabrication and construction. 2. To present design ideas to the clients with understandable visualization as a communicating medium. But the increasing intricacy of the design, the precise demands for measurements, and the amount of requirements for the reproduction of construction drawings made it extremely difficult to manually illustrate hard copies with hand drawing by traditional tools such as pen, paper, and ruler. Take aeronautical drawings for example, it is even more challenging, because of the demands to produce accurate drawings at 1:1 scale for large components of an aircraft,

where it is impossible to convert smaller drawings into the templates needed for production (Weisberg, 2008)(Figure 3.1). This is the moment when computers started to become important and be considered as a new medium/tool to assist and accelerate both the design and production processes. It was also this time when the terminology of "CAD"(Computer Aided Design) was first introduced to the world. Although the main goals of CAD techniques back then still remained embedded in storing and presenting designs, computational techniques have increasingly changed their role by providing multiple ways of generating, analyzing and visualizing data. This in-turn has resulted in developing informed complex geometry based design solution sets as novel spatial outcomes.

The Form is interplaying amongst itself as an information emergence by executing particular approaches for conveying information.

"**Design is the computation of shape information that is needed to guide fabrication or construction of an artefact**" is an apt definition for the early stage of Computer Aided Design by William Mitchell (Mitchell, 1990). However, in this case, information is mainly considered as shape/geometry related data, extracted from a pre-conceived form to assist in any production process after the design decisions have been mostly completed. However, since years of developments and evolutions of the computer technology utilized in architectural design, computers are not treated merely as drawing machines to generate documents for construction work, or modeling machines to create fascinating rendering graphics to present and convince clients. The computation technology has successfully adapted/shifted itself to become an "information processor" rather than a pure "information duplicator". In the publication of "***Algorithm Form***" (Terzidis, 2006), Kostas Terzids made an explicit distinction about "**Computerization"** and **"Computation**". "**The dominant mode of utilizing computers in architecture today is that of computerization; entities or processes that are already conceptualized in the designer's mind are entered, manipulated or stored in a computer system. In contrast, computation or computing, as a computer-based design tool, is generally limited. The problem with this situation is that designers do not take advantage of the computational power of the computer**". This concise quote not only reveals existing problems of architects being predominantly occupied with computerization, yet, it also indicates a clear turning point of feeding and extracting information to and from computers in a different but also efficient way.

"**Form**" has always been a complicated and debated topic as regards the role it plays in architectural design no matter what kind of dogma is followed. Here, it's crucial to state that this research emphasizes that form should be perceived as having an intimate relationship with relevant contextual data in a dynamic fashion, and the approaches involved in processing this data into form-finding information.

The key concept delivered here is: "**The form can be informed by contextual information as a continual process".** The following sections will open up discussions focusing on different strategies for associating form and information with different computational methodologies in architectural design (computation) far beyond the conventional computational approaches which served towards storing and presenting (computerization) form. These methodologies have been categorized as: **Form Sculptor, Form Generator, Form Animator, and Form interactor.**

§ 3.1 FORM SCULPTOR

= utilizing 3D software intuitively as an exploration tool for design purposes/intentions.

Form Sculptor is defined as a method wherein existing 3D software is used to explore design ideas in architecture. It doesn't sound like an innovative idea at all, but in fact, this methodology has been only executed since just over a decade. Sketchpad[28](Figure 3.2), developed by Ivan Sutherland using the TX-2 computer in Lincoln Laboratory in 1959, which was around 20 years before IBM released the first personal computer was one of the first pioneering CAD systems (Weisberg, 2008). Ivan's original idea about Sketchpad operating in the design process was clearly written in his Ph.D. dissertation: "**Construction of a drawing with Sketchpad is itself a model of the design process**" (Sutherland, 1963). However, major developments of implementing computer graphic systems in architectural design went in a contrary direction inclined towards becoming a convincing visualized representation of the designers' ultimate vision of the project. This conservative way of using computer graphic systems as a virtual template/ canvas or material to draw or model the final design project is obviously considered as a "computerization" process. Certainly, there is no design intention involved in the words and notions of making a "digital drawing" or "digital model", which is in a sense the common and typical misleading idea of the terminology of "Digital Architecture" and "CAAD Process" prevalent amongst the general public. Architects who engage with reproducing and storing tasks for the purpose of re-presenting their designs virtually with digital tools should thus not be considered as members of the "Digital Architecture" realm. **Form Sculptor** does not refer to such kind of computerized architecture.

28 Please check the videos for more information about Sketchpad: https://www.youtube.com/watch?v=USy-oT_Ha_bA and https://www.youtube.com/watch?v=BKM3CmRqK2o.

Manifesting curvilinear geometries has always been seen as a difficult geometric task within digital software and fabrication sectors for the architectural community. This issue, however, has been successfully addressed within the automobile, aircraft and naval shipbuilding industry with "Computer Numerical Controlled" (CNC) machines for fabrication purposes. But long before the computer was invented, analog crafting methods of building curvilinear structures have been developed with relatively conventional tools and devices. For instance, lofting is one of the crucial techniques of constructing a boat frame through several sectional profiles, and sweeping is another approach by carving out clay or sand as a doubly curved surface from the other directions perpendicular to the lofting axis. Both of these are fundamental functions in surface modeling software (Young, 2012). It took a few years for computer scientists to translate most of these crafting techniques into a computer algorithm to build up a curvilinear line with compatible computational processing power so that it appeared on screen in real-time. For instance, albeit it's still being in a wireframe geometric system, Pierre Bezier, in 1972, while working with Renault managed to mathematically define a digital automobile surface and generate data corresponding to it in order to drive a milling machine for production. This is when he created and implemented the well-known techniques behind the Bezier curves and surfaces. A year later, at the "PROLAMAT" conference, Ian Braid from Cambridge's CAD Center, presented BUILD using B-Rep(Boundary Representation) technology for 3D modeling in 1973. At the same conference, Professor N. Okino from Hokkaido University has developed a CSG-based solid modeling which could operate boolean combination with primitive shapes. B-Spline(Basic Spline), originally represented as a long strip of wood or metal to mark out the curves created by the lofting profiles while building a boat, was also described as a new digital approach by Rich Risenfeld, a Ph.D. graduate of Syracuse University in the same year. Two more Ph.D.'s from Syracuse University, Ken Versprille and Lewis Knapp, are credited by many people as being the developers and key figures behind the evolution of NURBS (Non-Uniform Rational B-Spline) around 1975 to 1979 (Weisberg, 2008). All the aforementioned researchers contributed to making a huge leap not only in the realm of CAD but also CG (computer graphics), but it was still somehow a bit difficult for architectural designers to execute directly such computer programs even during the year 1986 when AutoCAD initiated its launch of the first PC version software and took over the design software market. After years of evolution of UI (user interface) of modeling software, architects can now freely manipulate and improvise 3D modeling functions to create complex shapes. With this freedom of shape making, computers are now involved in the design process itself, rather than being used as a representational machine duplicating the designers' ultimate ideas. This conclusion coincidentally matches to what Mario Carpo stated in his article, an introduction of "***Twenty Years of Digital Design***" (Carpo, 2012), "**In fact, in the first instance, a meaningful building of the digital age is not just any building that was designed and built using digital tools: it is one that could not have been either designed or built without them**".

Thus, **Form Sculptor** should be seen as an approach where computational power becomes a necessity for discovering not only the form but also the spatial quality of architecture.

FIGURE 3.2 Introducing and Demoing the Sketchpad to the general public on a TV program. (source: https://www.youtube.com/watch?v=USyoT_Ha_bA).

Under this definition, architects who have been metaphorically deemed to be **Form Sculptors,** try to explore the diversity of forms to a certain extent by utilizing the capability of the 3D software. Forms should be seen as a "formation" process rather than a static and solid result. Unlike the real sculptors, who execute crafting techniques, like carving, shaping, modeling and fashioning according to chosen materials in the real physical world, the **Form Sculptor** deploys essential transformation functions within the selected 3D software, such as scaling, shifting, twisting, tapering, etc. in general digital modeling, and lofting, sweeping, patching, fairing, etc. in surface modeling work with principles of B-Spline, B-Rep, NURBS or even topological (blob) calculation to determine the shapes of the objects in a virtual reality space. However, a crucial aspect has to be repeatedly emphasized here: ideally, **Form Sculpting,** as a computational approach, should go beyond utilizing computer technology to produce pre-determined form. On the contrary, such free form manipulation with user-friendly interfaces in current 3D modeling software undoubtedly enhance design creativity and tend to further architectural development to challenge the conventional and conservative definition of space and functions (Carpo, 2012). In the stage of the **Form Sculptor**, the required information for the design tasks can be searched, filtered and digested during the operations of the modeling process with the powerful 3D software, and simultaneously collaborating with the designer's mind. Architects gain complex form modeling based advantages from powerful 3D modeling software, while the

disadvantages/missed opportunities are revealed from the way these are utilized. Since the **Form Sculptor** relies too heavily on existing generic modeling functions, to a certain extent it constrains the creativity within the box of "default modeling functions". Besides, the operation of form modeling remains a relatively linear procedure which also further limits the potentials of computational processes which could utilize modeling processes from a distributed perspective, akin to a swarm. In other words, modeling each and every step of the formation process in linear detail is a misuse of the powerful computational technology in design. The other crucial critique of **Form Sculptor** is that the design output depends too much on an architect's personal intuitive sensibility for aesthetics, albeit this also is a central issue in most other design methods wherein subjectivity determining computational processes remains difficult to prevent. In order to search for solutions to such challenges, some architects decided to look for an answer by shifting from the "**Form Sculptor**" to the "**Form Generator** " methodology as a potential escape.

§ 3.2 FORM GENERATOR

= development of algorithms with multiple parametric inputs to generate performative forms associated with the appropriate usage of computational power in architectural design.

The "algorithm" as an information processor, obviously becomes the crucial element within the form generating process. But before rapidly jumping into the world of algorithms, it's crucial to acknowledge the time when complex geometries were being visualized without the assistance of computers but via mechanical tools. This will help in establishing hidden connections leading to today's algorithmic and coding driven innovations. So to begin with a different starting point, instead of relating directly to mathematical formulas, algorithms can in a sense be realized as physical instruments to illustrate simple to complex geometric principles via mechanical tools since yesteryears. Parametric thinking resembles the logic behind mechanical equipment, which has been used for decades as tools to record crucial physical notation (such as a drafting compass). For instance, Albrecht Dürer, who demonstrated curvilinear-line tracing during the time of German Renaissance (Cache, 2012). The illustrating instruments Albrecht Dürer invented to a certain extent have already embedded the relevant algorithms defined according to the combination of a mechanisms' movements and the equipment's dimensions in a parametric relationship. More importantly, these logics underlie ways in which computers can realize similar

graphical effects via algorithms on screen in real time. Most of these drafting tools can be seen as a physical realization of the algorithms/formulas. Take the simple drafting compass as a common example, with its dimensions, it can concisely define the central point with the needle in one leg and open up a certain distance as a radius with the other leg attached to a pen. Then once the needle is fixed on the paper, the movement of spinning the head of the compass to make traces of the pen can be interpreted as a methodology to draw a perfect circle. This fundamental setting can be easily translated and applied to the computer as an algorithmic code to exhibit another perfect circle digitally on screen. Here, it is crucial to point out that without numerous explorations with such inventions of illustrating tools, computer graphics as a base of digital design would have been relatively difficult to realize through scripting alone.

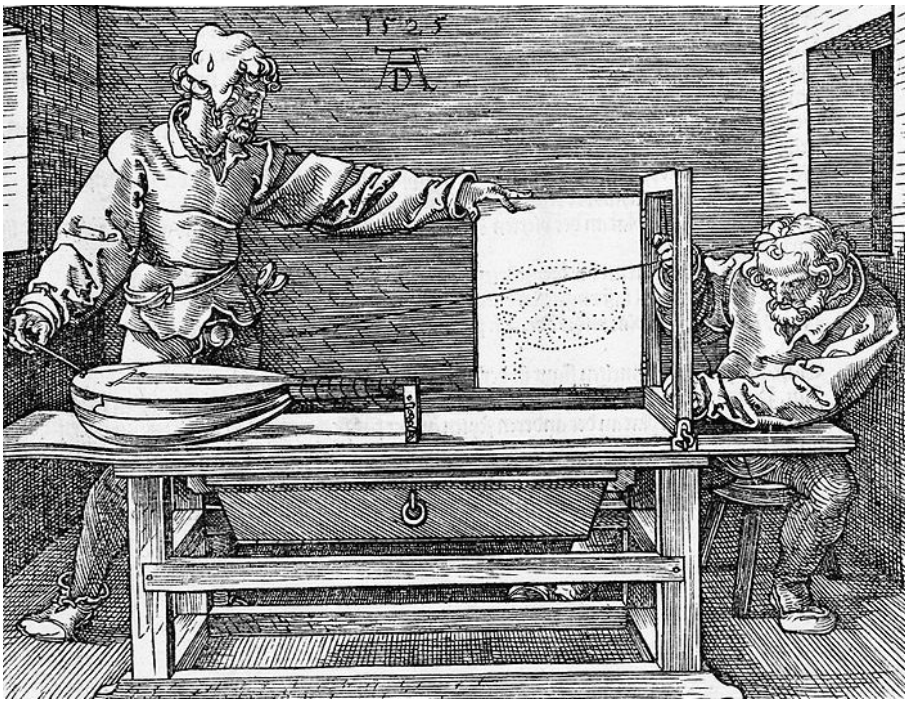

FIGURE 3.3 A drawing showing the usage of the perspective drawing instrument invented by Albrecht Dürer in the 15th century (source: https://commons.wikimedia.org/wiki/File:Duerer_Underweysung_der_Messung_fig_001_page_181.jpg)

In "***Underweysung der Messung (Instrument in Measurement)***" (Dürer, Albrecht & Formschneider, H. Andreas (Nürnberg), 1525) several curvilinear geometries are seen as the ancestors of B-Spline and NURBS used in current CAD software, such as

snail curve, spiral curve, epicyclic circle, serpentine curve (even in 3D). These were demonstrated by Albrecht Dürer's incredible mechanical drafting instruments. Amongst one of the first theoreticians of perspective, he even invented the first mechanical imaging device requiring no human eyes as references to visualize 3D space on a 2D planar surface (Figure 3.3). If Dürer's instruments can be interpreted as a graphics generating tool, so do algorithms in computers which can be seen as parametric visualization machines. The algorithms and parametric principles executed by the **Form Generator** are akin to Albrecht Dürer's physical instruments, attempting to discover more complex spatial formations with the assistance of computing power. This kind of parametric design thinking had been ignored for years until the invention of computers showed their potential in architectural design for generating relatively complex shapes under the demands initiated from the Deconstruction movement (Deconstructivism) (Carpo, 2012). Who would have imagined that this form-finding process with parametric algorithms would become mainstream today through the possibilities offered by coding techniques?

So far in this section of the **Form Generator**, algorithms were interpreted to play the roles of form finding/generating processes/tools operating on the input of a geno-type, which results in the production of pheno-type, without offering insight into the generative process (Akin to a blackbox). This sentiment is echoed in Malcolm McCullough's article "***20 Years of Scripted Space***" (McCullough, 2006): "**First you set up some rules for generating forms, then you simulate them to see what kind of a design world they create and then you go back and tweak the rules**". Especially after user-friendly visual programming languages were introduced to architectural design within parametric modeling software suits such as Grasshopper in Rhino, Dynamo in Revit, young architects were all fascinated to see what parametric computational technology can offer and were tempted to use these for the sake of generating more complex design. Instead of manipulating the virtual model with the default modeling functions of the 3D software step by step, architects can now generate thousands of iterations of emergent outcomes with different sets of input variables fed into the same algorithm within a short period of time. This ability to discover novel emergent forms by harnessing computational power can be seen as a push in the right direction if compared to acts of modeling a pre-determined form. Influenced by scientific discoveries, such as system theory, complex science and genetic engineering, and the improvement of personal computational processing power, some architects began to execute well-known algorithms, such as L-System, Fractals, Subdivisions, Genetic Algorithms, Game of Life, etc., to generate complex geometric shapes related to their design concepts. However, in less than five years of development, a plethora of misuse of algorithms only for the sake of making complex desired forms to satisfy the personal desire of architects has become unprecedented. Such a trend of processing computational algorithms in architectural design fails to empower architecture in the digital age.

For digital architecture to be seen as a continuation of Deconstruction, computation should have a chance to challenge the stereotypical definition of architecture followed for thousands of years, not only in terms of "the appearance of form" but also its fundamental essence, which remains somehow missing in this category. The "**Form Generator**"/algorithms applied here should be capable of filtering excessive data into useful information to produce meaningful results not to be judged by aesthetic intuitive evaluation. Creating an algorithm should be seen as an inclusive part of the design in the process of the **Form Generator**. No matter how simple or complicated the algorithm is, the utilization of the algorithm should be embedded within contextual data to aid in the informed generation of form rather than being used as a tool for purely aesthetics driven Form Generation. John Frazer's quote in his work, "***An Evolutionary Architecture***", "**What we are evolving are the rules for generating forms, rather than the forms themselves**" (Frazer, 1995), adequately portrays the ideal definition of the **Form Generator**.

§ 3.3 FORM ANIMATOR

= A process wherein surrounding/contextual forces become instrumental in actuating the evolution of form to ultimately reach a dynamic equilibrium in real time.

Information undoubtedly involves a dynamic flow of data, so should **Form,** if interpreted as a representation of this information. Therefore, the **Form Animator** must have the ability to deal with the dynamic flow of information in real time. The **Form Animator** portrayed as a **real-time** adaptive information processor can drive its represented form as an actively morphing object following the rules of dynamic equilibrium. If this is still too abstract to understand, try to imagine a free-falling raindrop from the sky. it continuously morphs its form each and every minute based on its interaction with the surrounding forces and the principles of dynamic equilibrium, until it lands. The name of the **Form Animator** is derived from the well-known article "***Animate Form***" by Greg Lynn (Lynn, 1999), but the essence of the **Form Animator** is mostly pointing against what Greg Lynn states. Greg Lynn tried to make a distinction between "motion" implying movement and action, and "animation" implying the evolution of a form and its shaping force. But under the definition of **Form Animator,** in this research, these two categories of Motion and Animation involves different time-scales: a relatively slow morphological progress of growth or an immediate reaction via fast movement. In other words, a form, if seen as a mathematical representation, should consider relevant forces/parameters as dynamic variables to alter the resulting

shape through time. This, coincidentally, matches the central idea described in "***On Growth and Form***" by D'Arcy Thompson (Thompson, 1992)(Figure 3.4) long before Greg Lynn's "***Animate form***". As a digital pioneer, Greg Lynn, in "***Animate Form***", attempts to break the dominant cultural expectations from architecture, which implied Architecture to be static and permanent. This is achieved by utilizing computational topological modeling tools, such as B-Spline, B-Rep and Blob techniques with continuous mathematical relationships to search for an optimized/universal geometric solution: a continuous unibody dealing with all the potential vector forces around it. Although simulation engines work in advanced forms in contemporary CAD software, such as MAYA, it still seems inefficient to manually drag control points of a relevant B-Rep body to model an optimized form and to examine it back and forth using evaluation tools. The **Form Generator**, with the assistance of computational technology, can generate optimized solutions, given a set of parametric inputs and fitness criteria. But if we accept that "information is dynamic", to a certain extent, the input parameters for generating form could also acquire a dynamic nature. This is what the **Form Animator** takes into consideration and should be thus seen as dealing with dynamic parametric inputs, which are able to produce multiple optimized sets of results.

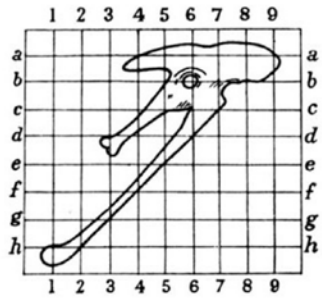

Fig. 161. Pelvis of *Archaeopteryx*.

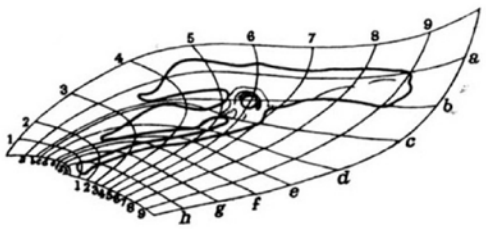

Fig. 162. Pelvis of *Apatornis*.

FIGURE 3.4 Analyzing the various morphology of animals using deformable grids by D'arcy Thompson (source: On Growth and Form, The Complete Revised Edition, New York: Dover Publications, Inc., 1992).

"**The problem with buildings is that they look desperately static,**" says Bruno Latour, who also further indicates that this is connected with the fundamental communicating medium, which architects use, namely, "**the drawings**" which are fixed to a particular perspective view illustrated in Euclidean space (Latour, B., & Yaneva, A., 2008). He further states that "**Euclidean space is a rather subjective, human-centered or, at**

least, knowledge centered way of grasping entities, which does no justice to the ways humans and things get by in the world". However, since architects work in Euclidean space, it becomes intuitive to ignore the fact that time and matter are actually married to space in a real living embodiment and not as static illustrations in the form of drawings. Even with current technologies to simulate render and animate, most building projects use them to merely portray lifestyle pertaining to how people adjust themselves to "happily" live inside the designed building, rather than using this data to take active action to influence spatial adaptation in a dynamic manner. The other issue Latour points towards is that since buildings have so many performative demands and considerations, that there is utterly no possibility to consider buildings as static artefacts which ultimately need to responsively transform with respect to internal and external forces they experience from the users and the environments they are embedded in real-time. He thus comes to the conclusion that **"...we should finally be able to picture a building as a moving modulator regulating different intensities of engagement, redirecting users' attention, mixing and putting people together, concentrating flows of actors and distributing them so as to compose a productive force in time-space**". This statement and its intent further points towards the essence of the next category, the "**Form Interactor**", which not only considers architecture as a one to one responsive system but also engulfs it within the domain of a collective intelligence based interactive system.

"**It is a liquidizing of everything that has traditionally been crystalline and solid in architecture.**" (Novak, 1991) Unlike Greg Lynn in Animate Form who realized static, yet complex architecture with existing topological modeling software, Marcus Novak proposed another vision of "**Liquid Architecture**" to liberate rigid architecture from the physical environment into Cyberspace (Novak, 1991). He argued that it is possible to envision architecture nested within architecture (Cyberspace), which basically proposed a co-existing environment where physical and virtual worlds bundle together. Within a virtual environment, architecture design can in a sense neglect the realistic physical constraints, such as gravity, but still, have the capability to deliver sensory perception via VR (Virtual Reality). Cyberspace is a virtual reality construct which smoothly liquidizes the hard boundary of physical space. To liquidize entities which have been crystalized in architecture is just the first phase of Marcus Novak's "**Liquid Architecture**", the ultimate goal is to adapt to real-time information flow and respond interactively to changing contextual data as an active living organism. Although this mode of conceiving architecture still involves an extensive amount of time to confront and resolve technical issues too, it still has the potential to ultimately change the dominant stereotypes of what architects could be doing. In contrast, it is quite disappointing, yet common, that most architects working in the digital realm do admit that working with dynamic information flow though does work at a theoretical and simulation level, but ultimately, they abandon this path to freeze the projects in

a static manner. Under the technical limitations of the 3D software back then, Greg Lynn stood up to the challenge to alter the fundamental essence of architecture from a computational perspective. His concept of "**Animate Form**", considering todays context of real-time information management, can now be re-interpreted and re-appropriated as "Animate Form (**Form Animator**)."

In a new interview: Pablo Lorenzo and Aaron Sprecher with Greg Lynn, documented in the publication "***Architecture in Formation: On the Nature of Information in Digital Architecture*** (Lorenzo-Eiroa & Lynn, Interview and projects by Greg Lynn FORM, 2013)", Lynn states: "**I had thought it was too simplistic and literal to reduce animation media to the role of designing moving projectiles and transforming objects. But, now I have to admit that a sensibility in culture is willing these moving environments into being. People expect their cities and buildings to literally move for a variety of reasons**", which to a certain extent is a modification of his former definition of "**Animate Form**." Based on Greg Lynn's re-interpreted notion of "**Animate Form**", Bruno Latour's, theories on liberation of building, and Marcus Novak's "**Liquid Architecture**", the **Form Animator** tends to inevitably operate more likely as a **Form Transformer.** This implies operating akin to a delicate mechanism constantly responding to input forces and actuating a relevant dynamic form. Moreover, the **Form Animator** can radically acquire the scope of a **Form Interactor,** which, not only passively react to direct environmental inputs, but can also pro-actively alter human and spatial behavior. Following this tendency, Latour's daring assertion of "**making all buildings move**" might actually come true.

§ 3.4 FORM INTERACTOR

= An emergent organic body composed of numerous singular intelligent entities possessing dynamic interaction. This dynamic interaction via internal/external information exchange can be seen during the process of growth and in the pro-active immediate behavior, which the body possesses.

Embedded in this immersive digital world surrounded by dynamic information flows, architecture has no excuse to keep with its static or essentially passive response state. It must be transformed into a living-creature-like entity which can react instantaneously and possess free will. It seems to be an inevitable trend that architects are yearning for making buildings as living organisms after adequate exploration of the **Form Animator**. Unlike the **Form Animator** principles, which are used for projects

which acquire algorithmically driven passive formal variations, the **Form Interactor** has an advanced proactive system akin to an artificial intelligence to make informed and immediate decisions compatible with dynamic data. The title of "**Form Interactor"** might at first seem misleading with the initial impression of merely focusing on the creation of an expressive phenotype, however, the "**Interaction**", is equally crucial for the creation of an implicit genotype, in an emergent fashion. In **Form Interactor**, the issue of interaction is different ways: "**Internal Interaction**", which, takes inspiration from biological growth processes, and "**External Interaction**", which mainly deals with immediate behavioral reaction, and both of these can be associated with the notion of "**Emergence**".

§ 3.4.1 Internal Interaction

Genetic Algorithms should not be seen as a process for optimizing form finding functions only: "Form Generator", but rather as an environmentally sensitive interactive process involving dynamic information flows: "Form Interactor".

The body, as a living entity, can be interpreted as a confluence of several complex systems interacting with each other akin to the multitude of systems which operate simultaneously to create architecture. During the growing process of an organism, there is, **Internal Interaction**, information embedded in genes as a basic instruction interacting with external factors from the environment to proportionally produce organic materials. This self-organizing process interested numerous pioneering architects to experiment with Genetic Algorithms, for form-finding purposes. These, however, turn out to be misleading examples, considering that the processes of real-time "Interaction" within natural growth processes tend to be completely missing during the computational processes of such algorithms. Michael Weinstock, in "***Morphogenesis and the Mathematics of Emergence***" (Weistock, 2004) clearly illustrated the generic computational approach of exploiting Genetic Algorithms in architectural design and other research fields, "**Genetic Algorithms initiate and maintain a population of computational individuals, each of which has a genotype and a phenotype. Sexual reproduction is simulated by random selection of two individuals to provide 'parents' from which 'offspring' are produced. By using crossover (random allocation of genes from the parents' genotype) and mutation, varied offspring are generated until they fill the population. All parents are discarded, and the process is iterated for as many generations as are required to produce a population that has amongst it a range of suitable individuals to satisfy the 'fitness criteria'**". Genetic algorithms undoubtedly enhance powerful computational applications supporting

the process of morphogenesis in architectural design. However, in most cases, Genetic Algorithms in architectural designs, based on defined fitness criteria are used for obtaining "Optimized", often static, outcomes for digital fabrication purposes. This, is contrary to the essential notion of "**growth**", which, is a real-time adaptive material producing "**process**". According to Micahel Weinstock (Weistock, 2004), "**Strategies for design are not truly evolutionary unless they include iterations of physical (phenotypic) modeling, incorporating the self-organizing material effects of form finding and the industrial logic of production available in CNC and laser-cutting modelling machines". This** illustrates the exact misuse of implementing Genetic Algorithms. However, still, a majority of architectural designers still use Genetic Algorithms specifically for producing aesthetically pleasing form without considering material performance and production logics. Genetic Algorithms directly implemented in architecture in this sense, act no more than an **algorithmic machine** akin to the role of the **Form Generator**, generating an optimized solution, opposed to John Frazer's idea to take natural science as a source of inspiration rather than explanation (Frazer, 1995).

John Frazer's idea of taking Genetic Algorithm as an **inspiration** implied not to directly execute these algorithms extracted from nature, but further, translate them into a design methodology for creating the instructions of the morphogenic formation in architectural design. **"...DNA does not describe the process of building the phenotype but constitutes instructions that describe the process of building the phenotype, including instructions for making all the materials, then processing and assembling them...These are all responsive to the environment as it proceeds...**". (Frazer, 1995). John Frazer emphasized in his significant article"***An Evolutionary Architecture***" that "**... what we are evolving are the rules for generating form rather than the form themselves. We are describing processes, not components**" (Frazer, 1995). This suggests that architects should design specific Genomes considering the context within which the design has to be embedded, rather than merely apply existing algorithms as a form-finding tool. Under the premise of John Frazer's rule-generating idea, the **Internal Interactions** of a living creature/building can be designed/interpreted as several internal information processing systems embedded in **Genomes** interacting with each other as well as external environmental inputs, forming a constant emergent mechanism for the overall growth "**process"**. The formation of the **Genome** is an on-going process with the inherited relationship of each cell that cannot be simplified as a one to one input-output mathematical formula neglecting the crucial fourth dimension, time, which is equal to the role of **Internal Interaction** implemented in the "**Form Interactor**". Common sense would state that "living" should be considered as an activity/state involving a continuous process involving constant data exchange between the body and its natural context, and thus can never be interpreted as an ultimate frozen state in time. If the **Form Interactor was** seen as a metaphor of a building, then it should also "live" in the existing environment rather than being

"located" or "crystalized" on site. "**Bones, for instance, which are full of living cells, can heal and adapt to their environment. In particular, the cells will rebuild the structure to adapt to the load it carries; a bone can change its physical shape after a fracture that heals out of position so that the load is adequately supported**" (Fox, Michael, & Kemp, Miles, 2009), the **Internal Interaction** within the example here reveals the vital ideas of real-time calculation, immediate adaptation and material interaction by distributed information systems amongst cells, to carry out the healing task, an which can be seen as an emergent behavior. Extending this healing function is associated back with the issue of "growth" and "fabrication", it is not that the **Internal Interaction** fights against the idea of fabrication, but the **post-optimised production** method is what the **Internal Interaction** refuses to accept in the section of the "**Form Interactor**". The ideal fabrication process within the concept of **Internal Interaction** should be akin to how an organism builds up its body based on the "Genome instructions" and "environmental influences" in real time. Each single moment is unique and with the summation of all internal and external forces emerging, the organism grows that particular body part based on each single task assigned to the living cells which cannot be repeated. This is exactly the emergent performance principles to be traced in **External Interaction**. (More Genetic Algorithm and Evolution Process will be discussed in the chapter of Bio-Inspired Architecture).

§ 3.4.2 External Interaction

External Interaction, following Swarm Behaviors-like principles, a dynamic equilibrium, should have capabilities to confront immediate circumstances locally to take action by individuals but interrelated componential intelligence agents and emerge from bottom-up as a global behavior to embody as a volatile actor.

To understand the issue of **External Interaction** in the **Form Interactor**, the notions of Emergence and **Swarm** Behaviour have to be introduced. "**...Emergence is applied to the properties of a system that cannot be reduced from its components. Properties 'emerge' that are more than the sum of the parts**", "***The Architecture of Emergence: The Evolution of Form in Nature and Civilisation***" (Weinstock, 2010), which simply and clearly explained the notion of **Emergence**. Michael, further quotes Aristotle's words to support this explanation, "**that 'whole' has distinctive properties that emerge through the processes of successive interaction between different levels of organization and integration**". It can thus be said that **Emergence** can be considered as a process of formation through interaction between different individuals systems/ entities, and the overall property of emergence cannot be observed by studying each

distinctive individual. Based on this definition, the **Internal Interaction** can definitely be seen as an Emergent Behavior, which merges several interactive interacting systems together to gradually develop the process of growth as a whole. In the case of **External Interaction**, the focus is the individual entity comprising the overall whole and the networked relationships between them. This idea of a larger property described by smaller componential entities can be traced back to the philosophical definition of a "Monad" in Gottfried Leibniz's Monadology back in 1714 (Leibniz, Monadology, 1714). The "Monad" here stands abstractly for the simplest substance which cannot be split apart and considered as a basic element comprising a composite object. As a result, in this respect, Leibniz made his point that "**In a plenum [= word that is full], any movement must have an effect on distant bodies, the greater the distance the smaller the effect…As a result, each body feels the effects of everything that happens in the universe, so that he who sees everything could read off from each body what is happening everywhere**". Therefore, every object, person, and every single matter existing in the world are all intimately interconnected to each other in this rapidly dynamic and hyperlinked Internet age. One can also connect, Emergent Behaviour to principles of Monadology, wherein every single monad, as a bird in a flock, has an influential interactive relationship with each other to emerge as a whole plenum(overall performative body) in a bottom up fashion. From the historical trajectory of these philosophic aspects, Leibniz's Monadology had great influence on Deleuze's thought process behind the "**Folding**" and "**Body Without Organs**" concepts, which profoundly impacted further philosophical inspiration in contemporary architectural design.

In Nature, **Emergence**, can be traced in the principles underlying **Swarm** behavior. **Swarm Behavior** principles embody numerous animal species, which tend to move collectively, for example, a flock of birds, a school of fish and a group of bees (Figure 3.5). Without any leader's top-down command, each individual forming in such groups of living entities make bottom-up decisions, resulting in bigger collective behavior. Each entity is equal, in stature, to each other and thus any of singular entity's movement/decision, profoundly impacts the overall performance of the whole. This characteristic fits perfectly with the science of **Emergence** as well as Leibniz's **Monadology** philosophy. After observing flocks of flying birds, Craig Reynolds, as a computer scientist started to develop swarm behavior simulation back in 1987 (Reynolds, Flocks, herds and schools: A distributed behavioral model, 1987). Three major principles of "**Separation**", "**Cohesion**", and "**Alignment**[29] underlying the steering

29 Separation implies avoiding crowding next to each other; Alignment implies steering towards the average direction of the neighboring flocks; Cohesion implies driving the agents' movement towards the average position of the local agents. More information can be found referring to Craig Reynold's website illustrating the flocking behavior: http://www.red3d.com/cwr/boids/

behavior behind a digital flock of birds was thus successfully realized in a virtual environment with intuitive and smooth movement. Since then, this Swarm behavior algorithm has been broadly applied to different paradigms of research including game design, swarm robotics, distribution and communication systems...etc., and certainly in architectural design applications as well. John Holland, another pioneer working on **Emergence** and genetic algorithms, pinpointed three major principles required to set up a basic Emergence system: **element**, **rules**, and **interactions** (Holland, 1998). If we follow John Holland's proposal, the three major principles of **Swarm behavior** simulation developed by Craig Reynold can be accordingly modified by further enhancing the fundamental principles in relation to implementation/task based deploy ability. Computationally speaking, one can modify the basic principles/ rule sets of agent interaction, in order to develop customized "**basic intelligence**" within the algorithm associated with smarm simulations. Recently, architects have taken advantage of growing computational power for developing **Swarm based design systems**, as novel approaches in architectural design.

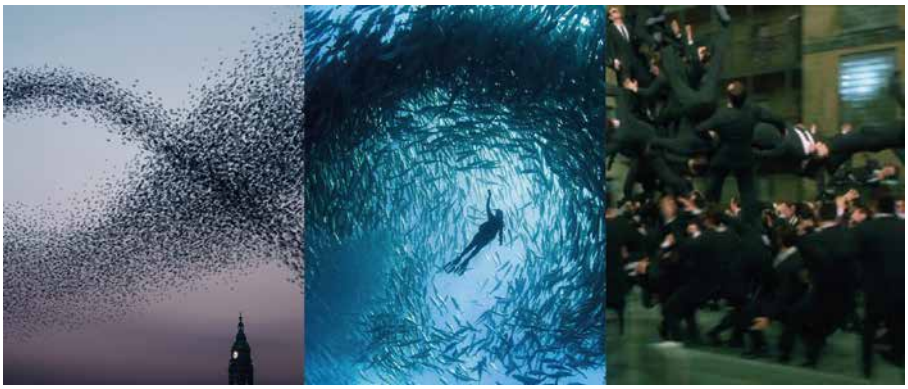

FIGURE 3.5 Images exhibiting the swarm idea either in nature or in the film. A swarm is a group of animals that aggregate and travel in the same direction(https://en.wikipedia.org/wiki/Swarm_(disambiguation)). From left to right: a swarm of insects, a school of fish, a group of agent Smiths in the Matrix (source from left to right: http://www.ayni.institute/swarm, http://www.dailymail.co.uk/news/article-2834570/Divers-caught-middle-huge-school-fish-snap-selfies-them.html, and http://movies.stackexchange.com/questions/27942/is-there-a-trope-for-a-pile-on-fight).

Roland Snooks, one of the leading characters in this new domain, also one of the directors of **Kokkugia** has for years conducted experiments using **Swarm** algorithms for promoting self-organization principles in architectural design, under the title "**Behavioral Formation**," which is also the title of his Ph.D. dissertation in RMIT, Melbourne. Some of the experimental designs were developed together with his design partner, Robert Stuart-Smith in practice, and some with his master students

both at the AA, London, UK and RMIT, Australia as design research experiments. Roland Snooks' idea of a self-organized body within **Swarm behavior** principles is explicitly illustrated as follows, "**These methodologies operate by encoding simple, local architectural decision within a distributed system of autonomous computational agents. It is the interaction of these local decisions that self-organizes design intention, giving rise to a form of collective intelligence and emergent behavior at the global scale. Such behavioral formation represents a shift from 'form being imposed upon matter', to form emerging from the interaction of localized entities within a complex system**" (Snooks, 2013). In other words, the behavioural formation can be interpreted as a self-organizing system constituting agents of a swarm, which produce unique/local material properties due to underlying collective decision-making principles set forth by the designer. Akin to the aforementioned concept of **Internal Interaction** constituting the process of growth or similar to the process of self-healing of living bone cells. However, even volatile topology has been heavily addressed in Roland Snook's concept with Swarm logics, as all his computational generative formation processes are frozen in a particular moment, which, is fundamentally against his original idea of "**volatility**". Various young digital savvy architects are extremely fascinated by this emergent behavior and its capability and have started following this trend of executing swarm algorithms in architectural design again as a form/pattern finding process. Roland Snooks' approach of utilizing swarm algorithms is still, in general, in a relatively initial stage. Although he advanced the development of algorithms for making local collective decisions to materialize creative projects, he somehow overlooked the inherent character of swarms, which, points towards a **continual dynamic process**, which cannot be crystallized at any moment in time. For instance, in nature, simply taking groups of ants for example, they can form an emergent holistic body such as a bridge, helping each other to cross a pond of water or gap between leaves. However, once the temporary goal is reached, they will re- distribute themselves going back to doing their own tasks and form new configurations according to the new tasks they need to accomplish in time.

To a certain extent, we can still interpret this as another much-advanced version of the **Form Generator/Animator** due to the fact that it remains frozen in its ultimate state, which, makes his projects less commensurate to the terminology of "Swarm Architecture". Swarm Architecture, in its true sense, should possess the substantial potential to deal with immediate interactions similar to how living entities adapt to dynamic contextual demands. "Swarm Architecture" based research should thus be highly advanced in order to produce intelligent buildings with capabilities of real-time adaptation and interaction. This is the ideal goal for what **External Interactions** should embody in a "**Form Interactor**".

"**Space is a computation**." Kas Oosterhuis made this bold and strong assertion in the very beginning of his article "***Swarm Architecture II***" (Oosterhuis, Swarm Architecture II, 2006), which was proposed years before Roland Snooks presented his Behavioral formation idea. According to Kas Oosterhuis "**space computes information**". This links perfectly with the key concept of this chapter: to consider **form(space) as an information processor**. Following Kas Oosterhuis' steps, an architecture can also be seen as a networking instrument communicating actively with the users of the space in real-time via various inter-connected actuated building components, "**The actuators are being orchestrated like the birds in a swarm**", Kas concluded. Kas Oosterhuis thus proposed the idea to bring computational technology for practical usage by embedding it into building components for active internal communication and external adaptation instead of utilizing the computing power merely as a form generating tool. This mode of thinking perfectly embodies the authentic intent of the "**Form Interactor**." This intent can further lead to the production of buildings, which, in essence, become alive and thus a species in their own right. This is further reinforced by, John Frazer's statement:"**We never try to copy the superficial appearance of a biological species. Rather we try to invent new species which by its complexity and due to their complex behavior may eventually familiarize with living objects as we already know**" (Frazer, 1995). It is time to shift towards utilizing computational power to develop practical operational spatial solutions rather than for creating front-end form generating machines. In other words, it is time to utilize the principles of "**Swarm behavior**" as the fundamental basis behind "**Form Interactor**" to develop a novel approach for integrating computational technologies within building component for developing a networked distributed system for realizing an architectural body which can adapt to its immediate context.

§ 3.5 Conclusion

In this chapter, "**Form**" has been interpreted as an information processor inspired by Kas Oosterhuis' "**Space is a computation**" approach. Actually, in every scale, all existing objects are to a certain extent related to information which can be translated and represented in diverse forms. Simply take a small device like a pen, for example, it has information embedded associated to its dimension, color of ink, and material it is made of. Furthermore, with its essence of being a pen, it has a given function of making traces. This kind of "Object-Oriented" concept is mainly utilized in computer science to illustrate a category, constituting certain characteristics, where you can generate objects from its essence, but vice versa it can logically categorize any existing

object with a similar principle. In the introduction section of this chapter, it is clearly emphasized that people have dealt with spatial information long before the computer had been invented, the only crucial difference is that the computational technology accelerated the processing of data. In the digital architecture domain, the means and degrees of utilizing computational technology have been categorized into different sections in this chapter, namely: **Form Sculptor**, **Form Generator**, **Form Animator** and **Form Interactor**. Not only the manner but also the philosophy and the logic of a computational application in architectural design behind them have been reviewed in this Chapter in order to trace the advantages and disadvantages within each category. There are definitely "pros and cons" but no "rights or wrongs" of these formative approaches from the design perspective, it is only a question of the methodologies and strategies the designer prefers. The **Form Sculptor** tends to favor a more intuitive approach compared to the **Form Generator** relying on rational algorithms as a form-finding method. The **Form Animator** starts to be aware of the influential impact from dynamic information flows, while the **Form Interactor** takes the dynamic information into account as either the slow morphing process, in the case of growth or immediate morphing process, in the case of an immediate reaction. **Form thus has** an intimate relationship between the architectural design process and contextual information.

Based on what Stephen Wolfram has stated in "***Towards a New Kind of Science***", **"...nature[the Universe] as we know it is a pure form of computation**" (Wolfram, 2002), it is extremely rational to claim that "**Space is a computation**" as proposed by Kas Oosterhuis in 2011. The other crucial factor of "**dynamic equilibrium**", indicates the need to be constantly changing/evolving with information flow, and that this will drive architectural design to acquire the dimensions of a living organic body. "**Liquid Architecture is an architecture that breathes, pulses, leaps as one form and lands as another...It is an architecture that opens hallways, where the next room is always where I need it to be and what I need it to be**" (Novak, 1991) noted Marcus Novak who proposed a volatile architecture operating as a living creature almost 15 years ago. During the same period of time, Kas Oosterhuis has even put this living architecture idea to the next level with an architecture that actually has its own will by proposing the **HyperBody** concept, "**True hyperbodies are pro-active bodies...actively propose actions. They act before they are triggered to do so. HyperBodies display something like a will of their own. They sense, they actuate, but essentially not as a response to a single request**" (Oosterhuis, HyperBodies: Towards an E-motive Architecture, 2003). In Kas Oosterhuis' mind, the way of constructing this intelligent architectural body with free will is not by complicated AI(Artificial Intelligence) system, but instead, by using Swarm logic as a system which thrives on collective intelligence. "**Think of a type of architecture where all building elements are intelligent agents flocking the herd, (re) configuring themselves in real time**" (Oosterhuis, HyperBodies: Towards an E-motive Architecture, 2003), this (re)configurable body can achieve real-time interaction

with relatively smaller entities with simple intelligence. In this case, computation is no longer seen as a form-finding machine which generates a nearly optimized, fixed architecture, but is embedded in building components which can communicate through protocols and to a certain extent actuate/react akin to living cells in an organic body.

References

Cache, B. (2012). Instruments of Thought: Another Classical Tradition. In C. O'Donnell (Ed.), *The Cornell Journal of Architecture 9: Mathematics* (pp. 17-26). New York: College of Architecture, Art, and Planning, Cornell University.

Carpo, M. (2012). Twenty Years of Digital Design. In M. Carpo (Ed.), *The Digital Turn in Architecture 1992 - 2012* (pp. 8-14). New York: Wiley.

Dürer, Albrecht & Formschneider, H. Andreas (Nürnberg). (1525). *Underweysung der Messung* . German: Nürnberg.

Fox, Michael, & Kemp, Miles. (2009). *Interactive Architecture.* New York: Princeton Architectural Press.

Frazer, J. (1995). A Natural Model for Architecture/ The Nature of the Evolutionary Mode. In J. Frazer, *An Evolutionary Architecture.* London: Architectural Association.

Holland, J. H. (1998). *Emergence: From Chaos to Order.* Oxford: Oxford University Press.

Latour, B., & Yaneva, A. (2008). Give Me a Gun and I Will Make All the Buildings Move: An Ant's View of Architecture. In R. Geister (Ed.), *Explorations in Architecture: Teaching, Design, Research* (pp. 80-89). Basel: Birkhäuser.

Leibniz, G. W. (1714). *Monadology.* (J. Bennett, Trans.) Continuum. Retrieved from http://www.earlymoderntexts.com/assets/pdfs/leibniz1714b.pdf

Lorenzo-Eiroa, P. (2013). Form:In:Form on the relationship between Digital Signifiers and Formal Autonomy. In P. Lorenzo-Eiroa, & A. Sprecher (Eds.), *Architecture in Formation: On the Nature of Information in Digital Architecture* (pp. 11-22). New York: Routledge.

Lorenzo-Eiroa, P., & Lynn, G. (2013). Interview and projects by Greg Lynn FORM. In P. Lorenzo-Eiroa , & A. Sprecher (Eds.), *Architecture in Formation: On the Nature of Information in Digital Architecture* (pp. 286-295). New York: Routledge.

Lynn, G. (1999). *Animate Form.* New York: Princeton Architectural Press.

McCullough, M. (2006). 20 years of scripted space. (M. Silver, Ed.) *Architectural Design Special Issue: Programming Cultures, 76*(4), 12-15.

Mitchell, W. J. (1990). A New Agenda for Computer-Aided Design. In M. McCullough, W. J. Mitchell, & P. Purcell (Eds.), *The Electronic Design Studio: Architectural Education in the Computer Era* (pp. 1-16). Cambridge: The MIT Press.

Novak, M. (1991). Liquid Architectures in Cyberspace. In M. Benedikt, *Cyberspace: First Step* (pp. 225-255). Cambridge: The MIT Press.

Oosterhuis, K. (2003). *HyperBodies: Towards an E-motive Architecture.* Basel: Birkhäuser.

Oosterhuis, K. (2006). Swarm Architecture II. In K. OOsterhuis, & L. Feireiss (Eds.), *The Architecture Co-Laboratory: Game Set and Match II, on Computer Games, Advanced Geometries, and Digital Technologies* (pp. 14-28). Rotterdam: episode publisher.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *Compute Graphics, 21*(4), 25-34.

Snooks, R. (2013). Self-Organised Bodies. In Lorenzo-Eiroa P., & A. Sprecher (Eds.), *Architecture in Formation: On the Nature of Information in Digital Architecture* (pp. 264-267). New York: Routledge.

Sutherland, I. E. (1963). *Sketchpad: A Man-machine Graphical Communication System.* Cambridge: University of Cambridge.

Terzidis, K. (2006). *Algorithmic Form.* Oxford: Routledge.

Thompson, D. (1992). *On Growth of Form.* London: Cambridge University Press.

Weinstock, M. (2010). *The Architecture of Emergence: The Evolution of Form in Nature and Cilvilisation*. New York: Wiley.

Weisberg, D. E. (2008). *The Engineering Design Revolution: The People, Companies and Computer Systems That Changed Forever the Practice of Engineering*. Retrieved from www.cadhistory.net

Weistock, M. (2004). Morphogenesis and Mathematics of Emergence. In M. Hensel, A. Menges, & M. Weinstock (Eds.), *Architectural Design, Emergence: Morphogenetic Design Strategies, Volume 74, Issue 3* (Vol. 74, pp. 10-17). New York: Wiley.

Wolfram, S. (2002). *A New Kind of Science*. Champaign: Wolfram Media. Retrieved from http://www.wolfram-science.com/nksonline/toc.html

Young, M. (2012). Digital Remediation. (C. O'Donnell, Ed.) *The Cornell Journal of Architecture 9: Mathematics*, 119-134.