Dynamic Descriptions: Steps Towards a Design Machine

by

Varvara Toulkeridou

Diploma in Architecture and Engineering,
Aristotle University of Thessaloniki, Greece, 2007.

Submitted to the Department of Architecture
in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Architecture Studies

at the

Massachusetts Institute of Technology

September 2010

Signature of Author .....

Varvara Toulkeridou
Department of Architecture
August 9th, 2010

Certified by .....

George Stiny
Professor of Design and Computation
Thesis Advisor

Accepted by .....

Takehiko Nagakura
Associate Professor of Design and Computation
Chair of the Department Committee on Graduate Students

Thesis Advisor:
George Stiny, PhD
Professor of Design and Computation, M.I.T

Thesis Reader:
Takehiko Nagakura, MArch, PhD
Associate Professor of Design and Computation, M.I.T

Dynamic Descriptions: Steps Towards a Design Machine

by

Varvara Toulkeridou

Submitted to the Department of Architecture
on August 09, 2010 in Partial Fulfillment of the
Requirements for the
Degree of Master of Science in Architecture Studies

ABSTRACT

This thesis questions which would be a valid approach for building design machine aided by computational intelligence capable of generating surprises for their designers-observers. There have been efforts since the 1960s towards developing frameworks for design machines that were envisioning computational systems as something more than tools for efficient production and representation. Some of them were dealing with design problems as complex systems that needed to be broken down in modular parts, for example Christopher Alexander's "Notes on the Synthesis of Form". However such strategies were associated with explicit languages of descriptions and strong hierarchies, defined in advance by the designer, that were constraining the design space to what these predefined descriptions were anticipating.

This thesis draws its motivation from the work of Professor of Design and Computation George Stiny on visual computations operating on non-fixed sets of primitives, as well as from research conducted in the field of Artificial Intelligence on alternative representations. I will propose a framework for a design machine highlighting the importance of it being able to generate its own dynamic descriptions, "entities" that bear content independent of the interpretations of their designers. Inspired by a computational system, developed by Stephen Larson (2003), capable of grounding its own symbols in perception, I will experiment with self-organizing map algorithms suggesting them as a possible way for a design machine to build up and update its language of description from its perceptual information.

Thesis Supervisor: George Stiny
Title: Professor of Design and Computation

Thesis Reader: Takehiko Nagakura
Title: Associate Professor of Design and Computation

Acknowledgements

I am deeply indebted to the following individuals and institutions for their support and inspiration:

My advisor, George Stiny for guiding me through my research process and for letting me free to explore my own research path; his work provided me with a valuable insight into design and computation.

Takehiko Nagakura for being a supportive reader.

Patrick Winston for his visionary approach and insight to intelligence; his class significantly formed the course of my research.

The Foundation for Education and European Culture (IPEP), the Konstantinos Katseas Fund and the Department of Architecture of MIT for their financial support without which my studies at MIT would not have been possible.

Kaustuv De Biswas for the accurate critiques and inspiring dialogue;
both played an important role on the development of my research.

All the Design & Computation SMArchS and Ph.D. friends, among them Skylar, Steffen, Asli, Alexandros, Adela, Shani, German, Mark and Murat for their comments and feedback; life at MIT would not be the same without them.

This thesis is dedicated to my parents.

Table of Contents

Introduction

[a] How everything started

I was always attracted to design examples that were experimenting with form. I was impressed with the novelty hidden in the ability of the designers to look at everyday things in a different way. For example, what inspires a designer to project in a simple folded paper the concept of a continuous architectural envelope?

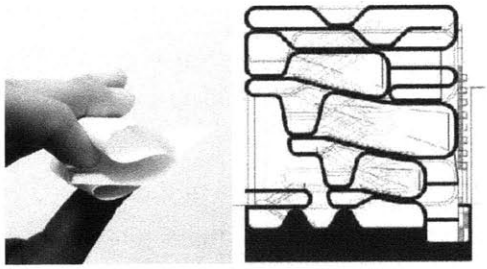


Figure 1: New ways to look at things

Sometimes I caught myself looking around, searching for elements, patterns, and formations in my everyday environment that could motivate me to conceptualize form in a different, novel way. Soon I carried this expectation into my interaction with the digital design tools I was using. I could run a small script and then observe what kind of transformations occurred; sometimes the results were beyond my expectation and in fact I could identify in the generated visual forms "new" ways to inhabit space.

One of my recent projects, during which I was developing an algorithm to generate spiraling patterns, fed my curiosity. I started questioning myself: why, instead of directly modeling a surface in the 3d environment where I could apply my spiraling generator script, did I prefer to play with random number patterns that would drive my surface geometry? Was this making me less of a designer? Was I handing over to the computer the agency to design instead of me?

In retrospect, I think I was trying to convert my script to a "black box." I wanted to introduce to the system elements beyond my understanding that would generate unexpected results, different from my sometimes fixed patterns of knowledge and thought, results that would motivate me to recognize new formal relationships.

[b] Thesis outline

The body of this thesis is formed by four sections. The greater question that runs through these four sections is whether and how our digital computational design "partners" could be generators of surprises for their designers-observers.

There have been efforts since the 1960's towards developing frameworks for design machines that were dealing with design problems as complex systems that needed to be broken down in modular parts with the objective to develop new methods of design and gain a better understanding of the design process. Other efforts were aiming at building architecture machines aided by computational intelligences that could facilitate a creative human-computer interaction. Section 2 briefly traces some of these efforts and discusses why some of these approaches were against the nature of design and against the emergence of novelty.

Section 3 argues that surprise is a driving force in the design process. It discusses the concept of a "black box" machine and forms the hypothesis that by regarding our design machines as black boxes we set the conditions for them to generate surprises that will motivate the design process. By highlighting some fundamental characteristics of the design process, at the third part of the section, a computational framework for a design machine that introduces surprise as its criterion is developed.

Section 4 outlines a classification of machines, discussing the concepts of a receptor machine, an effector machine, a reaction machine, a memory machine and finally a machine that forms intrinsic representations. It questions the "eligibility" of each of them as a possible design machine concluding that intrinsic descriptions are a step that can take the concept of a design machine to a higher level.

Section 5 moves a step forward towards a machine that can form intrinsic descriptions by inquiring how a machine can algorithmically decompose a phenomenon into discrete parts. Revisiting the research work of Stephen Larson I propose the use of self-organizing maps (SOM) algorithm as the infrastructure for the design machine. I demonstrate an introductory example run of an SOM algorithm built to organize fragments of a two dimensional input image.

PART **01**

[a] Historical background

Since the appearance of computer-aided design in architecture in the 1960s, there have been efforts towards developing frameworks for design machines, which envisioned computational tools as something more than simple tools for efficient production and representation. There has been a critical mass of people, who saw in digital computational tools the possibility of developing new methods of design. Some were intrigued by the chance of figuring out what was going on in the earliest design phases, in order to further improve and support the design process. Others, considering design simply as a form of information processing, were seeing computers as the mean to develop techniques for dealing with great amounts of information and for ordering and reordering it into useful subsets of manageable data.

A characteristic example is Christopher Alexander's (1964) book *Notes on the Synthesis of Form*, on a systematic design method based on decomposing a complex network of interacting design requirements into independent subsystems. Alexander and its research group regarded the decomposition of complicated design issues in a systematic way, not only as a chance to solve a design problem but also as a source for producing innovative solutions (Milne, 1975, p. 32). Some years later, trying to overcome some of the inconsistencies and mistakes of Alexander's programs Murray Milne built CLUSTR, a computational system meant to assist the designer at the beginning of the design process "in finding the structure inherent in his design problem" (Milne, 1971, p.242). Once the designer has supplied to the system the elements that define his design problem and their relationships, the computer generated reorgani-

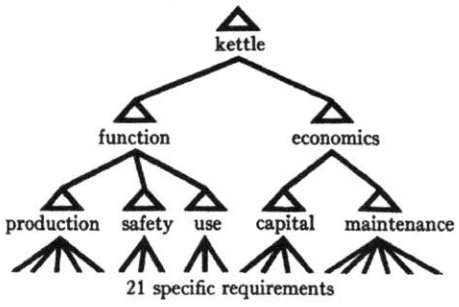
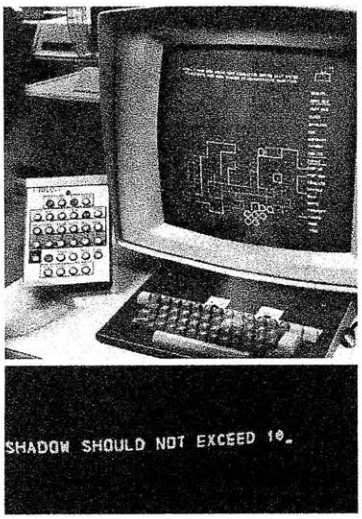


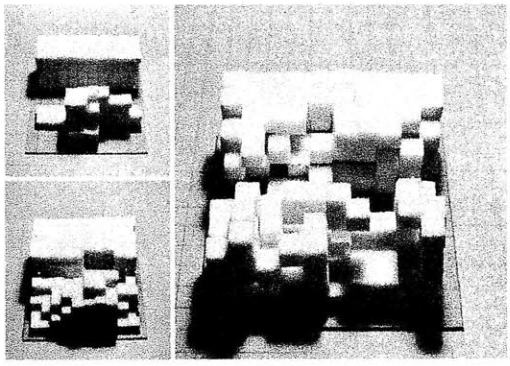
Figure 2:
Alexander - Decomposition of the design problem in a systematic way

zations of these elements. Milne (1975) built CLUSTR with the aspiration that the reorganization generated by the system would provide a solution that for the designer "would be completely new" (p. 35).

Another important contribution to the research on design machines was the work of the Architecture Machine Group, founded in 1967 at MIT by Nicholas Negroponte. The group was focusing its research efforts on "architecture machines", a term allocated by Negroponte (1970) to an artificial intelligence that "assisted, augmented and eventually replicated" architectural design processes. Negroponte (1970) in his book The Architecture Machine: Toward a more Human Environment states that computer-aided design cannot occur without machine intelligence. He envisions architecture machines that can be active partners, intelligent, capable of learning and improving over time; able to solicit information on their own, to acquire experiences and to understand human idiosyncrasies. They should be able to interrupt the designer's train of thought and to take initiative to do things as "remind, stimulate, reprimand, caution, or even abort action" (Negroponte, 1970).



< Figure 3: URBAN5 Architecture Machine Group
Snapshot of the conversation between the designer and the machine.



^ Figure 4: LEARN, Architecture Machine Group
The configurations of cubes on the left were two of the archetypes input to the system, the one on the right is a solution generated by the system.

The group implemented various design systems - LEARN, MEMORY, GROPE, URBAN5 to name a few. LEARN was a "computer mannerist"; it observed the designer's activities and was generating its own solutions. MEMORY was an information storage and retrieval system with "forgetting convenience". Over time the system could classify events either as strong remembrances or fainter recollections. Consequently, as time passes the responses of the system were gaining meaning with respect to its input. URBAN5 was a design machine that was conversing with a designer on an environmental design project. The system through a central attention mechanism "listens" to the designer, always giving him the opportunity to change his mind or restate a situation at any time.

I will close this brief historical survey with the framework for an autonomous system for creating designs, outlined by George Stiny and Lionel March (1981) in their paper "Design Machines". Having as a starting point Franz Rouleaux's argument that design (invention) is thought, they elaborate Kenneth Craik's schema for thought into a schema for design. According to Craik (1943) thought involves three essential processes: "translation" of external processes into words, numbers or symbols, arrival at other symbol's by process of reasoning, deduction, inference and "retranslation" of these symbols into external processes. Their framework divides the design process into four mechanisms: receptor, effector, language of designs, and design theory. The receptor creates representations of external conditions and the effector stimulates external processes or artifacts from designs.

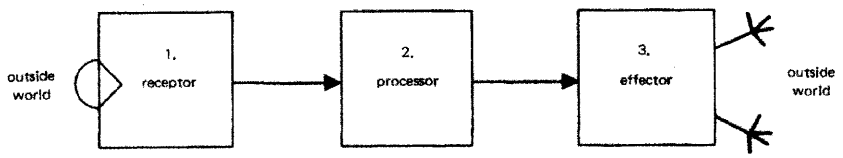


Figure 5: Kenneth Craik's schema for thought, base for the proposed framework for design machines by Stiny & March.

[b] Framing the problem

> Computers have come to play an important role in architectural practice. Nonetheless, the promise of computation as a creative partner in practice, and a means to better understand and support the design process has yet to be realized.
>
> George Stiny, Terry Knight (2001)

Christopher Alexander and others who approached the study of design methods through the use of computational systems, started with the hope that by writing computer programs they could capture the moment of novelty and creative insight a designer experiences in the design process. However, the experiments on systematic design methods included strategies associated with explicit languages of descriptions and strong hierarchies that were constraining the design space to what these predefined descriptions were anticipating.

Relevant discussion can be found in Ivan Sutherland's paper "Structure in Drawings and the Hidden-surface Problem". Sutherland (1975) reflecting on Sketchpad, the first computer-aided design program, writes that to a large extent the usefulness of computer drawings is their structured nature (p.75). However the need for structure and hierarchy stands in opposition to the nature of the design process. Stiny (2006) in agreement with Sutherland's point writes:

The difficulty in making computer drawings is their structured nature. It's hard - no it's impossible to tell what constituents to draw without foreknowledge. That's the trick - not to need foreknowledge so that the properties of computer drawings can alter freely to correspond with the properties of anything they're used to describe. (p. 135)

The designer looking for potential values during the design process is free at every step to decompose and reorganize anew the elements that characterize his evolving artifact. There is no structure or hierarchy for him to remember; there is no fixed representation and vocabularies. For Sutherland (1975) a designer is concerned only with drawings as the representations of the evolving

design, as "dirty marks on paper" (p. 75). On the other hand, the behavior of a drawing produced in a computer is "critically dependent upon the topological and geometric structure built up in the computer memory as a result of drawing operations" (p. 75).

Since Sketchpad, a wide range of computational tools has been developed that successfully address isolated aspects of design like analysis, fabrication, evaluation etc. This segmentation of the design process reveals the constraints of our digital design tools as far as the early stage of design is concerned, while the accuracy and speed of computers can be efficiently exploited for the later design phases.

It has to be mentioned though, that during the last two decades, there have been some new efforts to implement digital computational mechanisms in the early design stage. Such an example is the use of genetic algorithms in which a host of outside criteria not given directly in the rules of the algorithm plays an important role in the outcomes (Knight & Stiny, 2001). Knight and Stiny (2001, p.360) classify this type of computation as non-classical in process because the rules on which it operates are not fully understandable. Very often the outcomes of such processes are observed as unexpected and surprising by their designers. However even these cases still involve the need from the side of the designer to fix his way of "seeing" and explicitly define in advance the vocabularies of the computation. From that point on the machine just searches and generates solutions.

To sum up, what this thesis identifies as a problem is the need in a digital computation for the designer to fix a priori his primitive units of descriptions and to define the hierarchy by which these primitives are related. In other words, he should have understood very well how the computation will operate. But then there is no space left for surprises. Designers want to regard their computational design tools as creative partners in design, as open processes that generate opportunities for ambiguity and surprises. Francois Roche's (2009) quote is indicative of this desire:

> Machines are always pretending to do more than what they were programmed to do. It's their nature. Their behavior alternates phantasms, frustrations and fears inspired by their own ability to break free and threaten us...These multiple disorders, this kind of schizophrenia, could be considered a tool for reopening processes and subjectivities, for re protocolizing indeterminacy and uncertainties. Misunderstandings, in this sense, produce artifacts ... and apparatus can be considered generators of ambiguity and knowledge where non shaping protocols, protocols that emerge, contingently reveal the conditions of emission ...

If it is to envision digital design systems as generators of surprises and unpredictable results, we have to understand that it is necessary even for us that design them to regard them as black boxes. By the end of the thesis I will have demonstrated that a possible approach is building design machines able to sense their environment and to build their own descriptions out of their perceptual information.

Before moving to the next chapter, I would like to clarify that the decision to research on autonomous design systems did not originate from the belief that the designer should be removed out of the loop. It rather adopts Yona Friedman's standpoint that a "machine" does not become a machine except because of the user. No "machine" could be imagined that does not "contain" an intelligent observer (Negroponte, 1975, p. 93-94). Friedman mentions:

> I do not consider the "hardware" machine (or even the "hardware + software" machine) as the machine. I consider as "machine" only and exclusively a system containing "the machine and me".

His arguments signify a "machine" composed of two "submachines". The first is the "real world and the computer" and the second "the user and the computer". This thesis' interest lies on the first of the two. It looks for the properties that would constitute this submachine capable of generating the conditions for the observer-designer to encounter surprise.

PART 02 Surprise as a design criterion

This section comprises of three chapters. The central theme of the section is the element of surprise. The first chapter inquiries how surprise can be introduced as a criterion in a design machine by exploring the concept of a "black box". The second chapter attempts to explain why surprise is a driving force in the design process. Having highlighted the importance of surprise in design, the third chapter identifies the singular characteristics that would allow a design machine to act as a "black box", thus as generator of surprises. By the end of part 3 I will have outlined a framework of a design machine which includes surprise as its design criterion.

[a] Defining the "black box"

What is behind the curtain?
Ludwig Wittgenstein

Marvin Minsky (1967) uses the term black box to describe how a machine can be regarded from the point of view of the user or the environment as a closed box with input and output channels. From time to time the user acts on the machine through the input channels, and from time to time the machine acts on the user through the output channels. The user doesn't normally need to know just what really takes place inside the box. "That is, unless he is particularly interested in understanding the "works" of the machine, or in modifying it, he needs to know only what are its "input-output" properties" (Minsky, 1976, p.13).

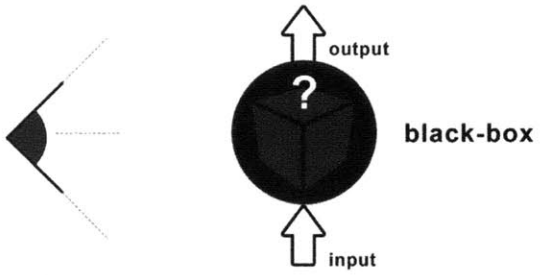


Figure 6: The machine as a black box to the outside observer

This thesis is based on the hypothesis that by regarding our design machines as black boxes we set the conditions for them to generate surprises that will motivate the design process. To understand how a "black box" can creatively contribute in an interaction we have to think this interaction in terms of a conversation. According to Ranulph Glanville (1997) conversation is a mechanism that allows us to communicate through ignorance – ignorance of what a person is thinking and of how that relates to their chosen form of representation.

Let me clarify the argument through an analogy used by Michael Reddy (1993, p.171-176), called the toolmakers paradigm. The toolmakers paradigm starts from the assumption that people are living isolated in slightly different environments. He depicts this situation as a wagon wheel where each pie-shaped sector represents a different environment. The two spokes and a part of the circumference form the walls of each environment. There is no way for the people to visit each other's environment or exchange samples of the things they construct. The only means of communication between them is a device that exists at the hub of the wheel by which they exchange crude sets of instructions on how to make things helpful for living. In this analogy the content of each environment represents each person's unique set of thoughts, feelings and perceptions. In the communication described in this paradigm, very often partial miscommunications and divergence of readings occur. However, the constant interaction and the effort provided by people to communicate and exchange information on their different backgrounds result in important progress in the quality of their living.

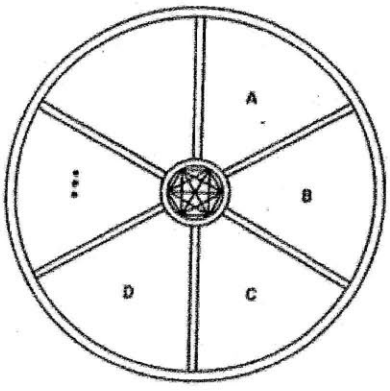


Figure 7: The toolmakers paradigm
A, B, C and D are living in isolated environments and communication is allowed only through the device at the hub of the wheel

The precondition in Reddy's paradigm for these miscommunications to occur is the "isolated" environments in which each person lives. What is going on in the mind and the world of person A is black box for person B; A and B stand at a "distance" in terms of how they perceive and understand the world. This distance is what makes it possible for person A or person B to receive a response that will offer different interpretations, variations or even produce surprises –unexpected outcomes that do not fit in their categories of knowing and patterns of thought.

The distance among the two parts is what Glenville (1997) calls the interface that allows us to remain ignorant yet still to interact. He writes that everything that has happened has happened between the observer and the black box and nothing is entirely of the one or the other. The communication is in the between. Moving back to the realm of design and our digital computational tools, when the designer writes code he assures that the computational system will "see" his own understanding of the world; therefore the in between distance shrinks. The vocabularies and structure that drive the computation have to be made explicit in advance. No "space" is left for misinterpretations and surprises.

If it is to envision digital computational systems as creative partners in design, we have to understand that we have to build the conditions for them to act as generators of surprise.

{b] Positioning surprise in design

I believe that surprise is a driving force in design. I designate the moment of surprise in the design process when the designer encounters the "unexpected"; in other words, the moment when in the eyes of the designer, the appeared form comes to be different from the expected form. By expected form, I mean any current organization of matter and information - described by the defining constraints and instantaneous forces that control the evolving artifact - that is either known or can be easily predicted by the designer-observer. Surprise resides in the appearance of reorganization and the emergence of

features that stand in opposition to the artifact's current defining character. Donald A. Schön (1987, p.28) defines as surprise an unexpected outcome, pleasant or unpleasant, produced by routine responses, that does not fit the categories of our knowing-in-action. Inherent in a surprise is the fact that it gets our attention.

The aspiration of being surprised by the "new" and unexpected leads the designer to a successive reinterpretation and restructuring of the design representations, which moves forward the design process. He is constantly seeking potential values in the evolving design while trying to balance a number of implicit or explicit criteria, looking forward to the moment of the creative leap. This moment of creative insight is defined by Murray Milne (1975, p.32) as "the sudden and spontaneous reorganization of previously dissimilar elements into an integrated whole, which the designer believes is different from everything else he has known before."

The designer is involved in a fluid and spontaneous mode of production; what constitutes this process as dynamic and fluid is the designer's freedom at any step to distinguish a new potential value, change his mind and focus on something else. This reinterpretation and restructuring of design representations is a fundamental property of design thinking. This property is facilitated by the interaction between the designer and the visual representations of the design artifact. When for example a designer sketches, he awaits the emergence concepts, ideas, and things with potential values. Designers therefore have a dynamic relationship with their sketches, where each perceived aspect suggests a move or alteration for the next sketch. Emergence or unexpected discovery refers to the creation of unanticipated, new ideas in response to visual cues from an existing sketch.

It is necessary to highlight at this point that surprise is not the agency of the visual representation. The detection of the "new" or "unexpected" is subjective and depends on the eye of the designer who is involved in the process. When the observer ascribes the quality of surprise to a phenomenon, he can only do this in terms of his own understanding.

[c] Opening the "black box"

In this chapter, I will identify characteristics that I consider fundamental for a design machine. The objective is always to define these elements that will make it possible for the machine to be regarded as a black box from its designer's point of view. As explained so far this is the precondition for surprises to emerge in the in-between space of the designer and the machine.

I have previously referred to the design process as a fluid and spontaneous mode of production where the "new" emerges out of the dynamic interaction of the designer with its evolving artifact. To illustrate my point I will look at the artist Jackson Pollock and the mode of production he followed for his painting "Autumn Rhythm". Pollock is an indicative example of an artist who builds a dynamic relation with his artifact. As Robert Goodnough (1951) states, Pollock is not concerned with representing preconceived ideas in his paintings. He is rather concerned

> with being involved in an experience of paint and canvas, directly, without interference from the suggested forms and colors of existing objects. The nature of the experience is important. It is not something that has lost contact with reality, but might be called a synthesis of countless contacts which have become refined in the area of the emotions during the act of painting.

The actions of the painter are a black box for an outside observer and sometimes for the painter himself. As Donald Schön (1987, p.28) argues the knowing is in the action; it is tacit and spontaneously delivered without conscious deliberation. I will attempt to "open" the black box of the artist's actions by observing a series of film frames from a black and white film by Hans Namuth (Karmel, 1999) which depict the artist at work, offering a significant insight of the process Pollock is following.

We have an observer, the artist and an object of observation, his canvas. Pollock looks at the white canvas and chooses the area on which he is going to ap-

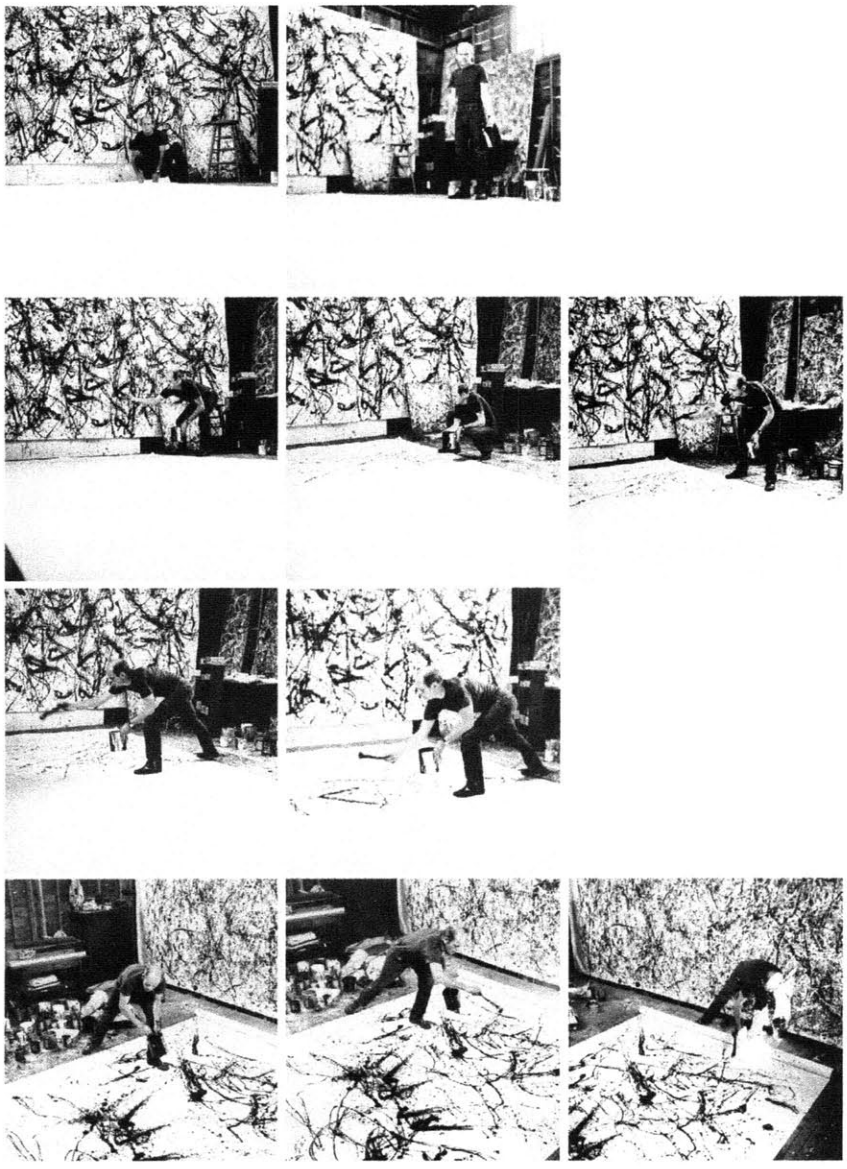


Figure 8: Jackson Pollock at work for the painting "Autumn Rythm." These frames are indicative of Pollock's dynamic relationship with his canvas.

ply a certain kind of organization. He selects his primitive units of descriptions - in this case his materials and his mode of expression, either a brush stroke or a mark or a drip etc - and he applies a transformation on the canvas (Verbeeck, 2006). Once he has applied the transformation he perceives the phenomenon again; he is left with the choice of how to continue the painting. He is free to select the same or a new area of intervention on the evolving painting and the same or new primitive units of description.

Simon (as cited in Stiny, 2006, p. 138) explains how each implementation creates a new situation. Every organization applied on the canvas provides a continuing source of new ideas to the painter. The same sequence of actions is performed over and over again in a recursive process during which the artist observes and interacts every time with the evolving artifact - phenomenon. Pollock's example substantiate Simon's argument (as cited in Stiny, 2006, p. 138) that the act of painting is "a process of cyclical interaction between painter and canvas in which current goals lead to new applications of paint, while the gradually changing pattern suggests new goals."

Feature 1: Intrinsic descriptions

> Designers need an involvement with the sensory aspects of our physical environment and is not difficult to imagine that their machine partners need a similar environment.
>
> Yona Friedman (1975, p.93)

One characteristic feature of the artistic mode of production is that during action remains undetermined what drives the choices of the artist. The artist is forming intrinsic representations. An intrinsic representation is a way of describing things that is subject to the eye of the beholder; there is no one-to-one correspondence between a phenomenon and an intuitive understanding of it. Donald A. Schön (1987, p.25) in Educating the Reflective Practitioner, defines these descriptions as constructions, attempts to put into explicit, sym-

bolic form a kind of intelligence that begins by being tacit and spontaneous. Similarly Stephen Larson (2003), in his thesis Intrinsic Representation: Bootstrapping Symbols from Experience, states that an intrinsic representation seeks to generate explicit symbols from implicit representational systems that will have an inherent meaning to that system.

The intrinsic nature of these representations relates to the fact that they carry their meaning along with them, rather than having their meaning given to them by another system. Therefore, the ability to form intrinsic descriptions presupposes interaction with the outside world, the environment where the phenomenon, the evolving artifact takes place. In the Pollock example the interaction between the artist and his canvas is evident in Pollock's words: "Having a canvas on the floor, I feel nearer, more a part of the painting. This way I can walk around it, work from all four sides and be in the painting, similar to the Indian sand papers of the West."

Designers also need the sensory involvement with their physical environment. Yona Friedman (1975, p.93) recognizes a similar need for the digital design tools. Supporting the same point, Negroponte (1970) writes in the introduction of the book The Architecture Machine:

> A design machine must have an artificial intelligence because any design procedure, set of rules, or truism is tenuous, if not subversive, when used out of context or regardless of context. It follows that a mechanism must recognize and understand the context before carrying out an operation. Therefore a machine must be able to discern changes in meaning brought about by changes in context, hence, be intelligent. (p.1)

Negroponte (1970) further supports his argument by highlighting that a machine not capable of forming its own intrinsic descriptions, but having instead the designer's personal "prejudices and distortions" and preconceived views of the world embedded in its system, would never offer a challenging environment to the machine-designer partnership.

For a design machine to be able to interact with the outside world should be equipped with sets of sensors, effectors and processors to view the real world directly and indirectly. Negroponte (1970) identifies two ways of contact of the machine with the world. The first is direct sensory information that passes into the machine through observation channels. The possibility of the machine to challenge and surprise is facilitated by the difficulty of the designer to completely control the data the machine collects. The second way of contact, that Negroponte (1970) proposes, is the information the machine could collect by eventually observing the representations of the designer.

The value of the second case does not lie on the possibility of building a machine adaptive to the methods and activities of his designer but on the possibility of the machine to contribute creatively to the conversation that takes place between itself and the designer. As mentioned before, the space where surprises can be observed is the in-between among the two members of a conversation.

Feature 2: Dynamic Vocabularies

> Architecture, unlike a game with fixed rules and a fixed number of pieces, and much like a joke, determined by context, is the croquet game in Alice in Wonderland where the Queen of Hearts keeps changing the rules.
>
> Nicholas Negroponte (1970)

The painting process of Pollock can be regarded as a computation. Pollock works his paintings by going through the same four schemas in a given sequence over and over again. In each sequence the artist chooses a frame of intervention on the canvas, applies a transformation, evaluates its outcome and removes the view frame (Verbeeck, 2006). This sequence of steps does not define how a gesture finds its form on the canvas. The artist's primitive units of expression are not defined a priori. He is free every time he applies a transformation to reflect on the entire painting and to choose anew his units of description.

The design process can be also considered as a kind of informal computation in which designs are transformed into other designs by adding, erasing or redrawing shapes (Knight, 2003, p.130). In the same way as described in the case of Pollock at every step of the process the designer "may recognize entirely new and unanticipated, emergent shapes in a design, and then use them in subsequent transformations of the design" (Knight, 2003, p.130). This freedom of the designer to change at any given moment the way of looking at things shows how the design process operates on sets of parts or primitives of the design that are dynamic, are continuously changing.

Surprise is very closely associated with emergence. Therefore a valid framework for a design machine that can generate surprises would involve the concept of emergence. Having explained the importance of changing vocabularies in the design process, the proposed framework for a design machine embraces the kind of emergence that becomes possible by computations that operate on non fixed sets of primitives.

Primitives are the elemental parts into which a system or an object is decomposed. A primitive has no internal parts of structure and cannot be constructed from combinations of the other entities from the perspective of that system. A representation in a computation has to do with the set of primitives the computation depends on and how these are combined. Knight and Stiny (2001) have classified computation in terms of the aspect of representation [1]. A computation classical in representation operates on fixed sets of primitives and explicit definitions of how they are combined, while a computation non-classical in representation operates without fixed units and primitives (Knight & Stiny, 2001).

---

1. In their paper *Classical and Non-Classical Computation*, Knight and Stiny (2003) classify four types of computation applying the classical/non classical distinction on two aspects of computation, representation and process. For the scope of the current thesis I am interested in the distinction provided when taking into account only the aspect of representation.

In a computation that is "classical in representation", emergence may arise from combinations of pre-existing fixed sets of primitives. On the other hand, in a computation "non-classical in representation" , emergence may arise by the creation of new kinds of primitives. These two modes of production

> provide two ways for describing and understanding change and creativity: as the unfolding consequences of fixed combinatorial rules on bounded sets of pre-defined primitives or as new processes and interactions that come into play over time to define new primitives (Cariani, 2006).

Stiny (2006, 1998) with his work on shape grammars offers a unique insight in the way novelty can occur when a computation operates on non fixed set of primitives. Shape Grammars is a visual computational system that does design directly through computation on shapes. At every step, the rules are applied on shapes not explicitly defined by either the author or the user of the grammar. A rule can be used to compute many different designs by applying or not applying it to different shapes in a computation. As Stiny (2006, p.53) writes in his book Shape: Talking about Seeing and Doing calculating with shapes requires no parsing of shapes into fixed parts. Ambiguity is always present in such a visual calculation as the compositional units can be recombined and decomposed in different ways. The novelty ambiguity brings makes creative design possible.

> Everything fuses and divides in between. The real secret to calculating with shapes is to see that there's always something new. No matter what I do, it's a surprise. What you see is what you get (Stiny, 2006, p.59).

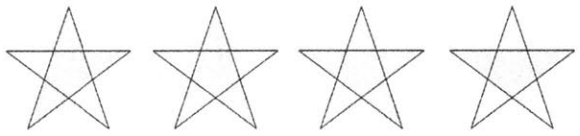


Figure 9:
Operating on non-fixed sets of primitives.

[d] A framework for a design machine

The cyclical process described in the example of Pollock and later extended to the design process can be depicted by the diagram of figure 9. The observer - designer looks at the phenomenon, the evolving design and decides on how he wants to decompose it into parts. He forms intrinsic descriptions meaningful to him.

The produced vocabulary is further manipulated, transforming the evolving artifact. After a step of transformations the parts of the artifact fuse back to form the new state of the phenomenon. The designer observes the current state of the phenomenon and he decomposes it into the same or new descriptions. This recursive process continues until the designer decides that the state of the artifact is satisfactory.

The two singular features that were pointed out as essential to be introduced to the framework for the design machine is the intrinsic nature of the descriptions formed by the designer and the ability of this cyclical process to operate on non fixed units of description.

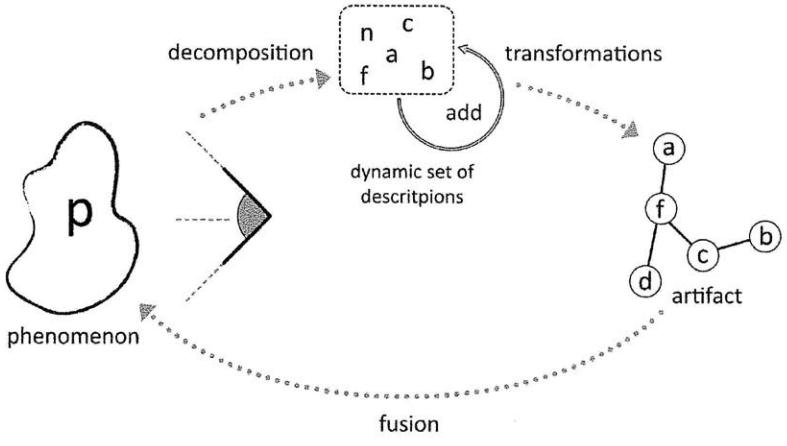


Figure 10: Cyclical process of interaction among designer and evolving design

If the human observer/designer is replaced by a machine, then we have to input to the machine hand-crafted knowledge about the world expressed in some knowledge representation language. From that point, on reasoning is separated from perception and action. The languages of descriptions are fixed and the artifact can only be a product of combinations and transformations of the primitive elements of the language. This framework describes most of the current computational design tools that designers use.

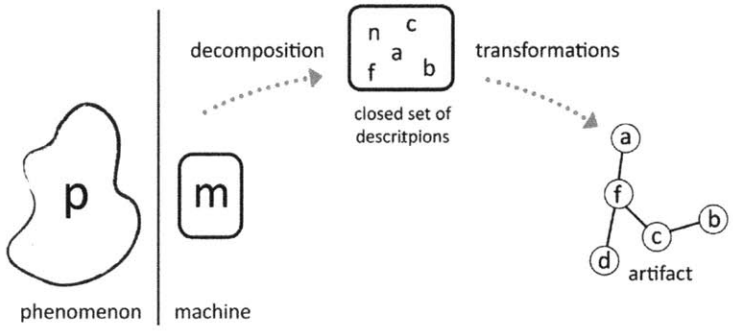


Figure 11: When the observer is replaced by a computer the open and dynamic set of vocabularies depicted in diagram of figure 10 becomes closed and static. The system comes to be completely disembodied from the outside world.

The proposed computational framework wants the design machine equipped with receptors, sensory mechanisms that enable the system to sense its environment continuously. The machine can decompose the observed phenomenon to discrete parts and generate its own descriptions. Transformations are applied on the generated symbols and an artifact is produced. At next step the machine can look back at the artifact and generate the same or new sets of descriptions. New transformations are applied on the evolving artifact and the process happens recursively. To summarize, the suggested computational framework operates on dynamic descriptions.

The word dynamic has to do:

> **with the ability of the machine to interact with the outside world through a cyclical process, forming descriptions that bear content independent of the interpretations of their designers.**

> **with the ability of the machine to operate on vocabularies that do not belong to a closed, static set but they could be constantly changing.**

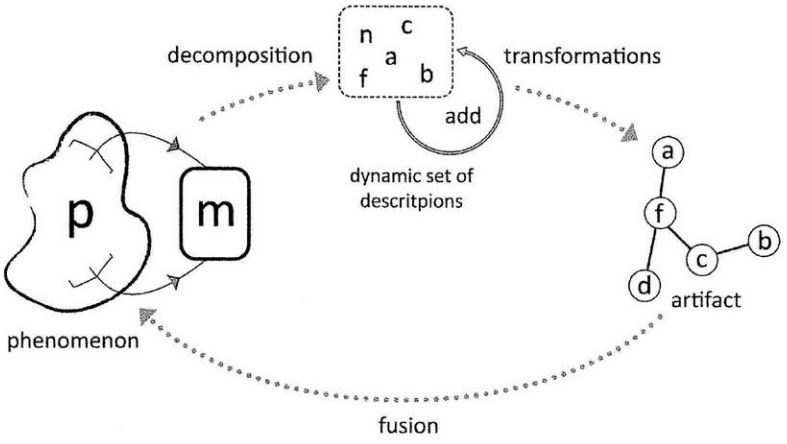


Figure 12: Proposed framework for a design machine operating on dynamic descriptions.

PART 03 On machines

[a] About representation

We are surrounded by a wide range of machines that vary from being physical and mechanical, for example bulldozers that restructure our environment, to being completely symbolic and disembodied from the outside world, like chess-playing computers and expert systems to name a few. The question is where in this range, a valid approach for a design machine could be positioned. The exploration of the characteristic features of design, deployed in the previous chapter, demonstrated that the issue of representation is a fundamental one in our inquiry on design machines. The research field of Artificial Intelligence has been founded on the issue of representation and has to demonstrate an important and relevant debate that has been going on over the last 50 years. During this period, the two main approaches that have dominated the efforts on building intelligent systems are the "symbol systems" and "behavior based" approach (Rao 2002).

The symbol systems approach is based on the ability to store information about the world in computer memory. The system usually contains hand-crafted knowledge about the world expressed in some knowledge representation language, for example if-then rules, frames, declarative statements etc. This knowledge and the world description are used by a problem solving mechanism to achieve some goal. The solution is passed to an execution module that performs actual actions in the world to achieve the goal. The great successes of this approach relied on the fact that digital computers are ideally suited to manipulate symbols, i.e. to generate combinations of symbol-primitives and logical operations.

There are several reasons why this approach is considered a valid one as far as design machines are concerned. First it is based on the separation of reasoning from perception and action and on the assumption that knowledge and the manipulation of knowledge can and should be separated from the physical body of the system (Rao 2002). Second, it requires representations, descriptions of the world, which are surrogates, substitutes for the actual phenom-

ena. Representations require determining consequences by thinking rather than acting, that is, by reasoning about the world rather than taking action in it.

For the needs of the current thesis, I will look at the behavior based approach which argues in favor of intelligences that are being built through the interaction of the system with the world.

[b] classification of machines

Moving towards computational systems that try alternative standpoints to representation, I will set the background for my further exploration by looking into a series of computational systems. I will question the "eligibility" of each of them as a possible design machine, in terms of the design criteria previously stated. At the end of the chapter I will have outlined a classification of machines through an additive bottom-up process from the most elemental version of a machine to more complex ones.

Let's define a machine as an entity with an internal configuration. Such a definition presupposes the existence of an external environment in which the machine is situated. If the machine is not equipped either with an input channel to receive a stimulus or with an output channel to give a response then there is no interaction between the internal configuration of the machine and the external environment. An example of such a machine that does not receive any input, nor has an output to cause a change to the external environment is the atomic clock, a device that uses atomic vibration to measure time.

Figure 13: The first atomic clock, constructed in 1949 by the US National Bureau of Standards

The internal configuration of the atomic clock is comprised of three parts: a crystal oscillator, a container of cesium-133 atoms and a detector. In an isolated, closed environment, by means of the crystal oscillator a microwave pulse of a certain frequency is created which excites the cesium atoms; when the atoms are excited with the exactly desired frequency they change energy state. The detector detects the change and locks the frequency of the oscillator to the frequency that caused the change of the energy state. This sequence of steps occurs in a continuous loop. This is an example of a system for which every change of its internal state depends on its configuration and its isolated internal environment; no change of its internal state is affected by external conditions or acts in some way on the environment.

01 receptor machine

In the previous chapter, I highlighted the necessity for a design machine to be equipped with sensory mechanisms which provide information about the world. Therefore a system with an internal configuration that does not allow any input of information and stimulus from the outside environment would never constitute a valid approach for a design machine.

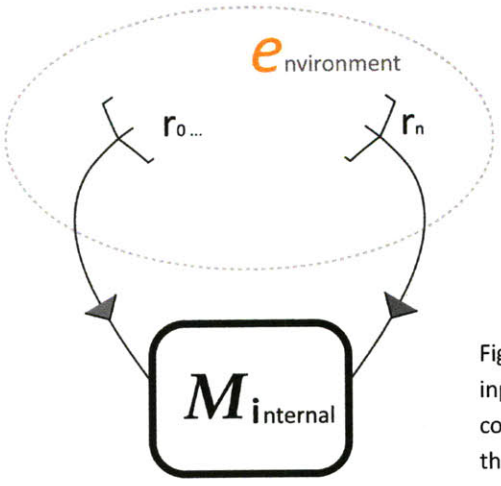


Figure 14: The receptor machine, input channels connect the internal configuration of the machine with the external environment.

Let's equip a machine with input channels, receptors that enable its interaction with the external environment (Figure 14). Receptors determine how events in the external environment are related to the internal informational states of the machine with the causal flow from the environment to the machine and what kinds of organizations (perceptual categories, features, and primitives) can be made on the environment (Cariani, 1998).

Stiny and Gips (1978, p. 19-31) in their book Algorithmic Aesthetics introduce receptors as a basic part in their proposed framework for a design algorithm. A receptor is what provides the sensory connection between the algorithm and the outside world. It consists of two parts: a transducer(s) and a linked algorithm. The transducer can be a sensor, camera or a traditional input device like a keyboard or a mouse. The algorithm linked to the transducer produces a finite sequence of symbols as output based on what the transducer has sensed.

Since the system is sensing natural phenomena, the signal from being continuous gets discretized. No matter what the nature of the signal that passes through the input channel (transducer) is - electric, electromagnetic, acoustic etc - at a given moment the channel is characterized by a set of distinguishable states. At each of the discrete moments in time, each channel will be found in one or another of a finite number of possible states or conditions. These states might be given any variety of symbolic names. This sequence of symbols is the descriptions and the interpretations of initial conditions that are to be used to make an object. To link this to our previous discussion these descriptions are the set of vocabularies on which the design algorithm operates.

Stiny and Gips (1978, p. 14) make it clear that in their proposed framework the conventions and criteria encoded in the design algorithms, which also include the algorithms that convert the input signal to a finite sequence of symbols, are static. They do not change over time; they correspond to some predefined and fixed approach.

02 effector machine

Let's return to the initial assumption of a machine with an internal configuration but with no interaction with the outside environment.

Design involves reorganization and physical manifestation of a changed order. Therefore, a design machine should be able to act upon the environment and apply a specific reorganization. To do so, it has to be equipped with output channels that can influence the outside world, the effectors. An effector determines how the internal informational states of the machine are related to the external environment with the causal flow going from the machine to the environment. Given some sequence of symbols as input, the effector performs some action or produces some object in the external world.

Effectors are the other of the three main parts of the framework for a design algorithm proposed by Stiny and Gips. They define an effector as the output device of the system comprising an algorithm and a linked transducer. In a design algorithm, the effector has as input the description of the object that is to be produced. The nature of the transducer in the effector would depend on the art form for which the design algorithm is constructed. Effectors might be printers, CNC machines, and even robots (Stiny & Gips, 1978, p.201).

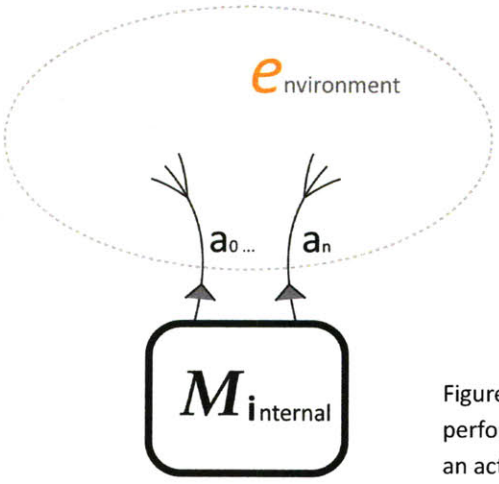
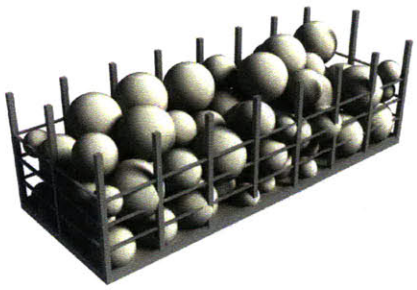


Figure 15: The effector machine, performs through output channels an action upon the environment

An example of an effector machine is the digital fabrication system (Figure 16,17) used by Fabio Gramazio and Matthias Kohler to construct brick walls with precisely controlled layouts. The system was composed of a design algorithm that was acquiring the data for generating brick wall patterns from a virtual simulation of falling spheres. The output of the algorithm was linked to an 8-axis robotic arm which was precisely placing the bricks according to their position and rotation.



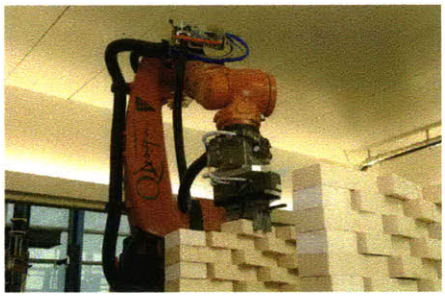


Figure 16, 17: From the project "Non-Standardised Brick Façade" by Gramazio and Kohler, Gantenbein Vineyard Facade, Fläsch (Switzerland), 2006

A reasonable criticism of this system, in terms of its validity as a design machine generator of surprises, would be that once the certain specifications and the vocabularies on which the system operates are decided, then it becomes closed, therefore no surprises are anticipated.

03 reaction machine

The exploration of the two previous two categories, the receptor and the effector machine, demonstrated that a design machine whose criterion is to generate surprise should include a combination of both receptors and effectors.

Let's now assume we have a machine equipped with both receptors and effectors. However, it has no storage ability, no internal memory. It can only receive a stimulus through its receptor and respond to the stimulus through its effector.

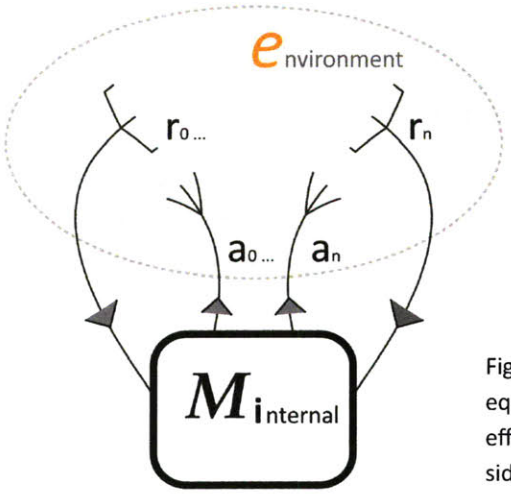


Figure 18: The reaction machine equipped with both receptors and effectors to interact with the outside environment

Let's consider as a typical example of such a reaction machine, the first types of the series of the Braitenberg Vehicles. In his book Vehicles: Experiments in Synthetic Psychology, the neuroanatomist Valentino Braitenberg (1984) describes 13 conceptual constructions, which he calls "vehicles". Vehicles are machines with a simple internal structure and simple control mechanisms.

The first and simplest vehicle has one sensor and one motor. The vehicle's sensor is the receptor of the system that senses a specific quality and its motor is its effector which reacts to the input of the sensor. The speed of the motor is controlled by the sensor. The vehicle can move only forward; the more there is of the quality the sensor is tuned the faster the motor goes.

By gradually adding more machinery to the vehicles he is building more complex behavior. Vehicles 2, 3 and 4 are equipped with two sensors to detect the environment and two motors to propel themselves. The relationships between the sensors and actuators determine the specific behavior for each machine. By connecting simple motors to sensors, crossing wires and making some of them inhibitory, simple machines can be constructed that show fear, aggression, love, affection etc.

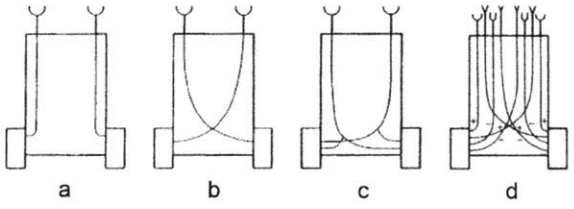


Figure 19: Vehicles of the types 2 and 3. The relationships between motors and sensors determine the behavior of the vehicle.

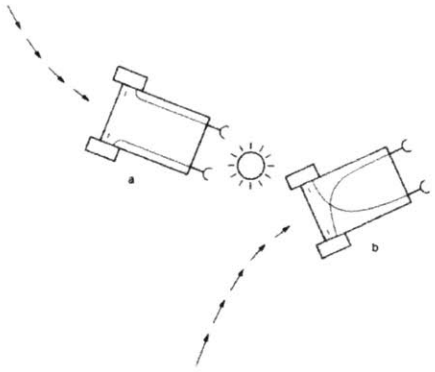


Figure 20: Vehicle of type 3 with inhibitory influence of the sensors on the motors.

It is important to mention that at the local level the internal configuration of each simple reaction machine is what determines its specific behavior. The structural connections of the sensors to the motors are predefined and designate how the vehicle will move when interacting with the source that "excites" its sensors. Up to this level the behavior of the vehicle is predictable, maybe even uninteresting; the vehicle will always be prisoner of the fixed topology connections of its sensor to its motors. Braitenberg is incrementally building more complex machines (from vehicle n.1 to vehicle n.4), with new behaviors

emerging each time by applying various combinations of the elemental topological relation of vehicle 1. However once the topological connections for every machine are decided and set then it becomes a closed system.

What makes this computational system relevant to the design machine exploration is that the interesting behavior (and this, as already mentioned, depends only in the eye of an outside observer) emerges through the interaction of the machine with its external environment. The external environment might comprise the sources that affect the vehicle's sensors as well as the other vehicles that coexist. When moving to the global scale of the observed behavior then there is no inherent central representation, no predefined vocabulary and structure in the way the vehicles are going to behave which is imposed or predefined, but through the continual exchange of information, unexpected behavior emerges. The vehicles demonstrate how interesting behavior can emerge out of the interaction of a set of simple machines operating without

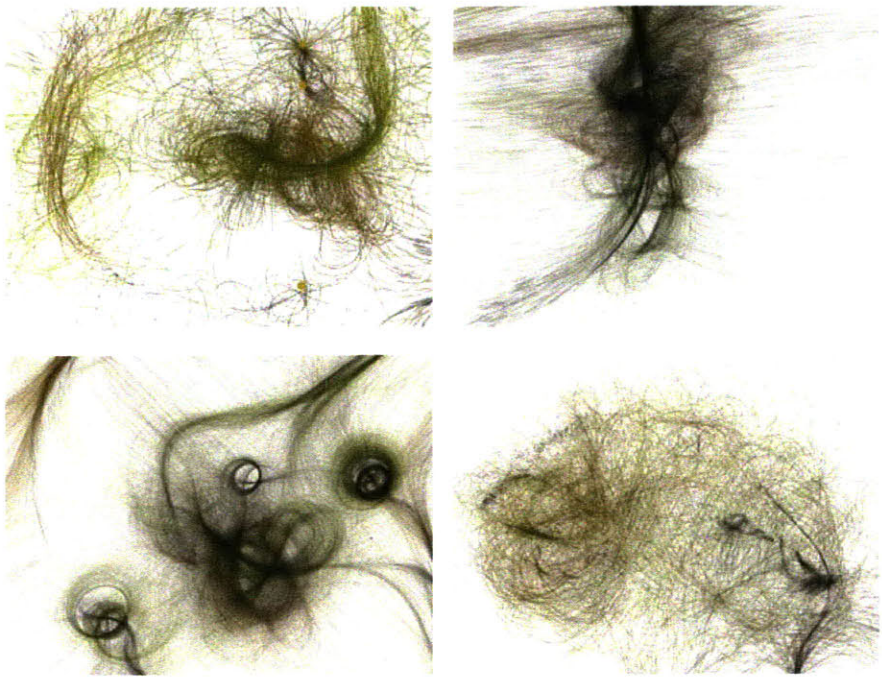


Figure 21 - 24: Tissue Software (2002) by C.E.B. Reas, the project explores the movements of synthetic neural systems. People affect the software by positioning a group of points on the screen.

centralized control. The global system remains open by setting the machines operate in a dynamic environment where new behavior can be determined and additional channels of interpretation for the observer can be opened.

An implemented example based on the Braitenberg conceptions is the Tissue Software built by the artist C.E.B. Reas. Reas (2007) developed his software with the intention to create an open field where "the material form as well as the semantic content is open". His work demonstrates how out of simple reaction machines in a dynamic changing environment unexpected patterns of visual form can emerge (Figure 20-24). He writes "Simple layers of code combine to create the deceptively complicated behavior of these software machines".

The word "deceptively" brings into the discussion the notion of the black box machine. The observer at the beginning cannot predict the global behavior of the system. However after a while, by interacting with the system gain a certain degree of understanding of how the system works.

Since the system operates via simple reflexive input and output signals, its behavior is a prisoner of its immediate external inputs. In order to move a reflexive system at a higher level of complexity and introduce possibilities of unexpected behavior, an elementary form of memory should be added.

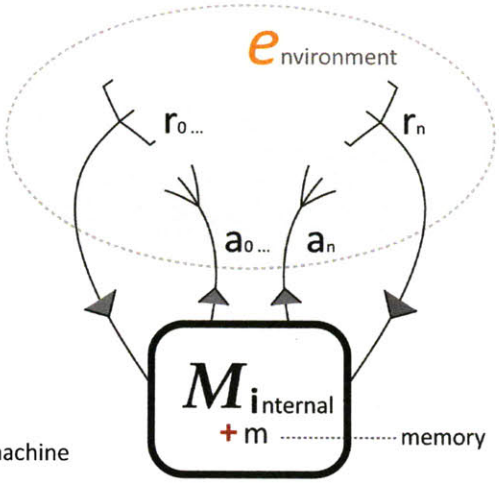


Figure 25: The memory machine

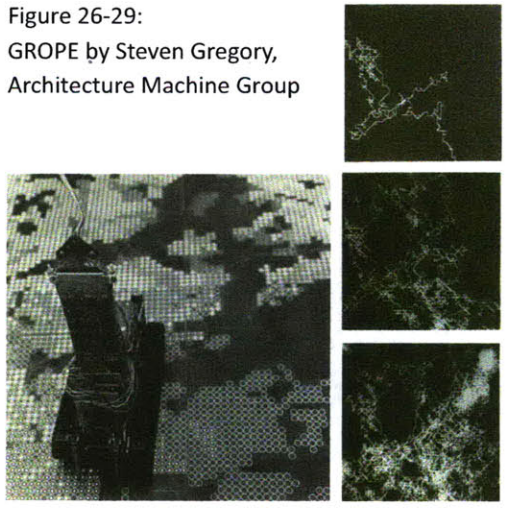
04 memory machine

The memory machine (Figure 25) is a system capable of interacting with the environment through receptors and effectors. It has an elementary ability of storage so that it can operate based on memory dependent mappings. Since memory dependent mappings are present, the system becomes less dependent on its immediate past input (case of the reaction) and more dependent on its recent history (Cariani, 1998).

A typical example of this category is GROPE, an Architecture Machine Group project developed by Steven Gregory (Figure 26). Grope was a machine able of interfacing with the real world. It was a mobile unit that could crawl over maps. It was equipped with a "seeing" mechanism constructed by simple photocells that could register on or off states according to whether they could identify light or not. Grope's role was to look for "interesting things". It compared its past to its present location and that determined its future move. It occasionally employed random numbers to avoid rats (Negroponte, 1970, p. 109-110).

Once more "unexpected" behavior depends on the eye of the observer. The observer looks at the behavior of GROPE to identify "interesting" things rather that receiving a testimony on what is interesting or not. GROPE was developed as one of the first appendages to an architecture machine because it is an interface that explores the real world (Negroponte, 1970, p. 109).

Figure 26-29:
GROPE by Steven Gregory,
Architecture Machine Group



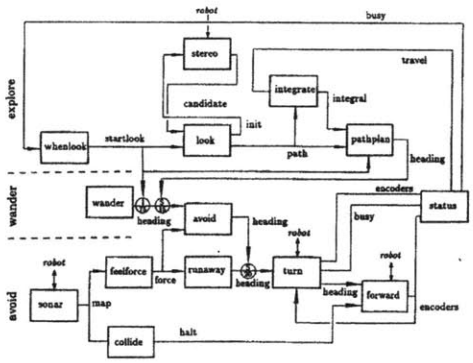


Figure 30:
Rodney Brooks subsumption architecture

Another example of the action-reaction relationship but with elementary storage ability comes from Rodney Brook's Subsumption architecture. Brooks builds "Creatures", autonomous mobile agents who can demonstrate low-level intelligence. Adopting an alternative view to representation from the traditional AI symbol systems approach, Brooks (1991) proposes that intelligence should be built incrementally through the interaction of the system with the world through perception and action.

A characteristic feature of the Subsumption architecture is it being composed of layers, each layer built on top of the others. The layers are combined through mechanisms of suppression and inhibition. For example in the system of Figure 30 there are three layers of control. The first and lowest layer of control makes sure that the robot avoids static and dynamic obstacles; the second layer imbues the robot with the ability to wander around; the third layer makes the robot try to explore. Each layer is composed of a fixed-topology network of simple finite state machines. The finite state machines run asynchronously, sending and receiving fixed length messages over wires.

The model of the subsumption architecture has some elements that seem relevant to a design machine. The system operates under no central representation interfacing "directly to the world through perception and action" (Brooks, 1991). According to Brooks by exploiting the complexity of the environment is possible to build systems that can "lead to complex behavior with non-centralized representations of the world".

For the scope of this thesis a design machine has been defined in terms of its ability to generate surprises, thus being a black box to its observer. The proposed framework wants the machine to generate its own dynamic descriptions. A design machine built out of multiple layers of finite state machines, as in the subsumption architecture, might be able to generate the conditions for a complex behavior to emerge through the interaction with the environment.

However it will not be possible for the system to form descriptions on which, it could apply meaning and eventually manage to learn, out of interfacing with the real world. It will manage at the beginning to surprise its observer but soon its behavior will become anticipated given that it will be always dependent on the topological connections of the different levels.

05 intrinsic representation machine

We reached a point where we need a system that has enough memory to store information in addition to the receptors and the effectors by which it interfaces with the real world. Its distinctive characteristic is its ability to form descriptions intrinsically about the external environment, "entities" that bear content independent of our interpretations.

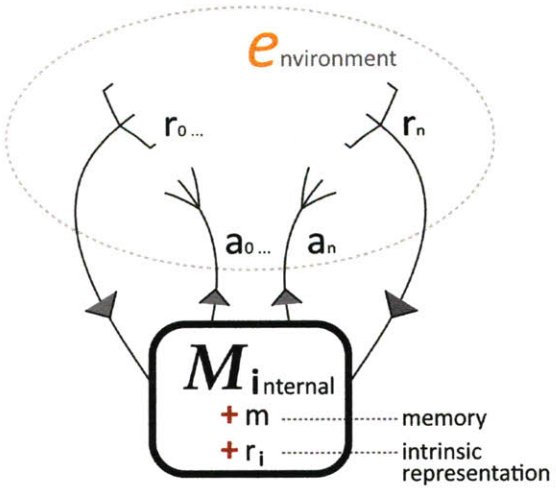


Figure 31: the intrinsic representation machine

This category was named after the intrinsic representation model built by Stephen Larson (2003). Larson proposes with his work that meanings can be learned by computational systems without explicit instruction. He believes that computational systems should be able to form their own representations of the world and he takes a step towards that direction by showing how descriptions can be build up from low level sensory information by making statistical distinctions between similar and dissimilar input.
He actually built a working system which successfully learns to associate symbols to blocks in a simple 2d blocks world and to associate the position of the eye with the position of its arm.

The intrinsic representation machine seems a valid approach to start as far as design machines are concerned. It satisfies both the conditions set in chapter 2. First, it is a system that is equipped with receptors and effectors therefore it can observe its phenomenon and eventually act on it. Second it suggests a possible direction on how a system could form dynamic descriptions, sets of vocabularies that could be continuously changing.

PART 04 A step towards intrinsic descriptions

[a] zooming into the machine-environment interaction

The first three parts of the current thesis were an inquiry meant to detect a valid approach for a design machine in order for it to constitute a creative partner in the designer-machine relationship. The criterion introduced as a precondition, taking into account the nature of the design process, was the element of surprise. With this criterion in mind, I demonstrated (chapter 3) why it is necessary for a design machine to be capable of sensing its external environment and forming intrinsic descriptions about the world, "entities" that bear content independent of our interpretations. At the end of chapter 3, I presented a framework for a design machine that includes the concept of dynamic descriptions.

To briefly summarize, the proposed framework is about a computational system equipped with receptors, sensory mechanisms that enable the system to sense its environment continuously. The machine decomposes the observed phenomenon to discrete parts and generates its own descriptions. Then, the system applies one or a series of transformations operating on the generated vocabulary and an artifact is produced. This sequence of steps occurs over and over again; the system looks back at the artifact and generates the same or new sets of vocabularies. New transformations are applied on the evolving artifact. This recursive process goes on until the evolving artifact reaches a satisfactory state.

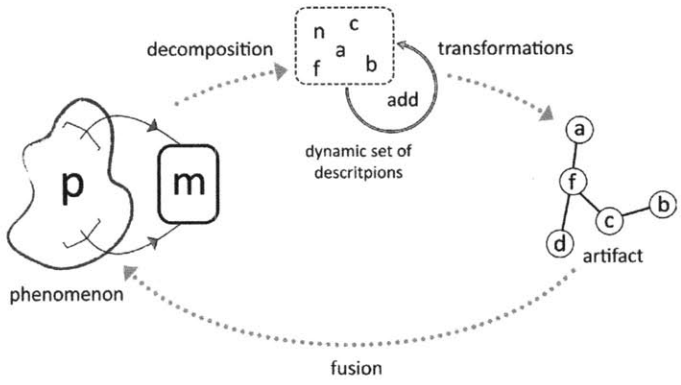


Figure 32: The proposed framework for a design machine

This chapter focuses on a specific part of this loop. It experiments on a possible way by which the design machine can algorithmically decompose what it senses from the outside world. The sensory mechanism under investigation is the machine's vision. Negroponte (1970) believed that the first sensory mechanism to be incorporated in an architecture machine would be a seeing mechanism. He writes: "At first, machines with eyes will observe simple physical models; eventually they will observe real environments" (Negroponte, 1970, p.29).

For the need of the current experiment the machine's "eye" will observe still images with an aspiration similar to Negroponte's vision that at a later stage the machine will observe real environments. The interest for the machine being able to sense the world lies on the desire to build a system that will generate its own descriptions about its environment. As a possible computational model to use as the infrastructure for my development, I will selected the research work of Stephen Larson (2003) who was the first to have a system intrinsically grounding symbols in perception.

[b] Larson's model of Intrinsic Representation

Stephen Larson (2003) with his model of intrinsic representation takes a step towards understanding how meanings can be learned by a computational system without explicit instruction. With his implementation he demonstrates how symbols can emerge from a system that receives low-level perceptual information through its sensors and without supervision discovers regularities in that information.

I will briefly explain how his implemented system works. Two sensory systems receive streams of data from the outside world. The data is organized by a sub-system of two self-organizing maps, which organizes them with respect to their similarity, placing similar regularities in proximity, and dissimilar regularities farther apart (Larson, 2003, p. 37). When the map reaches a specific criterion of self-organization the system uses a clustering algorithm to separate the major clusters in space (Larson, 2003, p. 48). Once grouped, a cluster gains the

ability to act as a unit that can be activated and deactivated.

Each of the two sets of information, the visual input from the system's "eye" and the input of the system's arm, is organized by a separate self-organizing map. As clusters are activated by the incoming data in the two parallel maps, they are associated together by their frequency of coincidence. The more often two clusters are active simultaneously, the more associated they are (Larson, 2003, p. 37). Through this model Larson built a successfully working implementation of a system capable of learning symbols for blocks in a blocks world and capable of associating the movement of the system's eye with the movement of the system's arm.

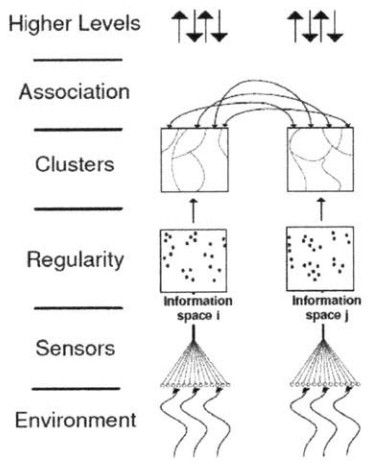


Figure 33: Larson's model of Intrinsic Representation

[c] On classification

The theory on which Larson (2003) builds his computational model is that the information the brain collects about the environment in order to form symbols contains statistically detectable patterns. Brains must reduce the informational entropy coming from its sensory mechanisms. In other words, in order for the brain to make sense of the world out of the significant amounts of information it collects, it manages to find relationships among the information it receives and filter out the irrelevant information (Larson, 2003, p.40). Negroponte (1966, p. 4) in his Master thesis in 1966, also argued in favor of the approach that the eye and mind work like a computer proceeding through a

complicated process of input, working on the information and finally organizing it to be intelligible input, at a much faster rate than the most sophisticated data process equipment.

I will use an example to illustrate Larson's and Negroponte's point of view. Let's assume that we are looking at a line drawing of a square on a white sheet of paper. Even though what we identify is the figure of a square, there is in fact an infinity of other configurations that could have produced the same set of lines. An explanation of why we do not notice all these alternatives could be the restricted storage capacity of our visual memory; to compensate for that our visual system chooses to identify a specific organization on the bare perceptual input. The inability of our perceptual system to store and process large amounts of information is evident also when we first try to identify the organization of a very complex pattern. We can probably get a vague idea of its organization but we cannot "photograph" its total configuration.

Design is a process very closely associated to perception. In order for the designer to be able to organize matter, is necessary to generate a description of what he sees, in other words to apply some kind of initial organization. The act of creating descriptions is a way of gaining control over the large amount of input that enters the designer's perceptual system; this organization further enables him to "handle" the information and reorganize it. Taking these points under consideration self-organization algorithms seems a promising computational model for experimenting with how a design machine could intrinsically make sense of the outside world.

[b] experimenting with self-organization

In this part, I will describe an experiment on implementing a self-organizing map algorithm intended to classify the visual data from a two dimensional image. I will outline the algorithm's pseudo code so that the reader can follow and understand the process. Then I will demonstrate the an example run of the algorithm on a black and white image.

Instead of working on the pixel level of the image, limiting the system to "read" only r-g-b color values, I chose to experiment classifying fragmented parts of the image. Working with fragments offers the flexibility to define various properties through which the system is able to "read" the fragment. These operations provide to the system with the data to be classified by the self-organizing algorithm.

Figure 15 is the key diagram for understanding the process of the implemented algorithm. The image is divided in a number of fragments, which is determined by the user of the algorithm. The algorithm runs a property operation on a selected fragment and an n-dimensional vector, a set of numbers that describe the fragment in terms of the property under investigation, is generated. This n-dimensional vector, is the sample vector that is being classified

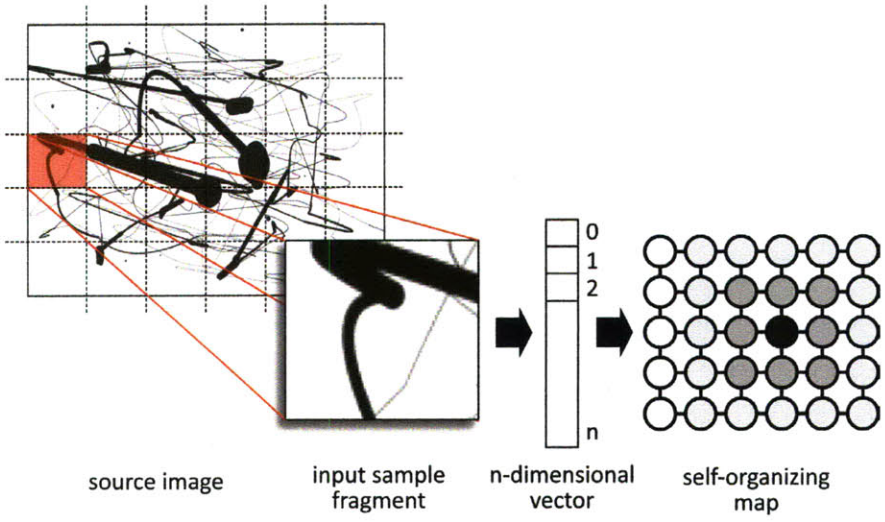


Figure 34: From an image to a self-organizing map

by the self-organizing algorithm. This sequence occurs for the total number of the fragments.

I will first define what I mean by property operation and then I will briefly explain what a self-organizing map is and how it works. Having clarified the various parts of the system I will describe the pseudo code of the algorithm and demonstrate the example run.

01 Property operations

The first natural question that arises is how the system interprets its visual input, which for the requirements of this experiment is a two-dimensional image. As already mentioned I chose to examine processes that interpret an image in terms of its pixels. However instead of limiting the system to "read" only r-g-b color values, I decided to look at operations that could provide a description about parts of the image.

Shimon Ullman (1996, p. 278) proposes that the visual system in order to perceive shape properties and spatial relationships assembles "visual routines" using sequences of basic elementary operations. By further combining these basic operations using different sequences it composes new visual routines mechanisms for different properties and relations. Motivated by Ullman's research work I tried to identify basic operations by which the system could interpret the fragments from the image. I would like to clarify that my intention is not to identify operations that will mimic the human visual system; it is rather an effort to explore how basic operations that are being used for years in the computer vision research field could be imported in a self-organization system. Some possible basic operations could be:

> Mapping of the red-green-blue color values of the fragment.
> Calculating the ratio of the black and white pixels of a fragment characterizes the contrast of an image.
> Calculating the "uniformity" of an image; every pixel is allocated a score according to whether its nearest neighbors are of the same or different color.
> Identifying connectivity; the operation starts from a single pixel and by check-

ing its nearest neighbors and the neighbors' neighbors can track boundaries of the object. The same operation could be used to identify spatial relations like e.g. enclosure.

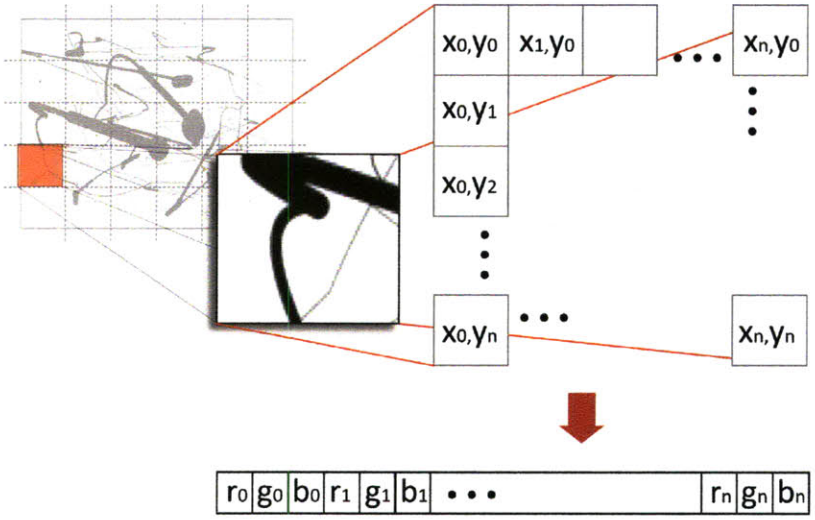


Figure 35: the r, g, b values of every pixel of the fragment are mapped in an array that represent the property vector of the fragment. The number of values to be stored to the list is equal to the number of dimensions inserted to the system.

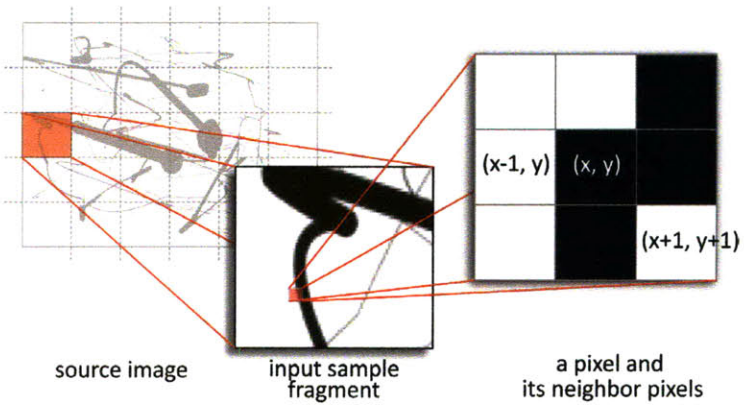


Figure 36: operation that scores every pixel of the fragment according to the number of neighbors that have the same color. For example, the pixel (x,y) of the diagram would be scored with 3.

02 Self-organizing map

Self-organizing maps (SOMs) are a computational model that creates a visual representation of a large body of data. The produced representation allows relationships in that data to become observable to a human viewer (Larson, 2003). An SOM arranges data by performing an incremental statistical analysis on the information it receives; it plots the similarities of the data by grouping similar data items together. It compresses information while preserving the most important topological and metric relationships of the primary data items.

In my implementation I used the basic form of an SOM, consisting of a two-dimensional regular grid of nodes. The map is populated with a number of cells which in our experiment is equal to the number of fragments in which the input image has been divided in. Each cell of the map is associated to a weight vector of equal dimensions to that of the sample vector of a fragment. Initially, the weights of the vector are randomly generated. At every step, the sample vector goes through all the weight vectors and calculates the metric distance between itself and each weight vector. The one with the shortest distance will be the winner cell.

The weights of the winner cell and of its neighbor cells will be updated according to a learning function to better match the sample vector. This process repeats for all the sample vectors available. The learning rate is reduced as the learning process evolves. The learning function is:

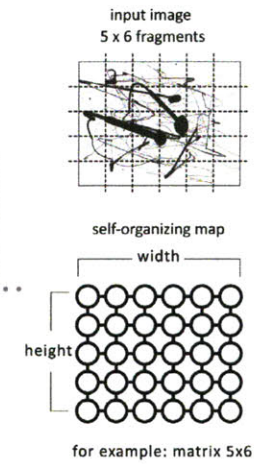
$$m_i(t+1) = m_i(t)+a(t)[x(t)-m_i(t)]$$

where m_i is the target vector for node i, x is the sample vector, a is the learning rate and t is a discrete time coordinate.

Next, I demonstrate the pseudocode of the SOM algorithm. The code was developed in JAVA using the Processing library for the visualization of the process.

to initialize and set up a self-organizing map:

- initialize a matrix of height and width equal to the number of fragments of the image in the y and x dimension
- populate the matrix with nodes
- allocated each node with a vector with random weights - the number of weights of every vector should be equal with the number of dimensions the property vector of the fragment has



For every fragment of the image:

- find the winner node whose weight vector best matches the property vector of the fragment comparing the euclidean distance
- attach the fragment to the winner node to enable the visualization of the process
- discover which are the nodes surrounding the winner node
- update the weights of the winner node and of its neighbor nodes according to the learning function

This sequence of steps repeats for all the fragments available for a defined number of iterations until the training criterion is reached

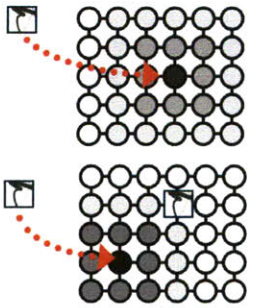
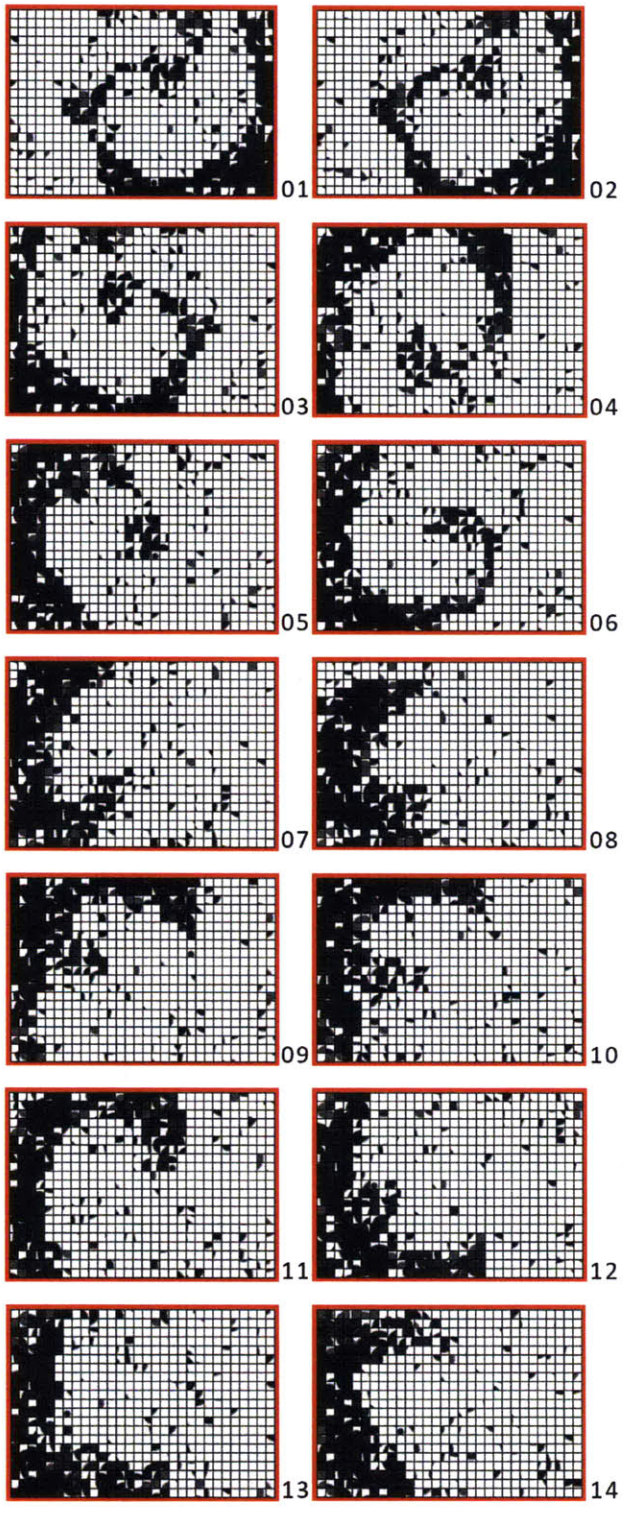


Figure 37

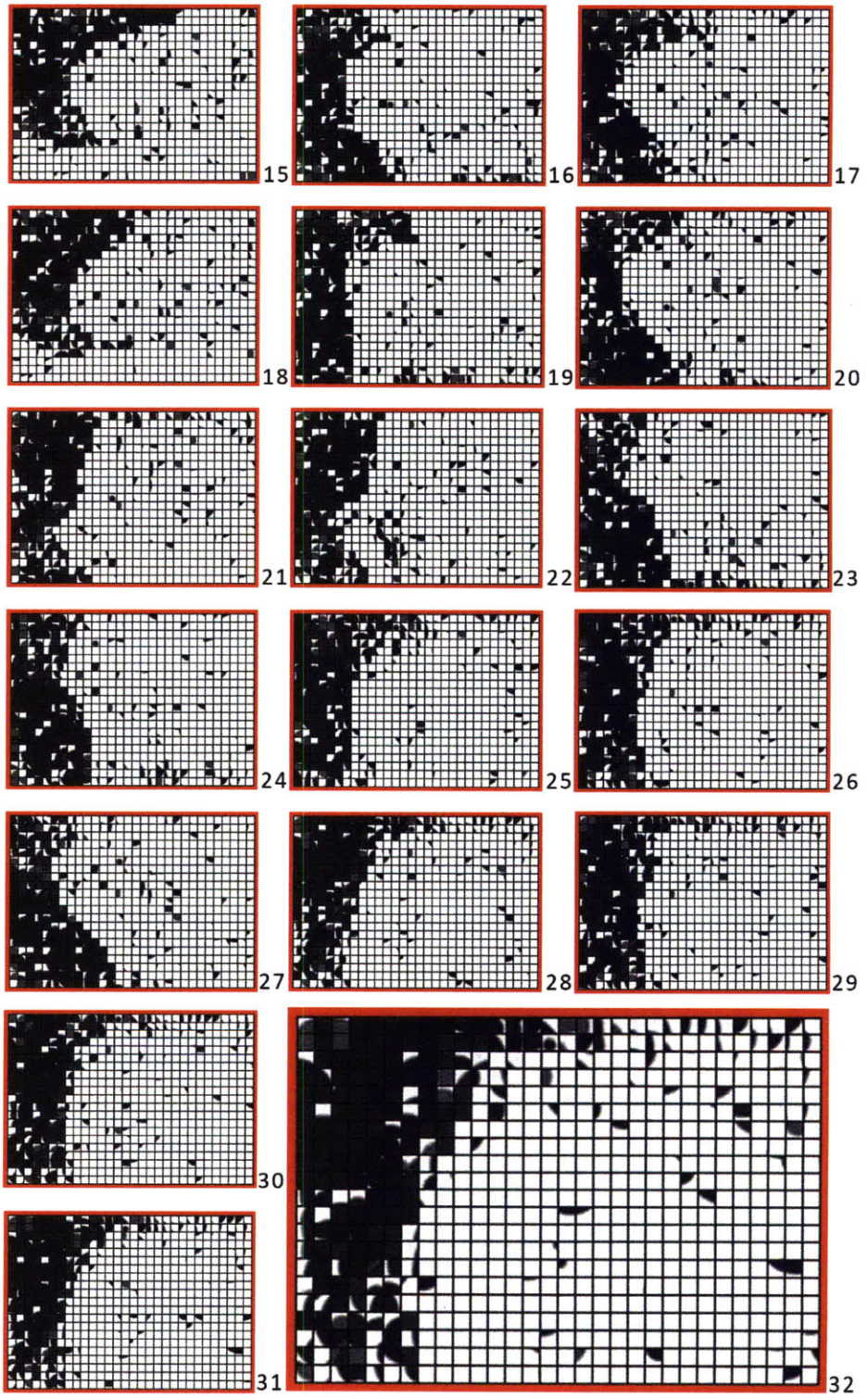
Figure 18 demonstrates an example run of the SOM algorithm on the fragments of an input black and white image. These frames are selected from a run series of one-thousand (1000) iterations. From the very first iterations one can see that the fragments start to organize into separate regions of similar brightness. As iterations progress the regions with dark and the ones with light fragments become more distinctly separate. One can see that the last frames are very similar to each other since, as the discrete time coordinate increases the learning rate decreases and the self-organizing map algorithm converges to on specific organization, as intended.

Figure 38:
Self-organization of the input image we see below

image resolution:
300x210
fragment resolution:
10 x 10
number of iterations:
1000
n-dimensions of each property vector: 100



The next step after the map has reached a level of organization is to run a clustering algorithm which would recognize the groupings in the organized data.


15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32

Contributions

In this thesis I have:

• introduced surprise as a design criterion for a design machine

• proposed a computational framework for a design machine able of perceiving its environment and of forming intrinsic descriptions. A key characteristic of the proposed framework is that through the constant interaction of the design machine with the environment the descriptions generated can be constantly changing. My suggestion is that such a feature is a step closer to building design machines capable of creative activity.

• I outlined a classification of machines with an alternative approach to representation from simpler to more complex. I examined every case in terms of its eligibility as a design machine.

• suggested a use of self-organizing maps implementation in the domain of design

Conclusive remarks

The "classical" - "non-classical" distinction, some years ago, motivated my interest in computation. That distinction was meant to distinguish the paradigm of modern thought in architecture, the classical thought, as associated with closed strategies for arriving at predetermined goals (Eisenman, 1984). At that point it seemed to me that computation could provide the means to release architectural form and the design process from the remains of the classical paradigm, any kind of origins, types, ideals that were associated to a priori sets of values.

Some years later, another distinction between the "classical" and the "non-classical", in terms of computation this time (Knight, Stiny, 1991), provided me with another important insight. The danger of closed strategies that impede novelty can as well exist in the realm of computation when using digital tools that operate on fixed units of description and strong hierarchies. This realization was the reason for this thesis; it formed my interest in exploring possible

ways by which computational tools can be creative partners in design. It led me to envision design machines as black boxes, generators of surprises for their designers-observers. It was an effort to move again to the non-classical: to the dynamic instead of the static, the pluralistic instead of the monolithic, to look for multiple ideas instead of ideals. Looking for ideas requires form appearing and disappearing momentarily. So design mechanisms that would trigger surprises for their observers, should allow divergence and changing vocabularies in continuum.

Having set the framework of a design machine capable of forming dynamic descriptions, the first step was to focus on how the computational system can algorithmically decompose a phenomenon into discrete parts. The computational model of self-organization I experimented on seems promising and it definitely provides directions for future research. One important direction is to explore how different self organizing maps that run at the same time classifying different properties of the same object could be associated in a meaningful way.

This approach could provide the machine with the ability to learn how to generate primitives in an unsupervised manner. However I would like to mention that on the first place trying to define the basic operations, the properties by which the machine in our case "reads" an image, is a way to define explicitly how the machine interprets a phenomenon. It happens at a very elemental level but still is a way of predefining closed loops between the machine and the environment. The possibility of unexpected, surprising behaviors to appear depend from that point on, on the unexpected associations and the fact that the machine eventually will exist in a real complex environment.

Another possible scenario for experimentation would be to rethink the cyclical process of interaction among the machine and the environment. As it is proposed in the course of the thesis, is characterized by the notion of a central system with perceptual modules as inputs and action modules as outputs. The important difference is that the symbolic description of the world is not hand crafted by the designer of the machine but intrinsically generated by the

system through the interaction with the environment which for the moment includes only its ability to "see". However it takes "a long chain of modules to connect perception to action" (Brooks, 1991). Maybe a direction for future research could explore Brooks' approach (1991) on decomposing a system without making distinctions between peripheral systems, such as vision, and central systems, but instead built layers of activities.

I will conclude by bringing back the concept of intrinsic descriptions and my remark that predefining the elemental operations the machine uses to interpret a phenomenon restricts the behavior of the system. The concept of building intrinsic descriptions and changing primitives implies "a form of learning of new categories rather than learning within existing categories" (Varela, as cited in Cariani, 1998). This leaves me with the aspiration: Would it be possible for the system to learn how to build its own interpretation operations rather than having them defined in advance?

Bibliography

Alexander, C. (1964). Notes on the synthesis of form. Cambridge: Harvard University Press.

Braitenberg, V. (1984). Vehicles, experiments in synthetic psychology. Cambridge, Mass: MIT Press.

Brooks, R. A. (1991). Intelligence Without Representation. Artificial Intelligence, 47(1-3), 139-159.

Cariani, P. (1998). Epistemic Autonomy through Adaptive Sensing. Retrieved from http://homepage.mac.com/cariani/CarianiWebsite/ISAS98.pdf

Cariani, P. (2006). On creating new informational primitives in minds and machines. Symposium on "Creativity: The Mind, Machines, and Mathematics". Retrieved from http://homepage.mac.com/cariani/CarianiWebsite/Cariani-Templeton2006.pdf

Eisenman, P. (1984). The End of the Classical: The End of the Beginning, the End of the End. Perspecta, 21, 154-173.

Gips, J., & Stiny, G. (1978). Algorithmic aesthetics: Computer models for criticism and design in the arts. Berkeley: University of California Press.

Glanville, R., (1997). Behind the Curtain. Ascott, R (ed) Procs of the First Conference on Consciousness Reframed, UCWN, Newport, 1997

Goodnough, R. (1951). Pollock Paints a Picture. ARTnews, L(3). Retrieved from http://www.artnews.com/issues/article.asp?art_id=2401

Karmel, P. (1998). Pollock at Work: The Films and Photographs of Hans Namuth. In P. Karmel & K. Varnedoe (Eds.). Jackson Pollock (pp. 87-137). New York: Museum of Modern Art.

Knight, T., & Stiny, G. (2001). Classical and non-classical computation. Arq: Architectural Research Quarterly, 5(4), 355-372.

Knight, T. (2003). Computing with emergence. Environment and Planning. B, Planning & Design, 30, 125-156.

Larson, S. D. (2003). Intrinsic Representation : Bootstrapping Symbols From Experience (Master's thesis). Available from DSpace@MIT database. (Umi No. 57124903)

Milne, A. M. (1971). CLUSTR: A Program for Structuring Design Problems. Proceedings of the 8th Design Automation Workshop: Annual ACM IEEE Design Automation Conference. Atlantic City, NJ: ACM, 242-249.

Milne, M. (1975). Whatever Became of Design Methodology?. In N. Negroponte (Ed.), Reflections on computer aids to design and architecture (pp. 30-36). New York: Petrocelli/Charter.

Minsky, M. L. (1967). Computation: Finite and infinite machines. Englewood Cliffs, N.J.: Prentice-Hall.

Negroponte, N. (1970). The architecture machine; toward a more human environment. Cambridge, Mass., M.I.T.: Press.

Negroponte, N. (1975). Soft architecture machines. Cambridge, Mass.: The MIT Press.

Negroponte, N. (1966). The computer simulation of perception during motion in the urban environment. (Master's thesis). Available from DSpace@MIT database. (Umi No. 13288)

Reas, C. E. B. (2007). Beyond Code. In A. Burke & T. Tierney (Eds.), Network practices: New strategies in architecture and design (166-177). New York: Princeton Architectural Press.

Reddy, M.J. (1993). The Conduit Metaphor: a Case of Frame Conflict in our Language about Language. In A.Ortony (Ed.), Metaphor and Thought (2nd ed.) (164-201). Cambridge: Cambridge University Press.

Schön, D. A. (1987). Educating the reflective practitioner (1st ed.). San Francisco: Jossey-Bass.

Simon, H. A. (1996). The sciences of the artificial. Cambridge, Mass: MIT Press.

Stiny, G., & March, L. (1981). Design Machines. Environment and Planning B, 8, 245-255.

Stiny, G. (1998). New Ways to Look at Things. Environment and Planning B: Planning and Design, Anniversary Issue, 68-75.

Stiny, G. (2006). Shape : Talking about seeing and doing. Cambridge, Mass.: MIT Press.

Sutherland, I. (1975). Structure in Drawings and the Hidden-Surface Problem. In N. Negroponte (Ed.), Reflections on computer aids to design and architecture (pp. 73-77). Location: New York: Petrocelli/Charter.

Verbeeck, K. (2006). Randomness as a generative principle in art and architectur (Master's thesis). Available from DSpace@MIT database. (Umi No. 71790501)

Figure references

Figure 2: Alexander, C. (1964). Notes on the synthesis of form. Cambridge: Harvard University Press. 62

Figure 3: Negroponte, N. (1970). The architecture machine; toward a more human environment. Cambridge, Mass., M.I.T.: Press. 85-86

Figure 4: Negroponte, N. (1970). The architecture machine; toward a more human environment. Cambridge, Mass., M.I.T.: Press. 42

Figure 5: Gips, J., & Stiny, G. (1978). Algorithmic aesthetics: Computer models for criticism and design in the arts. Berkeley: University of California Press. 22.

Figure 7: Reddy, M.J. (1993). The Conduit Metaphor: a Case of Frame Conflict in our Language about Language. In A.Ortony (Ed.), Metaphor and Thought (2nd ed.) (164-201). Cambridge: Cambridge University Press. 172

Figure 8: Karmel, P. (1998). Pollock at Work: The Films and Photographs of Hans Namuth. In P. Karmel & K. Varnedoe (Eds.). Jackson Pollock (pp. 87-137). New York: Museum of Modern Art. 93-94.

Figure 9: Stiny, G. (1998). New Ways to Look at Things. Environment and Planning B: Planning and Design, Anniversary Issue, 69.

Figure 10: DeBiswas, K. & Toulkeridou, V. (2010, ongoing work). A Computational Model of Artistry.

Figure 11: DeBiswas, K. & Toulkeridou, V. (2010, ongoing work). A Computational Model of Artistry.

Figure 12: DeBiswas, K. & Toulkeridou, V. (2010, ongoing work). A Computational Model of Artistry.

Figure 13: Retrieved from http://www.marcdatabase.com/~lemur/lemur.com/gallery-of-antiquarian-technology/horology/nbs-atomic-clock/nbs-history-476-atomic-clock-photo-1200-scale-1024x878.jpg

Figure 16: Retrieved from http://www.gramaziokohler.com/web/e/projekte/52.html

Figure 17: Retrieved from http://www.gramaziokohler.com/web/e/projekte/52.html

Figure 19: Braitenberg, V. (1984). Vehicles, experiments in synthetic psychology. Cambridge, Mass: MIT Press. 7

Figure 20: Braitenberg, V. (1984). Vehicles, experiments in synthetic psychology. Cambridge, Mass: MIT Press. 11

Figure 21-24: Retrieved from http://reas.com/texts/beyondcode.html

Figure 26-29: Negroponte, N. (1970). The architecture machine; toward a more human environment. Cambridge, Mass., M.I.T.: Press. 108, 110

Figure 30: Brooks, R. A. (1991). Intelligence Without Representation. Artificial Intelligence, 47(1-3), 152.

Figure 33: Larson, S. D. (2003). Intrinsic Representation : Bootstrapping Symbols From Experience (Master's thesis). Available from DSpace@MIT database. (Umi No. 57124903). 39