Foreword

Algorithmic Architecture or the Computer as a Double?

What should be the exact scope of the computer's involvement with architectural design? This question has been present since the beginning of computer aided architecture. It played, of course, a fundamental role in the first reflections and experiments regarding a possible computed or cybernetic architecture in the 1950s and 1960s. It did not disappear with the advent of post-modernism. The latter's concern with the linguistic dimension of architecture and urban design and the possibilities of formal exploration offered by the computer went hand in hand[1]. It is only during the last decade, with the spectacular development of computer graphics and the fascination exerted by the strange forms, the blobs and others that began to float on the designers' screens that this question was momentarily suspended. Now that this fascination is beginning to fade, the issue is back with all its complexity. There is no better proof of it than *Algorithmic Architecture*, since this book is primarily addressing the problem both at a technical and at a philosophical level.

Typically, the positions regarding the role of the computer in architectural design fall into two categories. For many designers, the computer is just an advanced tool running programs that enable them to produce sophisticated forms and to control better their realization. For those designers, although the machine does alter significantly the nature of the architecture that is produced, it is not necessary or even desirable to enter into the details of its inner processes. Despite their claim to the contrary, the greatest part of the blob architects fall into this category. Kostas Terzidis belongs clearly to the other camp

composed of those who think that it has become unavoidable to enter into the black box of programming in order to make a truly creative use of the computer. In this perspective, a large section of his book is devoted to the exploration of what the mastery of scripting techniques can bring to architecture.

More than these technical insights, the main interest of *Algorithmic Architecture* may very well lie in the relation it establishes between the detailed examination of the possibilities offered by the computer and more general interrogations, of a philosophical nature, on the design process.

One of Terzidis' fundamental tenets is that design is not properly an invention, the creation of something absolutely new. It should rather be considered as the result of an unveiling or a rediscovery process. There is something almost pre-Socratic, or, to take a reference closer to our time, neoclassical, in this conception of design as a kind of return to an existing state of things that has fallen into oblivion. The pre-Socratic perspective would be to consider after Empedocles or Parmenides that nothing comes out of nothing and that the new is just the extant seen from a different vantage point. Neoclassical aesthetics and design theory starts as for it from the assumption that the quest for beauty is about recapturing the fresh inspiration that prevailed at some point towards the origin of art, an inspiration that accounts for the enduring value of archetypes. Part of Terzidis' ambition lies precisely in rethinking some of architecture's most fundamental archetypes in the light provided by computation.

There is something both disturbing and stimulating in a conception of design centered on the unveiling or the rediscovery. The disturbance becomes even more profound when Terzidis tells us that we shouldn't consider the computer as an extension of the mind, but rather as a partner in the design process with fundamentally different aptitudes and ways to reason. The computer is the Other of the human mind, not its mirror. There, the possible points of reference are to be found rather in the first years of computer aided architecture, when pioneers like Nicholas Negroponte, the founder of the Massachusetts Institute of Technology Media Lab, were presenting the introduction of computing in the design process as a

dialogue between partners or "associates"[2]. However, Terzidis' perspective differs because of its insistence on the radical otherness of the computer.

But is this radical otherness fundamentally different from the estrangement from ourselves we experience day after day in the midst of any creative process? Something comes out from a region of our mind that we generally don't know much about, something that is both intriguing and secretly familiar. Kostas Terzidis is probably right in underlining the resemblance between invention and recognition. Without this familiarity, the new would be literally unthinkable. Its newness is nevertheless the product of an interior estrangement somewhat comparable to the distance that separates us from the computer.

At that stage, one might argue that the algorithmic procedures of the machine still remain fundamentally different from the way we think. But there again, a closer examination reveals a more ambiguous situation. For our mind follows rules in order to avoid the excessive familiarity that might otherwise defuse the originality of the creative endeavor, and these rules are usually as constraining as the algorithmic procedures run by the computer. In other words, the otherness that Terzidis attributes to the machine is also present in ourselves, in the apparent opposition between the creative impulse and the set of rules that enable us to control it. Is this opposition real? We know that rules can trigger imagination and that spontaneity always obeys to some hidden principles. The dichotomy between the spontaneous and the regulated has more to do with a polarity than with a clear-cut separation between two opposite faculties. One part of our inner self is constantly escaping regulation while the other tends to function in an almost mechanical way. Towards the end of the eighteenth century, the father of the *Encyclopédie*, the French philosopher Denis Diderot, was already wondering up to what point the mind is unpredictable and to what extent it could be compared to a machine[3]. Two centuries later, we are still in the midst of this conundrum.

The genuine excitement that Kostas Terzidis' book provokes might very well have to do with the perspectives he offers on this fundamental question. What if the radical other revealed by the computer was actually inside us,

waiting for the machine to actualize it? If such was the case the computer might very well be more like a mirror showing us a reflection or a possible double of ourselves than a creature from another world. One thing is sure; architecture always appears as a compromise between rules and their contrary. Its expressive power might very well have to do with the secret analogy between this hybrid status and the intimate nature of our creative process. Ultimately, Kostas Terzidis' *Algorithmic Architecture* is about this analogy. It is not a book on computer and architecture. It is a book on architecture.

Antoine Picon
Professor of the History of Architecture and Technology,
Graduate School of Design, Harvard University

Endnotes

[1]There is more generally a strong link between the perspectives opened by computation in architecture and the emergence of contemporary architectural theory; for further discussion, see A. J. Magalhaes Rocha, *Architecture Theory 1960-1980: Emergence of a Computational Perspective*, doctoral dissertation submitted to the MIT Department of Architecture, Cambridge, Massachusetts, 2004.

[2]See for instance N. Negroponte, "Towards a Humanism through Machines", in *Architectural Design*, September 1969, pp. 511-512.

[3]Cf. J. Proust, *Diderot et l'Encyclopédie*, Paris, Armand Colin, 1962.

Prologue

Computation is a term that differs from, but is often confused with, computerization. While computation is the procedure of calculating, i.e. determining something by mathematical or logical methods, computerization is the act of entering, processing, or storing information in a computer or a computer system[i]. Computerization is about automation, mechanization, digitization, and conversion. Generally, it involves the digitization of entities or processes that are preconceived, predetermined, and well defined. In contrast, computation is about the exploration of indeterminate, vague, unclear, and often ill-defined processes; because of its exploratory nature, computation aims at emulating or extending the human intellect. It is about rationalization, reasoning, logic, algorithm, deduction, induction, extrapolation, exploration, and estimation. In its manifold implications, it involves problem solving, mental structures, cognition, simulation, and rule-based intelligence, to name a few.

The dominant mode of utilizing computers in architecture today is that of computerization; entities or processes that are already conceptualized in the designer's mind are entered, manipulated, or stored on a computer system. In contrast, computation or computing, as a computer-based design tool, is generally limited. The problem with this situation is that designers do not take advantage of the computational power of the computer. Instead some venture into manipulations or criticisms of computer models as if they were products of computation. While research and development of software involves extensive computational techniques, mouse-based manipulations of 3D computer models are not necessarily acts of computation. For instance, it appears, from the current discourse, that mouse-based manipulations of control points on NURBS-based surfaces are considered by some theorists to be