**Cyphers: On the Historiography of Digital Architecture**

**Bottazzi, Roberto**



https://doi.org/10.34737/vwq21

University of Westminster – Faculty of Architecture and the Built Environment:

PhD by published work

# Cyphers: On the Historiography of Digital Architecture

Author:

*Roberto Bottazzi*

Supervisors:

Dr. Victoria Watson

Richard Difford

*Cyphers: On the Historiography of Digital Architecture*

*Roberto Bottazzi*

*A thesis submitted in partial fulfilment of the requirements of the University of Westminster for the degree of Doctor of Philosophy*

*London, May 2022*

**Abstract:**

This dissertation reflects on the methods and concepts employed in constructing a history of digital architecture. By focusing on the methodological issues, it complements and expands the research developed for the monographic study *Digital Architecture Beyond Computers* (*DABC*) and the book chapter "Crypto Architecture". In both pieces digital architecture is understood to cover a period of time that stretches well beyond the appearance of the modern digital computer (after World War Two). The notion of computing numbers and symbols to apprehend and intervene in our reality is in fact a much older idea than the invention of the modern digital computer. This dissertation reflects on the approach suggested by both writings by analysing the conceptual basis of computation in order to devise an appropriate historiographic approach to digital architecture. The aim of the investigation is to move beyond a technologically-driven, utilitarian view of computation in favour of a more conceptual position that foregrounds computation's fundamental logic and the role of the disciplines that informed and continue to inform it. This broader perspective aims at establishing a relation between the artifacts and the processes of digital architecture; that is, between what digital architecture is (which *DABC* explores through case studies in which computation and design affected one another), and how it is generated (the techniques and methods deployed to design architecture).

This dissertation introduces a specific conceptual figure to articulate the historiography of digital architecture: the cypher. Cyphers address the fundamental challenges emerging from constructing a history of digital architecture, they organise the vast collections of case

studies forming the history of digital architecture, foreground the conceptual motivations behind computation, and acknowledge the role that different disciplines (philosophy, logic, semiotic) have played in shaping what we call digital architecture.

**Table of Content**

**Acknowledgments**

I would like to profoundly thank my supervisors Dr. Victoria Watson and Dr. Richard Difford for their insights, rigour, and advice in guiding the development of the research. I would also like to thank Stefania Boccaletti for the constant support and invaluable feedback on all the pieces of research submitted in this thesis.

**Author's Declaration**

I declare that all material contained in this thesis is my own work.

**Introduction**

On October 23, 2018 auction house Christie sold the painting *Portrait of Edmond Belamy* by Obvious for $432,500, nearly 45 times its estimated price. The shocking news, though, was that the 'author' of the painting was not human but rather an Artificial Intelligence [AI] algorithm called GAN (Generative Adversarial Network). (Cohn 2018) Much of the debate that followed concentrated on questions of authorship of an algorithmic painting and on the painting's aesthetic merits. On this latter point, on the basis of formal resemblances between the two, commentators saw similarities between GAN-generated art and Surrealism. This debate seemed rather myopic as the qualities of GAN as a legitimate extension or renovation of the canons of Surrealism appeared to be a rather secondary issue compared to the novelty of the methods employed to produce the artefact. In other words, the mode of critique adopted was blind to the novel methodology of the GAN painting and implicitly introduced a clear division between the technique employed to make the painting and its final appearance. In GAN, as well as computational art in general, the algorithmic process deeply affects the final outcome: from the type of algorithm chosen to the type and quantity of data processed by the algorithm, all can radically change the final product. In short, in computation process and product cannot be distinguished. The reaction of art commentators is indicative of a more profound problem with critiquing digital artifacts in a manner that generally severs outcomes from processes. The book chapter (Bottazzi 2019) and monographic study *Digital Design Beyond Computers* (*DABC*) (Bottazzi 2018) which this dissertation illuminates and supports trace my efforts to develop intellectual instruments to critically analyse computational design, detect the relations

between technique and outcome, and provide a more mature, critical, and precise account of what is at stake when designing with computers.

This dissertation extends the discussion beyond the domain covered in *DABC* to provide methodological instruments relevant for both historical studies on digital architecture and contemporary issues in computational design (particularly the introduction of AI methods). The aim is to foreground the deeper motivations behind the use of digital technologies in design; a task that can shine some light on both historical case studies and the contemporary debate. Whilst this introduction will more precisely define the key terms employed in this dissertations, we can preliminarily establish that all the technologies discussed in *DABC* share the capacity to compute. The modern digital computer is only one of such technologies – though by far the most powerful – which also include: pantographs, perspective machines, graphs, and tree diagrams, to mention a few. Expanding the timeframe of what can be considered digital architecture opens up new possibilities. For the present point of view, such deep perspective breaks the uncritical equation computation = digital computers and questions the fetishisation of digital technologies to provide more refined instruments to discern genuine novelties. This move frees computation from a rather technical, utilitarian reading in favour of an intellectual approach that emphasises the techniques through which architectural thinking can be articulated as well as the influence on computation of several disciplines. Finally, existing case studies can be critically re-assessed and new scholarly knowledge can emerge.

What distinguishes the history traced in *DABC* from others is the adjective *digital* which acts as a discriminating category for appraising the vast historical production of

architecture. Digital is here understood not only as a particular type of technology (defined in this introduction), but also as a new frame of reference, a lens through which to re-read buildings, techniques, and designers whose work embodied computational processes which were however disregarded when analysed. For instance, Paolo Portoghesi's work has been thoroughly investigated by architectural critics and historians, however his "Teoria dei Campi" (field theory) has never been read as an anticipation of themes developed by digital architects in the 1990's. (Bottazzi 2018, pp. 50-53) To add the adjective digital to a history of architecture signifies a change in prospective that can be illustrated through a computational analogy. The application of two different algorithms to the same datasets will return distinct results; similarly, we can see the history of architecture as a fixed database, a set of events located in the past, which each type of algorithm will mine and remap according to different criteria. Such criteria crystallise around the figure of the paradigm first (chapter two) and then that of the cypher (chapter three); neither of these figures were analysed in *DABC* and form the core of this dissertation. To press our computational analogy further, we can observe that algorithms and outcomes are closely related; in surveying a dataset, the process followed directly determines what we will find and how such findings will be organised. Though the example chosen here is basic (but the *Portrait of Edmond Belamy* offers a much more complex example), it already indicates that in computation process and result are co-determined and speaking of them separately would profoundly distort any analysis. This is the position adopted in this dissertation and the main thesis that the following chapters will seek to articulate.

The search for historiographical instruments to articulate the complexity of formulating a history of digital architecture is also a response to the limited attention that this area of scholarly research has received so far. Studies on the history of digital architecture have often concentrated on either a specific period (Carpo 2001, Halpern 2014) or on the implementation of actual digital computers in architectural design (Cardoso Llach 2015). The domain surveyed in *DABC* is rather like an open territory lacking historiographic considerations for its articulation. Paradigms – which chapter two is dedicated to – are effective conceptual instruments to deploy to chart open, uncertain domains as they "can guide investigation in the absence of rules". (Kuhn 1970, p. 42) Paradigms in fact provide an organisational principle that is not based on rules, but rather on a dynamic, flexible framework that can begin to unravel the long and multi-disciplinary history of digital architecture. The use of paradigms as historiographical tools is unquestionably linked to Thomas Kuhn's (1970) work on the structure of scientific revolutions. Ever since the publication of this seminal book, a vast body of scholarly research has been working to extend the domain of application of paradigms beyond the sciences to also include the humanities. Particularly, the work by Giorgio Agamben (2009) on paradigms enriches Kuhn's analysis by not only linking it to the work of philosophers (Michel Foucault, Aristotle, etc.), art historians (Aby Warburg), or semioticians (Émile Benveniste), but also by understanding paradigms as methodological figures fusing theory and practice as well as process and output. These observations were essential in devising a structure for *DABC* as they offered a conceptual framework to operate in the uncharted territories of the historiography of digital design and to link processes to products. However, in order to apply paradigms to

computational studies on architecture an additional figure needed to be introduced: the cypher. Cyphers represent an evolution of paradigms because they are mathematical instruments, and, contrary to paradigms which pertain to the humanistic tradition, they are native to computation and can be employed directly without mediation. Consistently with the approach taken in *DABC*, this dissertation also foregrounds the conceptual roots of computation to grasp what is at stake in utilising cyphers as conceptual frameworks. This task requires development of an argumentation that departs from the world of diplomacy – which cyphers are commonly associated with – to deal with the mathematical and semiotic properties of cyphers and how they can engender patterns, regularities, and even permit order to surface. What emerges is a delineation of cyphers as instruments to navigate and structure open domains by providing intelligibility out of randomness. This dissertation discusses both how cyphers can be understood in historiographic terms and how they can be of relevance to the contemporary debate in the fields of digital architecture and urbanism, particularly in providing critical instruments guiding the absorption of AI methods by spatial design disciplines.

Before venturing any further, it is necessary to define more clearly the domain of investigation within which this dissertation operates by defining the three key terms at the core of *DABC* and "Crypto Architecture: Notes on Machine Learning and Design," which are: 'architecture,' 'history,' and 'digital.'

Of the three words, architecture is the perhaps the one requiring least explanation. In both *DABC* and this dissertation, architecture is first and foremost understood as an intellectual activity analysing and manipulating the built environment which extends beyond

the art of erecting structures to include design methods and the technologies employed to conceive them. Such technologies are in fact the main focus of *DABC* as design methods instrumentalise ideas by conflating process and practice in a way that is thought analogous to computation.

Historiographic issues are central to this dissertation and demand a dedicated space (paradigms are dealt with in chapter three and cyphers in the final chapter) to unravel them. The focus of the discussion, however, is digital architecture and issues related to the philosophy of history are only discussed insofar as they contextualise the arguments exposed here.

By digital we commonly mean both a particular kind of technological artefact-(digital computers, peripherals, networks, smartphones, etc.) and a specific type of computation that exclusively employs discrete quantities.[1] We will maintain such distinction by speaking of, respectively, digital computers and discrete computation. Regardless of the functions they perform, digital devices employ some form of discrete computation. In general, computation is understood to include any act of calculation independently of the agent carrying it out. *DABC* and this dissertation only discuss types of computations performed by machines. Machine computation further divides into analogue and discrete based on

[1] The philosophical debate on the differences between analogue and digital representation is complex and long-standing. Several key texts have been reviewed in this research (Goodman 1976, Haugeland 1981, Katz 2008, Lewis 1971, Maley 2011, Papayannopoulos 2020, Schonbein 2013) to eventually align this dissertation with the classification proposed by Corey J. Maley. Compared to other authors, Maley makes a further distinction between discrete and digital, where the latter is concerned with a particular procedure to write numbers. This position offers a clear distinction between terms such as discrete and digital which other authors overlap or conflate. From reviewing the key literature on the subject, it emerges that the definition of discrete computation (separate and distinguishable symbols) is uncontroversial and provides a solid position to argue from.

whether calculations are performed using continuous or discrete quantities. *DABC* discusses examples of both analogue and discrete computation, whereas this dissertation only concentrates on the latter. This is because discrete computation includes and extends analogue computing machines and propels any digital artifact currently used to design architecture. When we speak of discrete computation we refer to the model outlined by Alan Turing (1936) and now referred to as the Universal Turing Machine (UTM). The UTM is a discrete computer which recursively processes inputs according to a predetermined set of rules. The UTM is characterised by a number of properties relevant for this dissertation: it is discrete (it computes finite quantities in the form of symbols), economic (it uses binary numeration, the shortest set of signs to represent), and universal (it computes anything that can be computed).

Evens (2014) points out that even digital computers are ultimately analogue machines since they store bits of information (represented by the discrete symbols of 0 and 1) by taking advantage of continuous physical properties such as electrical voltage. Even if voltage varies in a continuous fashion, an intellectual operation performed through code overrides the physical properties of electricity to arbitrarily assign a 0 or a 1 to certain voltage ranges. This is an important point for this dissertation as it indicates that discrete computation consists of an arbitrary system of signs, an intellectual artefact to apprehend reality and, eventually, intervene in it. From this consideration it follows that to grasp what is at stake in digital devices we must look beyond their technical complexity, which unavoidably limits our reading to mere functional issues, to concentrate on what concepts have been embodied into digital computers. Rather, computation is first and foremost an

intellectual project stemming from the combination of philosophical speculations and mathematical instruments to apprehend, represent and intervene in our world. Furthermore, utilitarian readings of computation tend to link computers to a specific historical period which runs the risk of belittling the intellectual project that sustains the development of computation. Histories of technology, however, indicate that computation is a project as old as culture and has always transcended specific historical moments (Borzacchini 2005, Galloway 2021, Rossi 1960).

Returning to the UTM, the arbitrary nature of the binary system of signs used by a theoretical machine such as this invariably implies issues of translation. Binary code is always 'meaningless', an arbitrary representation that needs to be made intelligible through a translation, that is, an act of design. Not only does this consideration explain why cryptography is such a key concern in discrete computation, it also foregrounds how intelligibility emerges out of randomness. This is also a central issue that will run through the whole discussion and will constitute the central point of the chapter on cyphers.

The dissertation is structured in three chapters which progress from defining the domain of the investigation to proposing a methodological approach to studies on the history of digital architecture. The first chapter outlines four essential characteristics of history and philosophy of computation (non-chronological, non-fragmentary, transversal, and speculative). These four characteristics form a critical reading of computation that opens up a conceptual space where the notions of paradigm and cypher can be articulated. The second chapter is dedicated to paradigms which are identified as methodological figures able to address the points raised in the first chapter and foreground the deeper motivations

behind the integration of computational thinking in architecture. The discussion will largely rest on Thomas Kuhn's seminal book on paradigms and Giorgio Agamben's (2009) key text also on the same topic. Most importantly, whereas the work of the two philosophers mentioned above seeks to generalise the notion of paradigm, the second chapter will proceed in a different direction by thinking about paradigms in regard to the particular issues arising from the historiography of digital architecture. Debates on the philosophical approaches to history which are central for the two authors are here only employed insofar as they support or contextualise the arguments put forward. The final chapter introduces the figure of the cypher as an evolution of the paradigm; that is, as a synthesis of the computational, spatial, semiotic, and historiographic issues introduced in the previous two chapters. The overall journey illustrated in this dissertation aims at foregrounding the deeper, persisting, intellectual motivations behind the idea of computing and how this can affect our reading of history and architecture.

Chapter 1

## A critical overview of computation in design disciplines

The following chapter outlines some key features of computation that need to be accounted for when surveying the history of its relation with architecture. The four points raised here intend to open up a conceptual space foregrounding the characteristics of discrete computation and its history, so that they can play an active role in shaping the historiography of digital architecture. If discrete computation brings closely together process and product, we need to identify some key directions along which this happens and what implication for digital architecture it might have. The next two chapters on respectively paradigms and cyphers will provide a conceptual framework addressing the four points raised here.

### *Computation is not chronological*.

From an historical point of view, digital architecture conflates two fields – architecture and computation – whose origins, scopes, and developments are very different from each other and that have only recently merged (Hovestadt 2020) – relative to the length of their respective histories. The modern digital computer is the result of research carried out during WWII, it has a much shorter history than architecture. The profound differences between the two fields made their encounter opportunistic rather than the result of a joined effort.

The history of computation is far from being a linear one as it went through periods of intense experiments and discoveries (e.g., the renaissance with the emergence of new

cryptographic methods, memory palaces, and perspective machines) to others in which the relation between computation and innovation was dormant. A chronological account would struggle to reveal more than the sum of the events listed, potentially obscuring the more profound reasons behind the attempts to compute phenomena, spaces, or knowledge. More fundamentally, adopting a chronological method poses the issue of fixing origins and identities; this task proves very elusive in the case of computation whose origins seem to be as old as civilisation itself and involve a number of different disciplines. The method adopted in *DABC* and discussed in this dissertation rejects a closed reading based on identities to propose an open and adaptable approach that privileges paradigmatic moments in favour of foundational ones. This approach not only avoids fixing origins, but it also acknowledges the impact that different fields had on computation by concentrating on key moments in which architecture and computation converged to affect one another.

*Computation is not fragmentary*.

Though an approach through paradigms is more agile and accounts for different computational cultures and practices, it still leaves open questions regarding how to extract underlying, persistent traits of computation and its application to design disciplines beyond the sheer accumulation of fragments. This dissertation foregrounds some aspects, such as the transversality, the multi-disciplinary, speculative nature of computation that form a continuous background against which different design experiments emerge. The research carried out for *DABC* showed that the most common format for studies on the history of digital architecture is exemplified in the edited collection of essays (Goodhouse 2017; Lynn

2013; Oxman Oxman 2014;Vardouli Touloumi 2019; Zeynep Celik May 2019). If, on the one hand, this format also confirms the difficulties emerging from adopting a chronological account for the history of digital architecture; on the other, the very same format side-lines any discussion on, or proposal for a comprehensive approach to the history of digital architecture. The multi-authored nature of collections of essays unavoidably strives towards a diversity of approaches, but, at the same time, also eludes more profound questions regarding what is at stake when computation is applied to design. As a single-authored monographic study, *DABC* provided the ideal format for methodological issues to emerge and be confronted. The articulation of chapters according to paradigmatic technologies of computation presents an initial model for how to negotiate between fragments (i.e. individual case studies) and underlying themes in computation (i.e. the topic informing each chapter). The model allows for the merging between content and process, between disciplines in a way that seems better suited to foreground the intellectual opportunities presented by the penetration of computational techniques in architecture.

*Computation is transversal.*

The transversality of computation can be understood in three different ways. Contemporary computational culture infiltrates many different fields to re-organise them: this is the case for medicine, transportation, politics, and, of course, architecture, to mention a few. Be it through computers, the Internet, mobile communication, etc., there is basically no discipline which has not been impacted by computers. Computation – the 'engine' behind how digital computers work – is therefore not necessarily a new discipline,

but rather a way of doing things affecting all the fields it comes into contact with. In architecture one simple example to grasp the depth of this transformation is represented by the way CAD changed architectural representation, particularly, the construction and use of perspectival views. CAD-generated perspectives still abide by the same pre-digital mathematical operations utilised since the eighteenth century. Computation, however had a massive impact on the production of perspectival views by drastically reducing the amount of effort necessary to complete such task. Prior to the introduction of CAD, perspectival views were most likely generated at the end of the design process as a significant amount of time (and, to a certain extent, knowledge) needed to be invested to successfully master such art. CAD removed all these constraints by generating three-dimensional views of a piece of architecture instantaneously through a simple mouse-click. As a result, perspectival views became an integral part of the design process, and it is in fact common to model objects through a perspectival rather than orthographic window of a CAD modeller. Similar considerations can extend to other domains of design such as the treatment of curves.

The transversality of digital computation can also be understood at a much more fundamental level. To be computable, all media objects – audio, images, CAD models, etc. – must be translated into strings of binary code (specifically, numbers). Here computation cannot really be equated with natural languages[2], but rather as a set of procedures to format, organise, and transform all media objects into discrete signs (and vice versa). Once

[2] Salanskis (2011) points out that the three major works laying the foundations of the architecture of digital computers (Godel's (1931), Turing's (1936-37), and Church's papers (1936)) achieved similar results by adopting slightly different mathematical procedures. According to Salanskis, this fact indicates that digital computation cannot be really formalised as language but only described as a set of similar procedures related to calculations.

again, more than a means to an end, computation is better described as a method, a way of structuring objects to make them amenable to calculation, regardless of the final aim. The UTM is universal because its computation is universally applicable or, more precisely, any finite computation can be re-written into a UTM. Jean-Michel Salanskis (2011) links the capacity of digital computation to organise information and process it to its *meaningless-ness*: that is, the focus of computation is on procedures, on syntax rather than semantics. Stephen Wolfram reformulated the Turing-Church thesis through his Principle of Computational Equivalence by stating that "almost all processes that are not obviously simple can be viewed as computations of equivalent sophistication" (Wolfram 2002, pp. 716-717). Beside the ontological implications of this argument which are not the focus of this dissertation, from Wolfram's remark we can infer that "computation is therefore simply a question of translating inputs and outputs from one system to another." (Rowland Weisstein n.d.) In the second and third chapters of this dissertation, the pair paradigm-cypher will help conceptualise such processes of translation and what implications it might have for contemporary computational design.

Finally, computation can also be considered transversal as it emerges from the conflation of different fields. The history of computational technologies shows how different disciplines contributed to its development: primarily mathematics, logic, and philosophy, but also semiotics and rhetoric. Computation results from the merging of scientific and humanistic knowledge, a genealogy that cannot be grasped if solely understood from a technological perspective. The method of inquiry should therefore be able to unravel all these different strands: from technical innovations and philosophical ideas to material

properties and how they came together and transformed each other. As we will see in the next chapter, the notion of the paradigm provides an adequate conceptual framework to move between domains and detect the contributions made by different disciplines.

*Computation is speculative.*

Luciano Floridi (2020) reminds us that digital technologies are often compared to natural languages because they enable communication (for instance, people and machines take advantage of the Internet or social media to exchange messages). However, Floridi suggests that if this comparison is to hold, the digital should also possess two additional properties of language: that of describing phenomena and conceptualising them. He concludes that a truly digital culture will not emerge until the representational and conceptual qualities of the digital will be also grasped. Though Floridi's remarks speak of the digital in general terms, they are still valid if we limit the discussion to the narrower subject of discrete computation discussed in this dissertation. How is discrete computation descriptive and, subsequently, generative? How can it conceptualise objects and what affordances does it engender? These questions are central to this dissertation in which computation is understood as transversal, a discipline based on the manipulation of signs.

To begin addressing these issues, we have to return to the material basis of computation as they manifest themselves at the level of signal – the bit – and of logic, that is, the range of possibilities discrete computation offers as mechanisms for generative and speculative considerations. First we need to address the main difference between analogue

and discrete computation analysed from the point of view of design technologies. Perspective machines such as the ones designed by artists in the renaissance are good examples of analogue computational devices as they offer useful insights on the properties of analogue computing. Despite belonging to a distant past, there are different reasons to still consider them in a discussion on computational design. First, perspective machines applied computational techniques to spatial problems with the view to evolve both artistic and architectural expression. The introduction of linear perspective marked a paradigm shift in art affecting both the techniques and meaning of artistic production. Finally, perspective machines were mechanical devices in which the type of analogue computation performed was very simple (the analogue computers of the 1930's and 1940's would take advantage of electricity and electromagnetic properties of matter making their computation much more complex and invisible to human eyes). In short, perspective machines are technologically radically different from digital modern computers. If, on the one hand, a comparison between perspective machines and modern digital computers may appear unbalanced, on the other, the stark differences between the two types of devices will facilitate distinctions and comparisons.

Among the plethora of designs for perspective machines, Dürer's incision *Man with Lute* (1523) perhaps depicts the most accomplished example of such contraptions as the process of drawing a perspectival view of the lute is almost entirely automated (except for the human figure moving the needle along the surface of the musical instrument).

Perspective machines were by all accounts, computational devices that, contrary to modern digital computers, computed by measuring lengths, angles, etc. As is the case for

mechanical analogue computers, semiotically, perspective machines could be classified as iconic signs; more precisely, they were diagrammatic icons as they captured the ideal relation between phenomena and their representing sign (Pierce, 1998). The way in which perspective machines signified was not based on an arbitrary pairing between phenomena and signs, rather each material was chosen and parts modelled to best translate the theories of mathematical perspective in physical form. In other words, an informed observer could potentially infer the theory embodied in the device by simply looking at it. Perspective machines (which we now consider as very simple analogue computers) were not cryptic, they had certain pedagogical qualities deriving from the lack of abstraction of their computation, whereas, on the other hand, abstraction forms one of the central tenets of symbolic, discrete computation. One of the issues of analogue computation is that iconic signs are in principle compromised by material debasement. Perspective machines were no exception as they would not operate correctly, or at all, if materials were worn out or connections between parts stopped functioning. Finally, perspective machines could only perform a limited range of computations.

On the contrary, discrete devices computing symbolic signs (code) such as the UTM present different characteristics. Conceived as an abstraction, code is not developed on the basis of material properties as in the case of analogue machines. Semiotically, code does not belong to the class of iconic signs but rather to that of symbolic ones; it bears no direct visual or material link to the phenomena it represents or that represent it; the issue of material debasement that characterised analogue devices is here absent (the UTM was in fact a thought experiment). Abstraction and symbolic representation meant that the

intellectual project propelling the development of code as a language (what Paolo Rossi describes as *Clavis Universalis* (1960)) is first and foremost striving for universality. “Omnia per omnia” in Francis Bacon’s words (1962, p. 139).

Code carries no visual pedagogical qualities, the arbitrariness of the symbolic signs chosen removes the possibility of learning by observing. Because of their abstraction, symbolic signs can be combined more freely, according to a specific grammar. The main feature that syntactical manipulation offers is the possibility to express negative conditions. This possibility is not possible for analogue devices because:

“The analog computer cannot represent nothing (no-thing) because it is directly or indirectly related to ‘things’, whereas the ‘language’ of the digital computer is essentially autonomous and arbitrary in relation to ‘things’ (except in so far as all the information requires matter-energy in the form of markers for its transmission). The analog computer is an icon or an image of something ‘real’, whereas the digital computer’s relationship to ‘reality’ is rudimentarily similar to language itself.” (Wilden 1972, pp. 162-163).

This passage by Wilden also helps us clarify an earlier remark made in the introduction in which we claimed that even modern digital computers perform analogue computations. The earlier remark took advantage of the transversality of computation; that is, the ‘portability’ of symbolic signs which can exist independently from “something ‘real’” and, therefore, can be superimposed onto an analogue computational system.

Returning to our initial statement regarding the speculative qualities of discrete computation, we will proceed our argumentation by focusing on two separate and yet

sequential moments of the process of computing: first, the very roots of computation – i.e., the translation of phenomena into signals –, and then the logical operations that allows thought to be articulated. As already mentioned, the basic categorisation of computational devices distinguishes between analogue and discrete computation which, semiotically, can be respectively understood as iconic and symbolic signs. By applying Peirce's semiotics (1998), we can say that discrete computation makes use of symbolic signs rather than iconic or indexical ones as the strings of 0s and 1s do not bear any relation (visual or otherwise) to the phenomena they describe. What binary digits represent is rather the rudimentary conditions for difference to emerge. Difference is the minimum requisite for a system of signs to generate strings of statements. In the case of discrete computation, the economy of the sign system (0/1, two in total) has also important practical implications as it remaps itself neatly onto the possible states of a gate in an electric circuit (open/close, 0/1). The arbitrary nature of symbolic signs, however, always implies a gap between phenomena and their representations, which requires the presence of a connector, a cypher to manage the exchange between meaningless strings of 0s and 1s and intelligible messages. The history of computation has traditionally attributed the role of connector to cryptography. Cryptography, therefore, not only played an important historical role (it has been a classical computational problem since the Middle-Ages (Ellison and Kim 2017), but also, its relevance is still central today as we still confront issues of communication between machines and humans. Intellectually, cryptography is tasked with creating a communication channel between different domains: the alien quality of the cypher text and the intelligible realm from which meaning and design will potentially emerge. This dissertation will charge

cryptography and, more precisely, cyphers with historical, intellectual and practical agency to demonstrate their importance in the history, present, and future of computation in architecture. In modern digital computers, operations of translation are performed on strings of binary signs by algorithms whose operations are also represented by the binary sign system. It follows that the binary sign system used by computers can generate all the necessary properties to perform all its functions necessary to complete tasks. From the point of view of semiotics, cyphers are special signs: they are homogeneous with the set they are part of, but can also perform additional tasks. For now let us make a note of the special feature of cyphers, we will return to it again in the final chapter that will further explore the role and significance of cyphers.

As suggested, the choice of binary numeration is grounded in both practical and economic reasons, but it also presents similarities with other systems of signs such as natural language. To highlight differences and similitudes with natural language will help foreground how binary code conceptualises phenomena and how it possesses generative potentials. Several theoreticians and philosophers (Bühlmann 2020, Evens 2015, Langer 1948, Link 2016, Virno 2013, Wilden 1972) have drawn parallels between discrete symbolic signs and language based on their common differential nature which not only precedes meaning, but also the emergence of any formal element (morphemes, in the case of language, and strings of code, for discrete computation). The connection between computation and language is provided by Saussure when he stated that “language may be content simply to contrast something to nothing” (Saussure 1959, p. 124). Saussure’s

observation that language is only constituted by "negative facts"[3] (Virno 2013, p. 28), also provides a useful analogy for binary code as 0s or 1s do not represent anything other than their difference (for this reason, being discrete and distinguishable are the only fundamental properties they must possess). Similar to language, difference is the necessary pre-condition for binary code to function. More directly we can say that: difference comes before the emergence of actual statements. Binary code is 'negative' both because strings emerge out of an opposition between signs (0 is not 1, and vice versa) as well as because its signs do not represent empirical phenomena or a priori reality that would give it a foundation. David Link's remark (2016, p. 19) that the signals of binary code are bottom-less should be understood along these lines. Similar to the opening remarks of Hegel's *Science of Logic*, the 0s and 1s of binary code represent the minimal semiotic articulation to differentiate something (1) from nothing (0), make it amenable to calculations whilst carrying no semantic value (or not yet). Binary language simply represents pure difference[4] in the most generic form, so that it can be later translated into more specific entities such as numbers or letters. Hegel 's description of his logical system could also be interpreted as an illustration of discrete computation: "With this… indeterminateness and vacuity of conception, it is indifferent whether this abstraction is called space, pure intuiting, or pure thinking". (Hegel, 1990 quoted in Link 2016) We should note in passing that these considerations also show the limitations of what we could call a 'psychological' framing of discrete computation. Recent readings have interpreted binary code as symbolising different kinds of oppositions

[3] In Saussure's own words: "…in language there are only differences *without positive terms*". (Saussure 1959, p. 120).
[4] The only condition for binary code to work is that symbols are distinguishable (Link 2016, p. 18).

such as male/female, black/white projecting onto it values that belong to the interpreter rather than the actual logic that guides the application of binary signs to computational problems.

We can speak of the generative qualities of binary code as clear and identifiable entities that emerge out of a process of differentiation; they result from it rather than preceding it. Moreover, the gap between signs and phenomena should not be seen as a lack, a deficiency limiting the descriptive potential of digital computation; rather, it is precisely this gap that constitutes the conceptual space in which speculation can be articulated. Operations of translation and transformation of symbols are only possible because computational representations of phenomena, or objects, are not the object itself.

Once again, we observe an analogy between language and discrete computation, here represented by a fundamental property of both systems; that is, the ability to negate. Analogue computation rests on the existence of a physical property (length, voltage, etc.) which is replicated in the analogue computing device. Analogue computation necessarily establishes a positive relation between signs and phenomena. An analogue computer cannot say 'not-A' (Wilden 1972, p. 162); however, it can assign a zero value to one of its variables, which marks a conceptual difference between the notion of zero and that of negation. On the contrary, the denotative, arbitrary semiotics of discrete signs shares with natural languages the possibility to negate an object (this is 'not-A') and, therefore, to play with the disjunction between representation (code) and phenomena. As we have seen, this is a fundamental property of symbolic sign systems which are in fact nothing but a system to mark oppositions, or, better, differences from which more structured strings emerge.

The material basis of each type of computation has tangible consequences on the descriptive affordances it engenders. Analogue computation can only perform a limited range of calculations. If we briefly return to perspective machines, we observe devices that only compute by taking advantage of the iconic, 'positive' nature of analogue computation. Analogue computation presupposes a movement toward generalisation which starts with conceiving a device for a specific problem (e.g., the construct of a linear perspective) to then apply it to other computational problems to test its transferability. In analogue computers, computation and materiality are so fundamentally linked that the problem of universality cannot be posed at the outset but only introduced afterwards by extending the application of the analogue computer to a wider set of problems. As a result, the issue of transferability is always present when working with analogue computers. On the contrary, the notion of disjunction between signs and phenomena helps us grasp what is at the core of discrete computation. We can say that discrete computation is inherently speculative, because, whilst being arbitrary, it allows for a wider range of possibilities by exploring what is not and what could be. The possibilities foregrounded by the negative and disjunctive quality of computation also re-emerge at the syntactic level, that is, at the level in which logic provides the instruments for the articulation and the generation of thought through computational means.

Another space for speculative thinking emerges at a computational 'higher' level. We are no longer focusing on the material basis of discrete computation but rather the limits of the logical operators guiding the translation and recombination of symbols. By logic here we mean the broad range of operations that can be performed on binary numeration. Whether

formal logic was complete, consistent, and robust enough to found mathematics and, consequently, computation was the centre of the mathematical debate between the end of the nineteenth century and the 1930's. This debate was eventually concluded by Gödel's (1931) incompleteness theorem which proved the insurmountable limits of deductive thinking. Proving that the relation between truth and proof was not a stable one implied that "…(l)ogical mechanisms were not simply demonstrative of already made truths and facts, but were instead in search of solutions, learning from unplanned outcomes, and transforming universals into experimental axiomatics." (Parisi 2019, p. 41) Beyond strict formalisation and classical logic, new domains of research opened up for computation to operate within dynamic, open, uncertain, fallible forms of rationality. For instance, hypothetical assertions such as those presupposed by abductive methods or fallible, paradoxical thinking as those proposed by New Logic. (Magnani, ed., 2013) Again, these considerations also show the potential for computation to be not only understood functionally, as a means to an end, but also speculatively, as an instrument for searching, an extension of human cognitive abilities rather than its replication.

The theoretical perspective highlighted above can guide us to an alternative understanding of, for instance, digital simulations in architectural or urban design as such simulations never fully replicate the physical phenomena they are tasked to simulate and can only capture simplified versions of them. However, what may appear to be an insurmountable limitation (the digital simulation never exactly coincides with the phenomenon simulated) also represents an opportunity to use digital simulations speculatively, exploring scenarios that may not be, or not yet, possible to actualise. The

relation between possible, plausible, and actualisable is at the core of the design process understood as a search for opportunities and configurations which are not known or evident yet. Such speculative capacities of computational simulations have been the object of investigation of many philosophers of sciences. Recent academic literature on this subject (Varenne 2001) also reiterates how digital simulations can be experimental and mediate theory and practice. Similar intuitions on the role of computation can be traced as far back as Leibniz who already intuited that a mechanical, calculating devices "could be used to test hypothesis". (Goldstine 1972, p. 9)

To understand the speculative aspects of discrete computation we need to develop methodological instruments that reject identity-based, fixed readings in favour of non-chronological, transversal, and speculative ones. Such instruments will need to articulate the relation between computation and reality as well as between the different fields which contribute to the hybrid field of digital architecture. On the one hand, *DABC* seeks to foreground such an approach by discussing specific case studies taken from different historical periods and design disciplines; on the other, the next two chapters on paradigms and cyphers will put forward productive analogies to articulate the computation/reality relation and move from the field of humanistic studies towards architecture.

Chapter 2

**Paradigms**

One of the most effective concepts through which to appreciate the organisation of *DABC* is that of the paradigm. Each of the eight chapters of *DABC* is dedicated to a paradigmatic figure for computational design which reorganises historical precedents providing new material for novel critical interpretations. For instance, chapter eight ("Voxels and Maxels") discusses the computational work of Leonardo and Laura Mosso for the first time as well as provides a novel interpretation of Frederik Kiesler's late projects. Paradigms in *DABC* provide a conceptual framework to bridge between ideas and things, between the realm of the intelligible and that of sensible. It is along these lines that we can understand why Victor Goldschmidt-defined paradigms as "element-form" (Goldschmidt, 1959 quoted in Agamben 2009, p. 21); that is, as single instances (element) able to encapsulate a larger set of ideas or concerns (form). Similarly, the case studies analysed in *DABC* do not distinguish between the physical manifestation of an artefact and the idea it embodied (e.g., perspective machines developed in the fifteenth and sixteenth century are defined as 'built thought'). Within the context of this dissertation, the paradigm proves to be an effective instrument for investigation as it maintains a connection between process and product; a key element for historiography of digital architecture. In what follows I will analyse the figure of the paradigm in greater depth to address the four points raised in the previous chapter.

The notion of paradigm in historiographic studies is fundamentally linked to Thomas Kuhn (1970) who developed it in his famous book published in 1962 titled *The Structure of*

*Scientific Revolutions*. Though its genealogy is clear, in Kuhn's book the definition of what a paradigm is far less definitive: Margaret Masterman (1970) identified 21 different meanings of the word. The vast bibliography which followed the publication of Kuhn's study has exponentially increased the number of interpretations and applications of the concept. Architects have paid relatively little attention to paradigms and even the few exceptions we have, such as Colin Rowe's (1983) essay titled "Program vs Paradigm", rest on a rather personal definition of paradigms which bears little connection to Kuhn's. In Rowe's argument the paradigm is made to coincide with the notion of type. As such the theoretician saw paradigms implicitly promoting a "pessimistic" vision in which "…both present and future are to be no more than a continuation of the past (surely no better)". (Rowe 1983, p. 9) The notions of paradigm and type are not equivalent and cannot be used interchangeably because type is understood as an ideal, transcendental object (at least in its original definition by Quatremère de Quincy (2000)) which can never be fully actualised. It is a notion influenced by the milieu in which it emerged; that is, the philosophical vision of the Enlightenment, conjured up by so-called rationalist thinkers. Kuhn's original argument suggests that paradigms originated out of real scientific developments; that is, they are immanent, mutable figures which are subjected to modification or replacement. In short, paradigms are historical notions emerging out of tangible examples or ideas which are bound to evolve, be superseded, or mixed with other notions. Their historical, immanent character suggests that paradigms are dynamic and sensible entities, a point which cannot easily conform with the rationalist tradition. More recently, Pablo Miranda Carranza (2014) proposed a different prospective by aligning paradigms to computer programs in his

"Programs as Paradigms". Though rich with original considerations and examples of computational design, the essay does not question Rowe's "pessimistic" view of paradigms, but does acknowledge it as an important reference for the introduction of paradigms in the architectural discourse. Similar consideration can be extended to attempt to classify programming languages through paradigms. (Steinfeld Sandoval Olascoaga, 2014)

The key text building on Kuhn to provide a richer and more fruitful definition of paradigm is Giorgio Agamben's (2009) essay titled "What is a Paradigm?" in which paradigms are investigated beyond their role in guiding scientific revolutions. Agamben's essay will guide the examination of paradigms in this dissertation, it will help link the research methods employed in *DABC* and introduce the notion of the cypher, which will be developed in the next chapter. Agamben discusses two main definitions of paradigm provided by Kuhn himself (1970, p. 182). The first aims at identifying a scientific community which adheres to a shared (paradigmatic) set of models, techniques, and practices. The second one, more fitting for this discussion, conceives the paradigm as a single element within a set. What makes such a singular element stand out is that it acts as a common example for all the elements in the set. The paradigm both gives rises to and makes intelligible – at least in their main qualities – all the members of the set and the relationships between them. Kuhn calls a such a set "normal science". Paradigms differ from rules as they are dynamic figures which articulate a particular field without fixing strict rules or identities. Kuhn in fact famously defined them as able to "guide research even in the absence of rules" (1970, p. 42). The dynamism of paradigms can extend forward in time as it can also play "an essential role in preparing the way for perception of novelty" (1970, p. 57).

In *DABC* paradigms are employed in both senses, as instruments to structure both historical case studies, and as intelligible guides to future instances of digital design. The historiography of digital design is an area in which little scholarly research has been developed and the dynamic, incomplete character of paradigms serves well the purpose of charting such a vast and open territory, whilst avoiding more traditional approaches (e.g., chronological classification) which, as already mentioned, proves ineffective. To develop a richer, more apt definition of paradigm, we need to integrate Kuhn's analysis and Giorgio Agamben's reflections, which further liberate paradigms from any strict association with scientific discoveries and highly structured or analytical frames based on rules. Agamben's move builds on Foucault's epistemology, according to which the definition of paradigm (though Foucault seldom used this word) widens to no longer identify "normal science" only, but also to detect the simple existence of epistemological figures in the whole of the discourse of a certain historical period. Foucault's account of the panopticon provides Agamben with an example to articulate how the French philosopher challenged and expanded Kuhn's paradigm. The Panopticon is an actual example[5], a single instance that forms and makes intelligible a whole series of both synchronic and diachronic phenomena which exceed the disciplinary domain it first appeared in (Foucault himself used the Panopticon as a paradigm to reveal the disciplinary character of a whole series of

[5] The panopticon was a model for a particular type of prison which Jeremy Bentham first described in 1791 (Bentham, 1791). Though consisting of just a description and a formal model, it contained sufficient information to be easily built. In fact, in the course of the nineteenth century many prisons were erected according to Bentham's model. It is in this sense that we can understand Bentham's description as an actual instance; Agamben, on the other hand, follows Foucault who insists in describing the panopticon as a "dream building" and "a mechanism of power reduced to its ideal form" (Agamben 2009, p.17). As we have noticed when discussing Rowe's use of paradigms, it is misleading to keep relegating paradigms to a transcendental dimension; what is novel about their definition is the very fact that they are real – be it physically actualised or part of the discourse of a certain discipline.

institutions). Whereas Kuhn thought of paradigms as figures emerging in the present to affect future scientific research, Foucault used them to both reorganise historical material as well as to shed light on contemporary phenomena. In *DABC* this approach is stretched even further as paradigms are conceived in the present and cast back onto historical precedents. This operation provides new lenses to read historical case studies, re-organise them along paradigmatic lines, and let the deeper, persistent motivations behind the use of computational techniques in design emerge.

By building on these considerations, *DABC* is divided into eight chapters each dedicated to a paradigmatic figure for digital design. Each paradigm reveals the emergence of important ideas, techniques, or artifacts in architecture and urbanism, regardless of the historical period in which they manifested themselves. The approach echoes Foucault's as it widens the range of documents (statements, in Foucault's terminology) contributing to the discourse of a certain period. This 'flat' account is essential for the historiography of digital design as it avoids distinctions between theory and practice or between instruments, designs, and buildings which are all taken together as valid domains for epistemological discoveries.[6] The paradigmatic approach operates transversally both in time and in disciplinary terms.

Finally, a history of digital architecture articulated through paradigms is liberated from chronological accounts, thus removing the need to fix an origin for computation. The

[6] In David Hollinger's words: "the Kuhnian vision replaces …[the] materialist-idealist disputation with a dialogue between tradition and contingent experience, in order that the historian can more freely investigate the functions of cultural forms as organizing devices". In some respect, the above passage could be also understood as a reply to Rowe's characterisation of paradigms. (1980, p. 201)

notion of origin can be strictly understood both in time (single event which accounts for the emergence of a movement, technique, etc.) and authorship (a person or group to credit for a particular innovation). As seen in the previous chapter, both notions are at odds with how computation developed, a historiographic issue that *DABC* tried to address. Each of the eight chapters is not a search for fixed, definitive definitions of specific concepts/instruments. Rather, they are exercises in tracking the evolution of a certain set of techniques as they move through different cultures, applications, and disciplines without singling out any particular instance as solely responsible for their introduction into design disciplines. As we will shortly see when discussing the logical status of the paradigm, mapping a paradigmatic example onto a certain set of instances never leads to a complete definition. A paradigmatic history of digital design serves well contemporary readers who will recognise in the way they daily design with computers, some resemblance to different sets of experiments or ideas, developed in the past. This realisation will make instances in the history of digital architecture resonate with the present, thus making contemporary digital designers aware of the potential relevance and originality of their work.

The discussion that follows below explores the logical status of the paradigm. Logic occupies a central role in computation as it is one of the key disciplines that contributed to the development of symbolic code, a key element to program digital computers. Agamben defines the logical status of the paradigm by comparing it to the two most common models for inferential thinking: deduction and induction. If deduction generates knowledge by moving from the universal to the particular (universal rules give rise to particular instances), induction follows the opposite path. The former moves from wholes to parts, whereas the

latter infers the whole from its parts. Whereas both models indicate a general, strict, linear trajectory for knowledge to emerge, the paradigm contraposes a more open, less hierarchical model. Agamben quotes Aristotle to point out that the paradigm functions as a "part with respect to the part, if both are under the same but one is better known than the other". (Aristotle quoted in Agamben 2019, p. 19) To the strict structure of formal logic, the paradigm opposes the figure of the analogy. Consistent with Kuhn's second definition, the paradigm is a particular member of a set that can be utilised to map a whole domain of instances. In other words, the paradigm performs its operation from within as "it is…impossible to clearly separate an example's paradigmatic character – its standing for all cases – from the fact that it is one case among others". (Agamben 2009, p. 20) This process is different from the one presupposed by deduction and induction, both of which extrapolate knowledge that will become independent from the set itself.

The implications raised by the logical status of the paradigm are of great interest for computational design. On the one hand, the deductive model computationally aligns with a rule-based approach in which the (pre)determination of a set of rules allows for the generation of all the members of a set. This is the approach utilised, for instance, in Chomsky's linguistic, or shape grammar. (Stiny Gips 1972) It is a strict approach that does not suit historical studies on computation which are characterised by opportunistic encounters and offer little insights if analysed deductively. If we were to apply a rule-based approach to the history of digital architecture, we should start by determining a number of fixed invariants that, when combined, would give rise to the totality of instances of an hypothetical set called 'digital architecture'. To pursue such an approach would immediately

present insurmountable obstacles: how many and what kind of invariants should we fix? How to define such primitives? What would be the legitimate operations we could employ to combine invariants? It is evident that such methods would only account for a very small, if not negligible, set of instances and would overly rely on syntactical manipulation making the detection of influences coming from different disciplines (transversality of the digital) very difficult to discern. For these reasons, historiographical classification through deduction could not be adopted in *DABC*.

The so-called Big Data approach broadly aligns with the inductive method in which general knowledge is extracted from a variety of examples (the formula Big Data indicates that large amounts of instances are necessary for this approach to be effective). The approach is also at the core of AI methods such as Neural Networks (NN) which architects are increasingly employing in their research and practice. However, if we analyse more carefully some of the specific procedures utilised by NN, we will realise that such algorithmic procedures do not fully align with an inductive view of knowledge. First of all, NNs operate statistically, by approximating a result which can never be posed as a firm generalisation (or truth); even if we were to speak of inductive knowledge, this would only be understood in 'weak', statistical terms. Secondly, any method to survey the complex, multi-dimensional space of a NN (e.g., through dimensionality reduction algorithms[7]) always returns a partial

[7] The reduction of the number of dimensions in a dataset is one of the central challenges posed by working with Big Data. Digital designers working with large datasets are bound to encounter this issue at some point in their work. In data science dimensions do not have a spatial meaning, rather, they identify the number of features describing each data point. For instance, though never fully confirmed, it is rumoured that each Facebook user is described by 52,000 types of data (Green 2018). A hypothetical spreadsheet of all Facebook users would therefore consist of 1.69 billion rows (approximate number of Facebook users at the time of writing) and 52,000 columns or dimensions. Due to the impossibility for human cognition to navigate such a vast data space, computer scientists developed dimensionality reduction algorithms whose function is to

account of such a space. Again, the kind of broad generalisations underpinning inductive models would not be possible as algorithms like these only connect particulars to particulars. The figure of the paradigm provides a better analogy to grasp such processes. Taking the example of dimensionality reduction algorithms, the relation between the algorithm and the dataset is one in which a part (the algorithm) connects to a part of the set (data in multi-dimensional space). No algorithm can capture the identity of the space of multi-dimensional data points, which remains indiscernible in its totality. Analogically, each of the eight paradigms animating the chapters of *DABC* is cast onto a complex, varied, diachronic set of precedents to return only a salient aspect from them, rather than unifying them under a single set of laws. Such an approach is not only open to modifications and additions, but it also allows influences between different disciplines to be traced, thus providing a more complex and accurate history of digital design.

These considerations confirm the potential for paradigms to be helpful instruments to not only organise past examples, but also guide future design research. As we have seen, the issue remains one of translation, of putting into communication diverse domains in order for knowledge to emerge through interaction and speculation. The field of cryptography has already been singled out as central in the history of computation and will

diminish the size of the dataset (both in terms of number of features and, potentially, data points) whilst minimising the loss of information contained in the original dataset. There are many different algorithmic procedures available to reduce the number of dimensions of a dataset, however all the literature on the subject concedes that the reduced dataset will always only be a partial representation of the original (though the reduced dataset will capture the essential qualities of the original one). Dimensionality reduction is one of the most common and debated procedures in data science; as such, it is an issue that anybody working with Bid Data eventually confronts in their work. For these reasons, dimensionality reduction algorithms provide a good testing ground to analyse and prove the notion of the cypher. See both 'Dimensionality reduction'. *Wikipedia*. Available at: https://en.wikipedia.org/wiki/Dimensionality_reduction (Accessed 6 December 2021) and 'Feature (machine learning)'. *Wikipedia*. Available at: https://en.wikipedia.org/wiki/Feature_(machine_learning) (accessed 1 January 2022).

be further analysed in the next chapter in which the notion of the cypher will be understood as a computational evolution of the paradigm as it closely shares both definition and functioning. Cyphers, however, will allow us to move within the domain of computation as concepts and mechanisms can be numerically expressed.

Chapter 3

## Cyphers

In order to deepen the discussion on issues of translation, this chapter introduces the notion of cypher as a conceptual figure moving the notion of the paradigm closer to computation. In fact, if paradigms are methodological figures emerging from the field of the humanities, cyphers are mathematical instruments which are an integral component of computation. If the former can be utilised through analogy, the latter are employed directly without mediation. The task of this chapter is to expand this initial discussion on cryptography and position it as a central topic in computation, articulating the relation between knowledge and computational instruments. The conceptual implications of deepening the role of cryptography are profound and impact both historical studies and contemporary computational design methods (particularly the use of so-called intelligent algorithms). The aim is to show that the methodological framework outlined in this dissertation has merits that go beyond the application to historical analysis and can also be effective to discuss contemporary issues. This chapter will unravel the notion of the cypher along different vectors having to do with the historical, semiotic, and architectural connotations of this concept.

What is a cypher? Most commonly, cyphers are understood as rule-based mechanisms to hide (encrypt) or reveal (decrypt) messages.[8] Etymology, however, takes us much deeper

[8] "An algorithm for encryption or, in its inverse form, for decryption". (Daintith and Wright, 2008).

into mathematics as the word ‘cypher’ derives from the French *cifre,* denoting “the arithmetical sign for zero”, which in turn descends from the Arabic *sifr* and the Hindu *sunya* “zero, empty, nothing”.

The mathematical sign ‘zero’ enters western culture in the renaissance, providing a new conceptual instrument that will impact different cultural domains. Brian Rotman (2014) effectively maps its penetration by tracing the impact of the ‘zero’ mathematical sign on renaissance mathematics, painting, and finance. What Rotman shows is how the concept implicit in the sign ‘zero’ operates on a variety of levels: first in its ‘purest’ form through the application to mathematics, and then to other domains such as art and commerce. And so it seems that cyphers, in the sense of the sign identifying the empty set, the zero, like paradigms do, operate transversally, across disciplines restructuring them according to new parameters. The comparison between cyphers and paradigms rests on their common definition. A cypher is a particular element in a set (a code made up of letters or numbers) that re-organises all items in that set. This definition matches Kuhn’s second description of the paradigm: cyphers both belong to and organise the set of signs they give rise to. Semiotically, cyphers are thus *meta-signs* with a double function: they give rise to a set that, simultaneously, they make intelligible.

The meta-semiotic nature of cyphers provides an instrument for both negotiating issues of translation and circulation of knowledge. More precisely, by being applied to random, ‘noisy’ strings (cypher text), cyphers generate intelligible messages (plain text). Cyphers are designed objects; by playfully interacting with the cypher text, they give rise to a new set whose elements share common characteristics determined by the rules set by the cypher

itself (algorithm). They pertain to the non-mimetic tradition as they represent phenomena through notation (symbols) rather than by replicating or emulating their sensible appearance. One pre-modern manifestation of a notational system of this kind is found in Ramon Llull's metaphysics, which availed of proto-computational machines, in the form of concentric wheels. These machines, according to Llull, could decipher the hidden order of the cosmos. Conceived between the end of the thirteenth and early part of the fourteenth century and described in his *Ars Magna*, Llull's concentric disks were engraved with the first nine letters of the alphabet (from b to k) and, once set in motion, would return alphabetic codes that could be decrypted by using Llull's own tables. (Bottazzi 2018, pp. 17-22) All combinations thus decoded would reveal aspects of a perfectly ordered cosmos and implicitly demonstrate the superiority of its creator: the Christian god. In order not to contradict Llull's own metaphysical presuppositions, many combinations did not feature in Llull's tables, massively limiting the power of the combinatorial logic adopted (as Leibniz also noted). Some of the potential of operating through cyphers was however evident in Llull's project: the wheels were devices to mobilise knowledge, vehicles for exploration; in short, they acted "… as a way to *work* and to actually investigate and 'compute' the world" (DuPont, 2018, p. 101). Not long after Llull's experiments, the *De componendis cifris* (1466) by Leon Battista Alberti proposed a much more sophisticated cryptographic method based on polyalphabetic cyphers (which also employed concentric wheels). A polyalphabetic cryptographic method is much more difficult to break as it relies on multiple cyphers that change numerous times throughout the encoding/decoding process. What is relevant to foreground here, are the mathematical tools necessary to decrypt polyalphabetic cyphers.

Every time the cypher changes, the size of the combinatorial space of all possible combinations will exponentially increase, giving rise to what mathematicians call a combinatorial explosion.[9] Mathematics becomes the only instrument to survey such a space as its enormous extent vastly exceeds human cognitive abilities.[10] In the renaissance, cryptographic methods found applications well beyond protecting diplomatic secrets and philosophers deployed them in attempts to reveal the very secrets of nature as, for instance, in the case of Francis Bacon. Through operations of translation, substitution, and rewriting, cyphers generate intelligible sets of elements out of random ones (and vice versa). Such operations rely on syntactical rigour rather than mimetic (visual or otherwise) similarity between cypher and plain text.

Cyphers can perform such operations in virtue of the introduction of the number zero in mathematics. As meta-signs, zero signs not only act as ordering devices, but also allow mathematics to abandon the Greek tradition which considered numbers as entities for counting real objects, denying them an autonomous existence. (Rotman 1994, pp. 7-14) The zero sign opens up a playful space for speculation (through the non-coincidence between sign and phenomenon) that can be creatively exploited (in Rotman's account the diffusion of mathematical prospective and paper money were enabled by the introduction of the zero sign).

[9] "The exponential growth rate experienced in many search problems". (Daintith and Wright, 2008)
[10] Polyalphabetic cryptography enjoyed great success and was considered the most advanced method for encryption between the fifteenth and the nineteenth century. It is worth mentioning in passing, that the invention of a formal method to decrypt polyalphabetic cyphers is credited to Charles Babbage, a key figure in the history of computation.

Cyphers acquire speculative power; rather than seeing them as functional devices (to decode diplomatic secrets), they can be used as instruments to search an unstructured domain, in order to provide intelligibility, patterns, or even order. Cyphers allow speculative thought-experiments to be performed on sets of signs by mobilising them, combining them with other instances, or, in other words, by putting them to work. The labour of historians of digital architecture and designers consists in developing the process of exploration by providing different frames of reference to contextualise and instrumentalise the production of new signs by cyphers.

Cyphers' ability to articulate strings of signals derives from their semiotic status of 'empty' signs; that is, the value of cyphers does not reside in their semantics but rather in their eminent ability to structure other signs. As meta-signs, cyphers enable the production of new signs emerging from newly established relationships. Such production is the result of a constructed act which operates both internally to the set (cypher as an element of a set) and externally by restructuring any domain it is applied to. (*ibid*., p. 28-32). Contrary to the Platonic tradition, numbers are no longer entities to count real objects only, through the introduction of the zero sign, numbers can also be used independently from real objects, that is, they become instrument for speculative reasoning. Perhaps, this capacity is best appreciated in algebra in which letters stand in for entities which are unknown. Semiotically, the introduction of cyphers moves the production of signs from an iconic status (counting real objects, what Rotman calls "proto-numbers" (*ibid*., p. 13)) to a symbolic one, as an instrument for reasoning and speculation; a shift which is closely reminiscent of the properties of discrete computation illustrated above in chapter one.

This way of looking at cyphers offers several characteristics which resonate with the notion of paradigms as discussed in the previous chapter. Like paradigms, cyphers operate relationally forging relationships between previously disparate signs. Their similarity to paradigms also extends to logic: cyphers also establish particular-to-particular connections between sets of examples, different from the hierarchical models of induction and deduction. Let's take again the example of the dimensionality reduction algorithms which was already discussed in the previous chapter and re-read it through the figure of the cypher. In fact, dimensionality reduction algorithms are 'made' of the same material as the set of instances they are applied to (binary numerals, in this case) which they structure by establishing new relations. The characteristics (rules) of the cypher determine how it will survey the domain of instances it will be applied to. It follows that the image returned can only be an incomplete representation of the set surveyed. In other words, we have a *particular* instance in our set (dimensionality reduction algorithm) constructing a precise and yet *particular* image of the entire set (database). In architectural representation, a similar instance occurs when a three-dimensional object is represented through orthographic drawings. For instance, a section through a building can only return a partial image of the object it investigated, and yet how a section is drawn follows precise and rigorous rules that determines what and how the building will be captured in the section. But the section is a sign-object, not a building, it elicits further manipulations by either applying different sets of criteria (e.g., by concentrating on the structural, programmatic, material qualities of the building) or by changing the very parameters that generated it (changing the position of the section plane or the conventions applied). It is in this sense

that we can speak of cyphers making data intelligible, that is, foregrounding aspects of a domain that is otherwise cognitively impossible to grasp and facilitating the production of signs that will be amenable to further manipulation. Cyphers reveal common connections between instances without prescribing their semantic orientation. Just like paradigms, cyphers can be understood as diachronic figures that allow the mapping of an uncharted domain (such as that of the history of computational design in *DABC*) without forcing it into static categories or linear accounts.

In conclusion, the computational and semantic analysis of cyphers allowed us to draw analogies with the notion of paradigm and place it within the domain of computation. Such analogies are supported by the common definition that both paradigms and cyphers share. But they are also supported because both paradigm and cypher provide a conceptual frame to investigate the history of digital architecture that operates in a non-chronological, structured, transversal, and speculative fashion. The effectiveness of the approach outlined here should not be solely confined to debates on historiography, but can also effectively guide the investigation on current issues in computational design.

**Conclusions**

In approaching Kuhn's seminal work on paradigms, we encountered two definitions of the word. However, we only made use of one of them, the more 'technical' one, that defined the conditions under which a special set called 'normal science' emerges. It is perhaps time to briefly address the other, more 'social' definition which charged paradigms with the potential to form scientific communities adhering to a shared set of models. The collection of paradigms listed in *DABC* aims at fostering an approach to digital architecture which not only acknowledges the complex trajectories, processes, histories, techniques, and concepts that have informed computation and its impact on spatial disciplines, but also makes digital architects more aware of the deeper ideas and motivations behind the emergence of computers.

Increasing such awareness is not only beneficial for digital architects as they will have a better grasp of the tools they use, but it will also change how digital technologies are implemented in the design process by foregrounding their conceptual rather than utilitarian or techno-fetish value. The successive figures of the paradigm and the cypher provide a conceptual framework to track the evolution of computational instruments in design, rejecting identarian, foundational categorisations which struggle to account for the historical evolution of computation and the different ideas it conflates.

The critical studies on contemporary digital architecture are still largely oblivious to the impact that the production of data will have on historiographical methods. If

historiography has developed many successful models to make up for the scarcity of historical documentation to conjure a coherent picture of the past, the present time offers a reversal of such conditions as every act is first and foremost recorded as data to be eventually discarded or analysed. Though this issue was not central to this dissertation, cyphers' and paradigms' capacity to operate in the absence of general rules to survey open territories could be a useful property to navigate through an over-abundance of evidence in order to reassess historiographical models for contemporary digital architecture.

Paradigms and cyphers have been chosen not only because they address the four critical points related to the historiography of computation illustrated in chapter two (non-chronological, non-fragmentary, transversal, and speculative); but they are also methodologically analogous to the very subject they are deployed to map: digital architecture. Process and product are no longer kept apart as two independent moments, rather they combine to affect one another. This approach offers a much richer framework for critiquing digital creative production by avoiding falling back onto previous models and thereby granting computation conceptual depth.

**The Published work**

Bottazzi, R. (2018). *Digital Architecture Beyond Computers: Fragments of a Cultural History of Computational Design*. London: Bloomsbury Visuals.

Bottazzi, R. (2019). “Crypto Architecture: Notes on Machine Learning and Design”. In *Ghost of Transparency: Shadow Cast and Shadow Cast Out*. Edited by M. R. Doyle, S. Savić, V. Bülhmann. Berlin, Boston, MA: De Gruyter, pp. 21-30.

**Bibliography**


Agamben, G (2009). “What’s a Paradigm?”. In *The Signature of All Things: On Method*. Translated by L. D’Isanto with K. Attell. New York: Zone Books, pp.9-32.

Bacon, F. (1962) *The Advancement of Learning.* Edited with an Introduction by G. W. Kitchin. London: Dent. 1st editions published in 1605.

Bentham, J. (1791). *Panopticon: or, the inspection-house. Containing the idea of a new principle of construction applicable to any sort of establishment, in which persons of any Description are to be kept under Inspection. and in particular to penitentiary-houses, prisons, Houses of Industry, Work-Houses, Poor-Houses, Manufactories, Mad-Houses, Hospitals, and Schools. With a plan of Management adapted to the Principle*. In a series of letters, written in the Year 1787, From Crecheff in White Russia, to a Friend in England. By Jeremy Bentham, of Lincoln's Inn, Esq. Dublin: Thomas Byrne.

Borzacchini, L. (2008). *Il Computer di Platone*. Bari: Edizioni Dedalo.

Bottazzi, R. (2018). *Digital Architecture Beyond Computers: Fragments of a Cultural History of Computational Design*. London: Bloomsbury Visuals.

---- (2019). “Crypto Architecture: Notes on Machine Learning and Design”. In *Ghost of Transparency: Shadow Cast and Shadow Cast Out*. Edited by M. R. Doyle, S. Savić, V. Bülhmann. Berlin, Boston, MA: De Gruyter, pp. 21-30.

Bühlmann, V. (2020). *Mathematics and Information in the Philosophy of Michel Serres*. Bloomsbury Academic.

Cardoso Llach, D. (2015). *Builders of the Vision: Software and the Imagination of Design*. New York and London: Routledge.

Carpo, M. (2001). *Architecture in the Age of Printing: Orality, Writing, Typography, and Printed Images in the History of Architectural Theory. Cambridge*. Mass: MIT Press.

Carranza, P. M. (2014). “Programs as Paradigms”. In *Architectural Design*, “Emphathic Space: The Computation of Human-Centric Architecture”, Edited by C. Derix and A. Izaki, Vol. 84, Issue 5. Chichester, West Sussex: Wiley, pp.66-73.

Church, A. (1936). "An unsolvable problem of elementary number theory". *American Journal of Mathematics*. 58 (2): 345–363.

Cohn, G. (2018). AI Art at Christie’s Sells for $432,500”. In *The New York Times*, 25 October 2018. Available at: https://www.christies.com/features/A-collaboration-between-two-artists-one-human-one-a-machine-9332-1.aspx (accessed on 5 January 2021).

Daintith, J. and Wright, E., eds. (2008). Combinatorial explosion. In: *A Dictionary of Computing*, 6th ed. [online] Oxford: Oxford University Press. Available at: https://www-oxfordreference-com.libproxy.ucl.ac.uk/view/10.1093/acref/9780199234004.001.0001/acref-9780199234004-e-857?rskey=LRpMhl&result=959 (accessed 1 December 2021).

---- cipher. In: *A Dictionary of Computing*, 6th ed. [online] Oxford: Oxford University Press. Available at: <https://www-oxfordreference.com.libproxy.ucl.ac.uk/view/10.1093/acref/9780199234004.001.0001/acref-9780199234004-e-746?rskey=bZvgsX&result=839> [Accessed 1 December 2021].

DuPont, Q. (2018). “The Printing Press and Cryptography: Alberti and the Dawn of the Notational Epoch”. In *A Material History of Medieval and Early Modern Cyphers*, edited by E. Ellisonw and S. Kim. New York, NY: Routledge, pp. 95-117).

Ellison K.E., Kim S.K., eds. (2017). *A Material History of Medieval and Early Modern Ciphers*. New York, London: Routledge.

Evens, A. (2015). *Logic of the Digital.* London: Bloomsbury Academic.

Floridi, L. (2020). *Le tre funzioni del linguaggio digitale e le loro conseguenze*. Available from: https://thephilosophyofinformation.blogspot.com/2020/09/le-tre-funzioni-del-linguaggio-digitale.html?spref=tw (accessed on 27 October 2021).

Galloway, A. (2021). *Uncomputable: Play and Politics in the Long Digital*. London: Verso.

Gödel, K. (1931). "Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I", *Monatshefte für Mathematik und Physik*, v. 38 n. 1, pp. 173–198.

Goldstine, H. H. (1972). *The Computer – From Pascal to von Neumann*. Princeton, NJ and Chichester: Princeton University Press.

Goldschmidt, V. (1959). *Le Paradigme dans la dialectique platonicienne*. Paris : Vrin.

Goodhouse, A., ed. (2017). *When is the Digital in Architecture?*. Montreal, Berlin: Canadian Center for Architecture, Stenberg Press.

Goodman, N. (1976). *Languages of Art: An Approach to a Theory of Symbols*. 1st edition 1969. Indianapolis; Cambridge: Hackett.

Green, A. (2018). “Facebook’s 52,000 data points on each person reveal something shocking about its future”. Available at: https://www.komando.com/social-media/facebooks-52000-data-points-on-each-person-reveal-something-shocking-about-its-future/489188/ (Accessed on 1 January 2022).

Halpern, O. (2014). *Beautiful Data: A History of Vision and Reason Since 1945*. Durham, NC: Duke University Press.

Haugeland, J. (1981). “Analog and Analog“. In *Philosophical Topics*, Spring, Vol.12, No.1, Functionalism and the Philosophy of Mind, pp.213-225.

Hegel, F. W. (1990). *Science of Logic [1812-31]*. Translated by A. V. Miller. London, NJ: Allen & Unwin.

Hovestadt, L. (2020). “Artificial Intelligence”. In *Architecture and Naturing Affairs*, edited by M. An and L. Hoverstadt. Basel: Birkhäuser, pp. 226-231.

Hollinger, D. A. (1980). “ T. S. Kuhn’s Theory of Science and Its Implications for History”. In Gutting, G., ed., *Paradigms & Revolutions: Applications and Appraisals of Thomas Kuhn’s Philosophy of Science*. Notre Dame, IN: Notre Dame University Press, pp. 195- 222, p.201)

Katz, M. (2008). “Analog and Digital Representation”. In *Minds & Machines*, Vol. 18. Berlin: Springer Science+Business Media, pp. 403-408.

Kuhn, T. (1970). *The Structure of Scientific Revolutions*. Chicago, Ill.: University of Chicago Press. 1st ed. 1962.

Langer, S. (1957). *Philosophy in a New Key*. Cambridge, Mass.: Harvard University Press, 1st ed. 1948. !

Link, D. (2016). *Archaeology of Algorithmic Artifacts*. Minnesota: Univocal Publishing.

Lynn, G. ed. (2013). *Archaeology of the Digital*. Cambridge, Mass.; London: MIT Press.

Magnani, L. ed. (2013). *Introduzione alla New Logic: Logica, filosofia, cognizione*. Genoa: Il Melangolo.

Maley, C. J. (2011). "Analog and digital, continuous and discrete". In *Philosophical Studies: An International Journal for Philosophy in the Analytic Tradition*, August, Vol. 155, No.1, pp. 117-131.

Masterman, M. (1970). "The Nature of a Paradigm". In *Criticism and Growth of Knowledge*, edited by I. Lakatos and L. Musgrave. Cambridge: Cambridge University Press, pp.59-89.

Oxman, R., Oxman, R., eds. (2014). *Theories of the Digital in Architecture*. London: Routledge.

Papayannopoulos, P. (2020). "Computing and modelling: Analog vs. Analogue". In *Studies in History and Philosophy of Science*, Vol.83 , pp. 103-120. https://doi.org/10.1016/j.shpsa.2020.05.001.

Parisi, L. (2019). "The alien subject of AI". In *Subjectivity*, 12, pp. 27-48.

Peirce, C. S. (1998). " What's a Sign?". In *The Essential Peirce: Selected Philosophical Writings*, Volume 2 (1893-1913). Edited by the Peirce Edition Project. 1st ed. 1894. Bloomington, ID: Indiana University Press, pp. 4-10.

Quatremère de Quincy, A-C. (2000). *Dictionnaire Historique d'Architecture*. Translated by Samir Younés. 1st edition 1825. London: Papadakis Publisher.

Rowe, C. (1983). "Program vs. Paradigm: Otherwise Casual Notes on the Pragmatic, the Typical, and the Possible". In *The Cornell Journal of Architecture*, no.2, pp.8-19.

Rossi, P. (1960). *Clavis Universalis: Arti Mnemoniche e Logica Combinatoria da Lullo a Leibniz*. Milan: R. Ricciardi editore.

Rotman, B. (1994). *Signifying Nothing; The Semiotics of Zero*. Stanford, CA: Stanford University Press.

Rowland, T. and Weisstein, E. W. (n.d.) "Principle of Computational Equivalence." From *MathWorld*--A Wolfram Web Resource. Available at: https://mathworld.wolfram.com/PrincipleofComputationalEquivalence.html (accessed on 2 January 2022).

Salanskis, J.-M. (2011). *Le Monde du Computationnel*. Editions Les Belles Letters.

Saussure, F. (1959). *Course of General Linguistics*. Edited by C. Bally and A. Sechehaye, with the collaboration of A. Riedlinger; translated and annotated by W. Basking. 1st ed. 1916. London: Duckworth.

Schonbein, W. (2013). "Varieties of Analog and Digital Representation". In *Minds & Machines*, Vol.24. Berlin: Springer Science+Business Media, pp. 415-438. Lewis, D. (1971). "Analog and Digital". In *Noûs*, Vol.5, No.3, pp. 321-327.

Steinfeld, K., Sandoval Olascoaga, C. E. (2014). "IMPERATIVE/FUNCTIONAL/OBJECT-ORIENTED". escholarship, University of California. Available at: https://escholarship.org/uc/item/4b7146r2#main (Accessed on December 8 2021).

Stiny, G., Gips, J. (1972). "Shape grammars and the generative specification of painting and sculpture". In *Information Processing* 71. North-Holland Publishing Company, pp. 1460–1465.

Turing, M.A. 1936–1937. "On Computable Numbers, with an Application to the Entscheidungs Problem". *Proceedings of the London Mathematical Society* (2) 42: 230–265.

Vardouli, T., Touloumi, O., eds. (2019). *Computer Architectures: Constructing the Common Ground*. New York: Routledge.

Varenne, F. (2001). “What does a computer simulation prove?”. Published in *Simulation in Industry, Proc. Of the 13th European Simulation Symposium*, Marseille, France, October 18-20th, 2001, ed. by N. Giambiasi and C. Frydamn, SCS Europe Bvba, Ghent, 2001, pp. 549-554.

Virno, P. (2013). *Saggio Sulla Negazione*. Turin: Bollati Boringhieri.

Wilden, A. (1972). *System and Structure*. London: Tavistock Publications.

Wolfram, S. (2002). *New Kind of Science*. London: Wolfram Media.

Zeynep Celik, A., May, J., eds. (2019). *Design Technics: Archaeologies of Architectural Practice*. Minneapolis: University of Minnesota Press.