Dimensions of Architectural Knowledge, 2021-03
https://doi.org/10.14361/dak-2023-0512

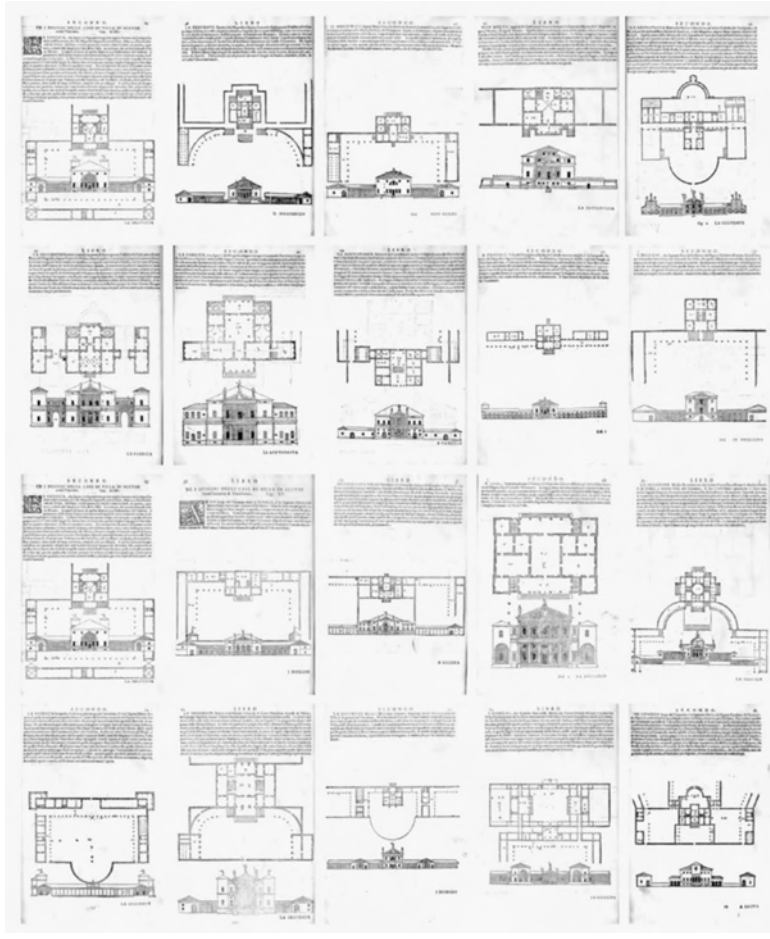
Palladio, the Computer Program. Source Code and Architectural Principles

Pablo Miranda Carranza

Abstract: In 1992 MIT Press published P*ossible Palladian Villas (Plus a Few Instructively Impossible Ones)*, the result of a chance collaboration between Richard Freedman, an undergraduate majoring in Computer Science, and George Hersey, Professor of Art History at Yale. The book described their findings while writing the software called Palladio, which was included in a floppy disc accompanying the book. Software and book presented a new type of architectural historiography, one that used writing and running programs to analyze and explain a body of architectural work. The third partner in this collaboration was the computer. In its capacity to impersonate Palladio and mimic their intellectual toils the computer embodied many of the myths, and even some of the anxieties, of 20th-century architecture. This paper looks at how Palladio, the software, incorporated and exposed some of the contradictions of a project of disciplinary autonomy partly resting on analyses of Palladian villas by Rudolf Wittkower and Colin Rowe. It proposes a close reading of its code, written in the C programming language, to understand how the principles of this disciplinarity, presented as so intrinsically human, could be translated into the mechanical operations of a computer program.

Keywords: Architectural Principles; Formalism; Disciplinarity; Interactivity; Debugging; Conversation; Code; Heuristics.

Corresponding authors: Pablo Miranda Carranza (Department of Architecture and the Built Environment, LTH, Lund University, Sweden); pablo.miranda@abm.lth.se; http://orcid.org/0000-0001-9963-5759.
Open Access. © 2023 Pablo Miranda Carranza published by transcript Verlag. This work is licensed under the Creative Commons Attribution 4.0 (BY) license.



1.
Andrea Palladio: Pages 47 – 65 from Il Secondo Libro dell'Architettura.

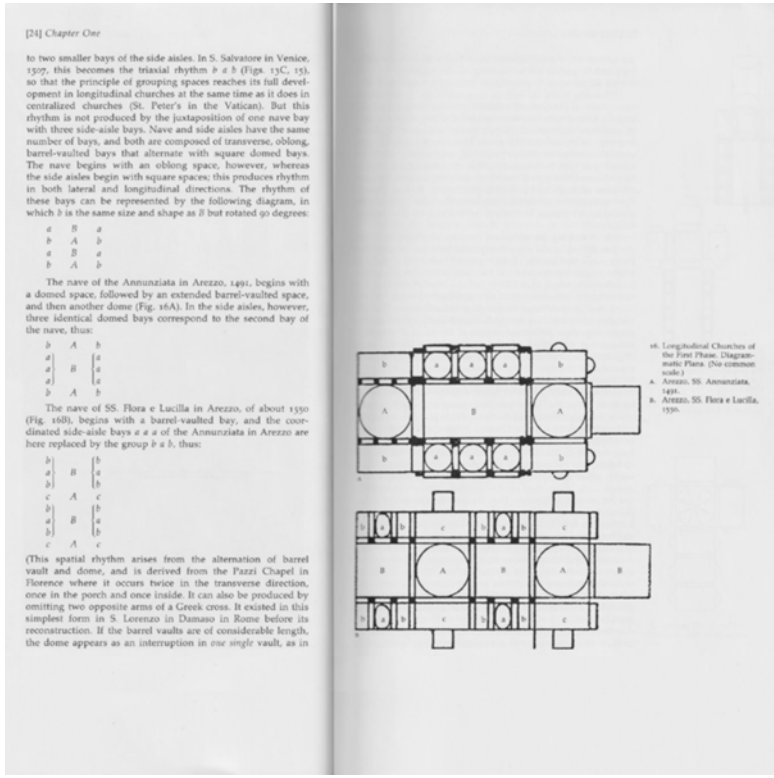
Palladio, the Author of *The Four Books*

»He is inside my Mac!«[1]

Since its publication five centuries ago, architects have been fascinated, time and time again, by the work of Andrea Palladio. Besides the neo-Palladianism of Lord Burlington and Thomas Jefferson, or Palladio's influence on Jean-Nicolas-Louis Durand, analyses of Palladio's architecture became central to articulating the idea of a disciplinary autonomy during the mid-20th century. Palladio's productivity stands as one of the reasons for this interest. From 1531 to 1580, his output included 143 buildings and architectural works, a productivity hard to match by most architects then and now (Puppi 1975). As with Frank Lloyd Wright almost four centuries later, Palladio's sizable output was the result of a building bubble, one that demanded a new form of exurban dwelling in response to the agricultural reorientation of the Venetian Republic's economy. Palladio reinterpreted the villa's original Roman model and delivered it to the administrators of the new type of agricultural estate that emerged in the Veneto during the sixteenth century (Ackerman 1966: 50).

This profusion of work was paralleled by Palladio's own efforts to present his architecture as something beyond its factuality as buildings. As the representative of the new cadre of humanist architects, Palladio bolstered his authorial credentials by including detailed descriptions of his own works in his famous architectural treatise *The Four Books of Architecture*, of which his *Second Book* was dedicated to his villas and palazzi. The purpose of this inclusion was to present his »inventions«, the innovations in the layout and design of domestic architecture for a clientele of affluent landowners and financiers around Vicenza, alongside the classical models he documented as their sources. According to Kurt Forster, *The Second Book* constituted the first *oeuvre complète* in architecture, identifying the person of an architect as coincident with their work (Eisenman 2000). Palladio's technique of representation coordinated plans, elevations, and sections to present idealized versions of his own work, linking them to classical examples from Antiquity by drawing them the same way. This systematization through graphical conventions set the stage for its future interpretations. It implied, particularly to modernist

1 Richard Freedman, author of *Possible Palladian Villas*, referring to Andrea Palladio in conversation with the Author.



[24] *Chapter One*

to two smaller bays of the side aisles. In S. Salvatore in Venice, 1507, this becomes the triaxial rhythm *b a b* (Figs. 15C, 15), so that the principle of grouping spaces reaches its full development in longitudinal churches at the same time as it does in centralized churches (St. Peter's in the Vatican). But this rhythm is not produced by the juxtaposition of one nave bay with three side-aisle bays. Nave and side aisles have the same number of bays, and both are composed of transverse, oblong, barrel-vaulted bays that alternate with square domed bays. The nave begins with an oblong space, however, whereas the side aisles begin with square spaces; this produces rhythm in both lateral and longitudinal directions. The rhythm of these bays can be represented by the following diagram, in which *b* is the same size and shape as *B* but rotated 90 degrees:

a B a
b A b
a B a
b A b

The nave of the Annunziata in Arezzo, 1491, begins with a domed space, followed by an extended barrel-vaulted space, and then another dome (Fig. 16A). In the side aisles, however, three identical domed bays correspond to the second bay of the nave, thus:

b A b
a | *B* | *a*
a | | *a*
a | | *a*
b A b

The nave of SS. Flora e Lucilla in Arezzo, of about 1550 (Fig. 16B), begins with a barrel-vaulted bay, and the coordinated side-aisle bays *a a a* of the Annunziata in Arezzo are here replaced by the group *b a b*, thus:

b | | *b*
a | *B* | *a*
b | | *b*
c A c
b | | *b*
a | *B* | *a*
b | | *b*
c A c

(This spatial rhythm arises from the alternation of barrel vault and dome, and is derived from the Pazzi Chapel in Florence where it occurs twice in the transverse direction, once in the porch and once inside. It can also be produced by omitting two opposite arms of a Greek cross. It existed in this simplest form in S. Lorenzo in Damaso in Rome before its reconstruction. If the barrel vaults are of considerable length, the dome appears as an interruption in *one single* vault, as in

16. Longitudinal Churches of the First Phase. Diagrammatic Plans (No common scale.)
a. Arezzo, SS. Annunziata, 1491.
b. Arezzo, SS. Flora e Lucilla, 1550.

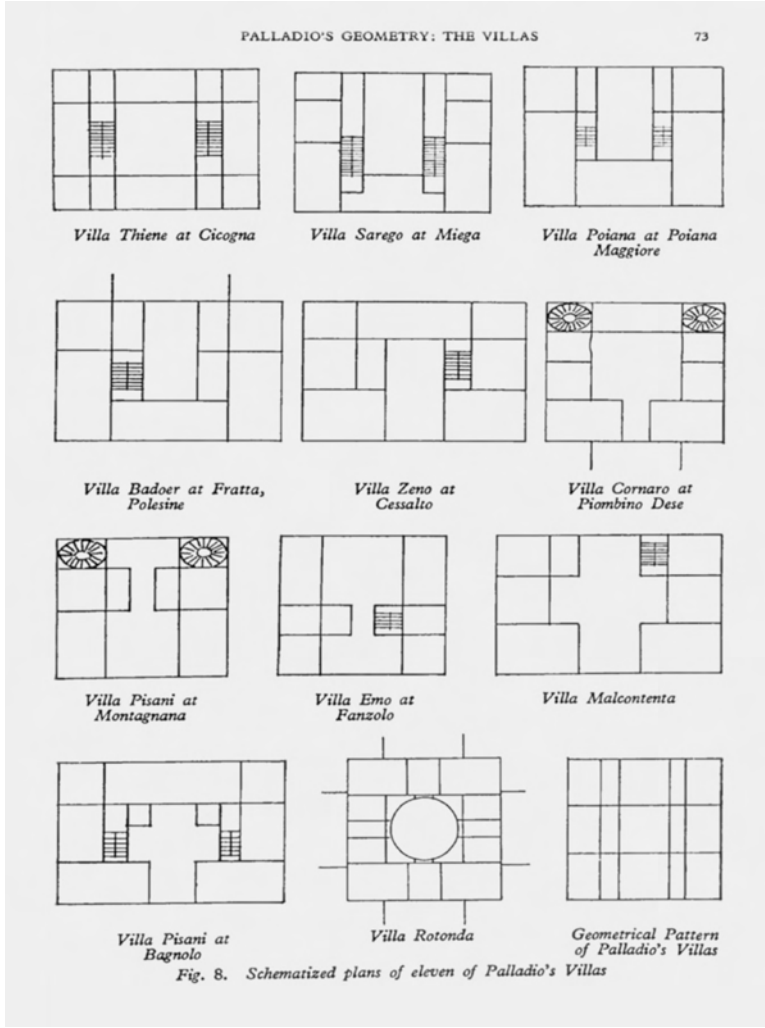
2.
Paul Frankl: Analyses of Churches from Principles of Architectural History: The Four Phases of Architectural Style 1420–1900.

readers, a system that would perhaps not just underpin the disposition and presentation of the work in print, but also explain the mechanisms behind its production (fig. 1).

Palladio's systematic representations lend themselves as ideal objects for the formalist architectural historiographies of the mid-20th century. Their methods of diagrammatical abstraction and systematization were in fact not that dissimilar from Palladio's own, as exemplified in Paul Frankl's typographical abstractions of configurations of Renaissance churches (Frankl 1968) (fig. 2). Following the logic of estrangement underlying formalist methods of analysis, which, as in literary analyses of Russian formalism, attempted to dislodge form from its context to discover what was intrinsically literary, or in this case, architectural (Jameson 1972), Rudolf Wittkower further isolated Palladio's readily abstracted plans in search of their »Palladian-ness«. By focusing exclusively on the master houses of the villas and disregarding their wings or »barchesse«, as much as signs of their productive and social functions as architectural responses to the landscape and conditions of their locations, Wittkower effectively foregrounded them as abstract compositions disentangled from their contexts. Most of this foregrounding was already implicit in *The Four Books*, which relegated any information external to the abstracted architecture of its graphical conventions to text, as noted by Kurt W. Forster (Eisenman et al. 2000). In Wittkower the geometric syntaxes of the villa's central bodies became the artistic and symbolic expression of a humanist culture manifested through proportions, ratios, and symmetries, rather than indexes of political, social, and economic conditions. James Ackerman explained how this »almost religious« commitment to an idea of universal harmony was closely lodged in post-war architecture and zealously sought by the likes of Wittkower after a war that they saw as the result of political extremism (Cohen/Delbeke 2018).

As part of this investment of meaning into abstract form, Wittkower and his most eminent follower, Colin Rowe, saw the plans of villas in *The Four Books* as indices of Palladio's own intellectual work, of his tinkering and struggles. Their analysis would reveal the cognitive processes behind the genesis of his architecture, even if the plans in the book were known to be subsequent idealized versions of the actual buildings.[2] This identi-

2 Or as Wittkower asked: »What was in Palladio's mind when he experimented over and over again with the same elements? Once he had found the basic geometric pattern for the problem villa, he adapted it as clearly and as simply as possible to the special requirements



3.
Rudolf Wittkower: Diagrams of Palladian Villas from Architectural Principles in the Age of Humanism, *1949.*

fication between Palladio's thoughts and the abstract geometric patterns traced by Wittkower and Rowe on the plans of the villas followed the principles of gestalt psychology driving formalist analysis in art (Jarzombek 1999). As gestalts, mental entities born from Palladio's toils, these patterns synthesized the disparate intentions and specific conditions of each project into a whole. These figures signified Palladio's architectural essence, the »principle« resulting from and governing his mental operations. During the following decade, and especially through Rowe's influence, this reading of Palladio became the template for architecture as a humanist autonomous discipline. This disciplinarity was based on being literate in, on being able to read and write, these abstractions of form that were its »principles«. These defined its theoretical domain, what was intrinsically architecture. Writing and reading these forms, or rather drawing and perceiving them, defined architecture as more than a mere response to economic, social, or technological demands (fig. 3).

Enter the Computer

In 1985 Richard Freedman, a student at Yale majoring in Computer Science, enrolled on a course that art history professor George Hersey was teaching on the Italian Renaissance. Interested in architecture, Freedman studied a copy of Palladio's *Four Books* that he had borrowed from a close relative, particularly the over 40 villas and buildings from *The Second Book*. Seen through his programming habits, he explained how *The Second Book* constituted a database, one sufficiently large and regular so that it might allow him to abstract some underlying design rules, so that a computer could be programmed to generate Palladian villas. With Hersey's encouragement this became the theme for his course assignment and an article in the journal *Architectura* (Freedman 1987). It finally led to their collaboration on the book *Possible Palladian Villas* (Hersey/Freedman 1992), which explained the development and consequences of Freedman's program.

In contrast to the art theoretical ideas of Wittkower and Rowe, Freedman's digital encoding of Palladio recast his work into the different technologies

of each commission. He reconciled the task at hand with the ›certain‹ truthof mathematics which is final and unchangeable. The geometrical keynote is, subconsciously rather than consciously, perceptible to everyone who visits Palladio's villas and it is this that gives his buildings their convincing quality.« (Wittkower 1944: 111).

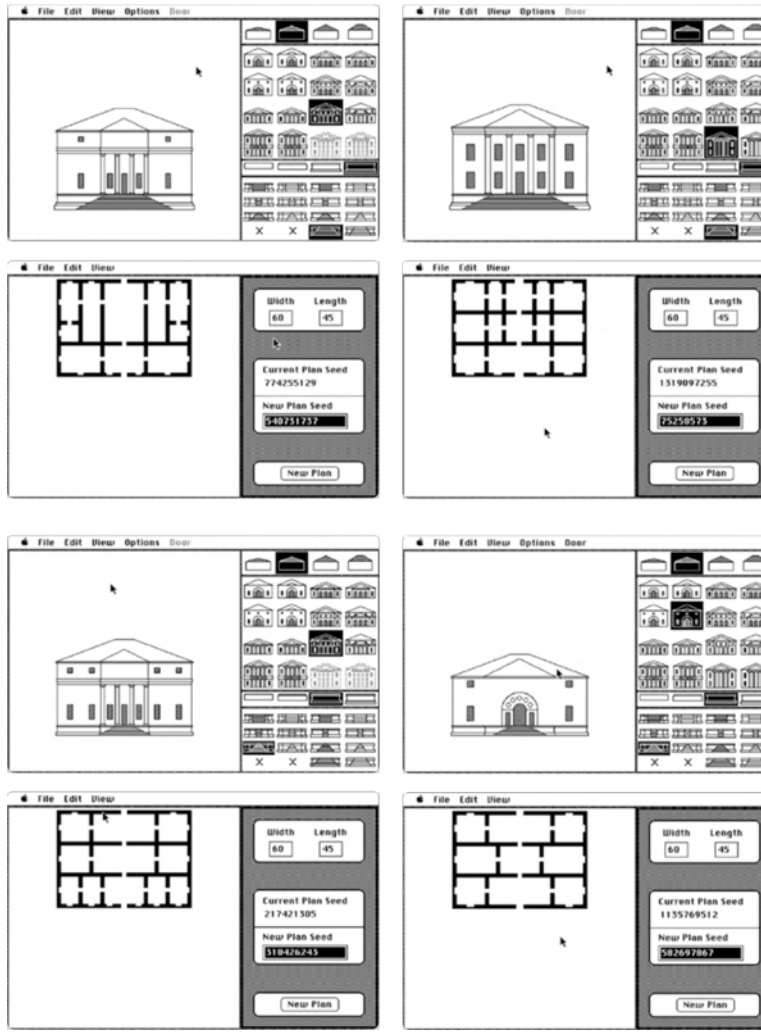
and ideologies of programming. Though codified through computer science, programming as a practice also consists of a multitude of conventions, habits, and tacit know-hows that regulate how to develop the complex artifacts that are programs. The practices Freedman deployed came from his own story with computers: As did many other programmers, he learned how to code as a teenager using the popular but limited BASIC programming language, which during the 1970s became the entry point for anyone not studying science or technology at university and wanting to learn how to use computers. Yet despite the increasing availability of computers that enabled teenagers like Freedman to learn programming, writing programs were still a complicated business in the early 1980s. Anyone wanting to compile larger programs beyond what BASIC interpreters could execute needed access to machines that were often too expensive to be personally owned. Their availability at Yale, Freedman explained, allowed him to learn how to put larger and more complex programs together (Richard 2022). Things were changing rapidly though. In 1984, just a couple of years before Freedman took Hersey's course on the Italian Renaissance, the Apple Macintosh had made the use and programming of graphics available to a general public and successfully commercialized the *Graphic User Interfaces* (GUI) first developed at Xerox PARC during the 1970s. Freedman first used the larger computers available at Yale to write the programs explained in *Possible Palladian Villas*, but the popularity of the Macintosh gave him the idea of distributing his programs along with the book. After many months of working at a Macintosh SE using the Aztec C compiler, he could finally run his interactive software; the result, he described, was the feeling that »Palladio was inside my Mac« (fig. 4).

Debugging

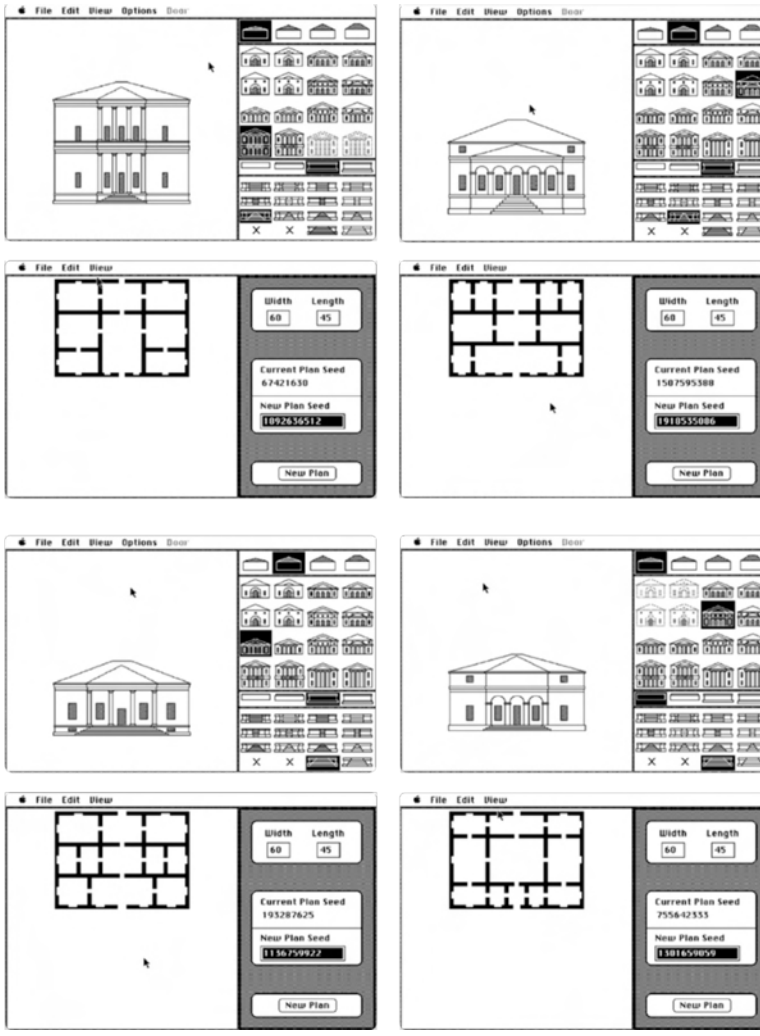
Freedman's initiation into programming in BASIC was that of the self-taught enthusiast, rather than the result of his more formal and academic studies in computer science. This may seem anecdotal, but this influenced his reading of Palladio, especially compared to the parallel efforts of William Mitchell and George Stiny, who instead used »shape grammars«, a mathematical formalism first proposed by them in 1971 to encode the generation of Palladian villas (Stiny/Gips, 1971; Stiny/Mitchell 1978b, 1978a; Mitchell, 1990). Most relevant to this case were the programming methods that began with BASIC. Mainframe computers such as the IBM 700/7000 series, first available to governmental organisms, universities, and corporations during the

1950s and 1960s, operated under what is known as »batch mode«: programs were first written into »batches« of punched cards that would run during some allocated time in a mainframe. The results would then be printed out or otherwise punched back into cards to be processed further. Since computers were in short supply, time allocated in a mainframe was considerably expensive. Access to these scarce computational resources was limited to scientists and technicians in the universities and corporations that could afford them. The BASIC language that allowed Freedman and many other teenagers to learn programming was an effort to make programming available to a wider public as computers became increasingly available. Developed first at *Dartmouth College* in the early 1960s, BASIC wanted to be a user-friendly language for non-scientists. Rather than batch mode, BASIC initially made use of time-sharing, the technique to grant access to many users to the same computer and the basis of multi-tasking in today's computers. What this implied, compared with »batch mode«, was interactivity: programs could now be written by anyone with access to a teletype-like terminal (later substituted by a keyboard and a screen) linked to the mainframe, and executed immediately. The increasing availability of computers, which had prompted this change in the first place, also meant much cheaper computer time; now it was affordable to waste it running incorrect programs with bugs and errors. Programming changed then from a process of carefully engineering code to one of iteratively writing and testing programs, observing their behavior, and modifying them accordingly, what is generally known as »debugging«. It became a sort of conversation between a programmer and a computer which, given a set of instructions to execute, would either answer with its results, often not necessarily the expected ones, or with error messages, to which the programmer would respond by rewriting the program. The text of *Possible Palladian Villas* and the code that is at its core mirror this process of software production and which informed the personal habits and practices of programmers like Freedman. A close reading of the code that formed the basis of *Possible Palladian Villas* shows how the technical conditions of writing programs also shaped the conceptual framework within which Palladio was transposed into the computer (Hersey/Freedman 1992).[3]

3 Richard Freedman kindly provided me access to the code written in C language of »Palladio«, the software for the Macintosh that was published at the same time as *Possible Palladian Villas* and distributed in a floppy disk.



4.
Richard Freedman: Screen captures of a Macintosh Emulator (Mini vMac), running System 6, and the Palladio *software.*



5.
Richard Freedman: Screen captures of a Macintosh Emulator (Mini vMac), running System 6, and the Palladio *software.*

Setting up the Conversation

The result of these »conversations» between Freedman and the computer was the code, written in the C language, that ran *Palladio* in the Macintosh. All C programs have a function or subroutine called **main();**[4] this is the entry point for the execution of the program, the starting point from where its labyrinthine structure will unfold in time. An inspection of *Palladio*'s **main()** discloses a record of the technical conditions at the end of the 1980s when it was written **main()** called for example another function, **iwindows()**, which set up the monochrome display of 512 × 342 pixels of the Compact Macintosh. This limited screen real state, considered high resolution at the time, presented a design challenge to Freedman, who had to use it as efficiently as possible both to interact and display the results of *Palladio* (Miranda Carranza 2022). **main()** took care of dealing with all the preliminaries of the program: it initialized all parts of the interface such as fonts, windows, menus, and dialogues, and managed all the necessary memory, a requirement in a language like C. It also set up the bitmap image in which to draw the plans and eventually the facades of the generated villas (with **setbitmap()**, called from **startplan()**). The first drawing in this image would be an undivided rectangle defining the generic perimeter of the villa (via **drawroom (pr)**, also called from **startplan()**). All the drawing was done using the Macintosh QuickDraw 2D Application Programming Interface (or API) for the Classic Mac OS operating system and which defined the operations, such as drawing a line or a rectangle, on a Macintosh. QuickDraw is still accessible from contemporary MacOS versions more than 30 years later, a digital fossil lodged in the operating systems of 2023. Unravelling the function call in **main()** also exposes the interesting transfer of typographic conventions into digital screens: The resolution of the Macintosh display followed the convention originated in mechanical printing of 72 points per inch, translated in this case to a resolution of 72 pixels per inch (Apple Computer, Inc. 1994). Correspondingly, all translations between dimensions on the plans of Palladian villas and those on the screen were done using a constant defined in the program as **PIXPERFT** or pixels per Vicentine feet, defined in the **plan.h** file and called through **startplan()**. **startplan()** in turn

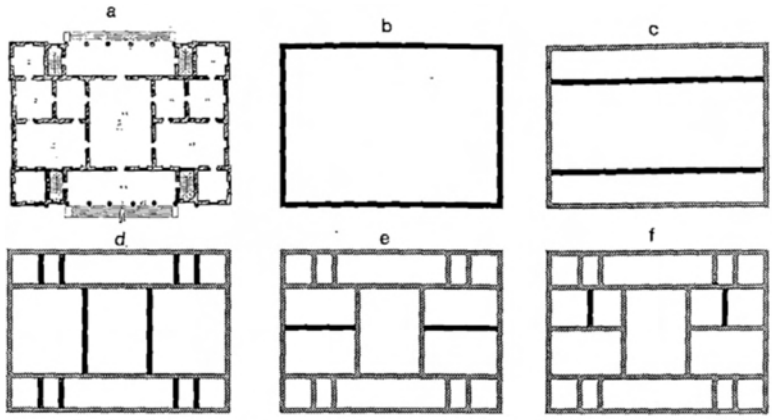
4 Function is a sequence of instructions that are grouped under a name that can be invoked anywhere else to run it. Functions can receive data as input for instructions and output or »return« data back to wherever in the program they were called from.

contained all the necessary steps to generate the plans of Palladian villas, and it was also called through the **eventloop()** function, which would take care of generating Palladian villas after pressing the »new plan« button by the user. Called here in **main()**, **startplan()** generated the first plan to display by the software. Besides taking care of setting up the data necessary to calculate a plan, **startplan()** contained the kernel of *Palladio*, the **split()** function, which encapsulated the automation of Palladio's design process. Its code was the final result of the conversation between Freedman and the computer. **split()** had as its input parameter a *pointer* to a *structure* describing a room in the villa (initially the whole undivided perimeter of the villa).[5] The result returned by **split()**, its output, was another *pointer* to the data of the left-most and top-most room of the subdivisions generated by the program, a position in the generated plan that would allow the program to access all other rooms in the villa.

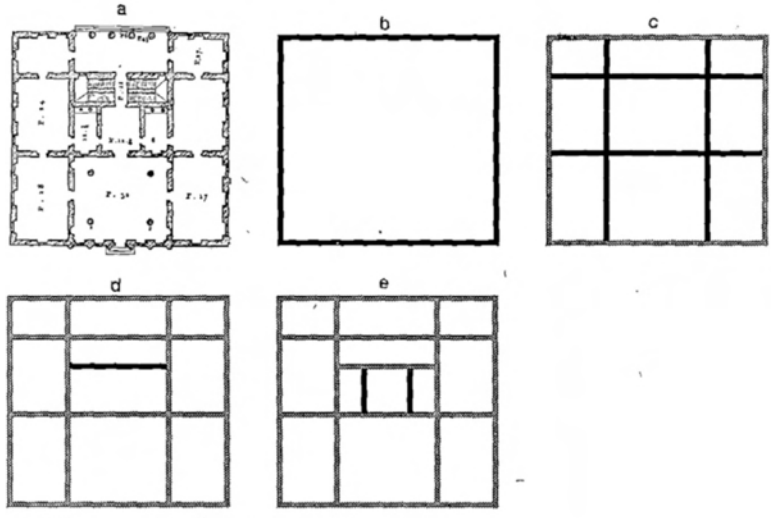
```
main()
{
  itoolbox();
  iwindows();
  imenus();
  idialogs();
  iprint();
  startplan();
  eventloop();
  terminate();
}


itoolbox()
{
  InitGraf (&qd.thePort);
  InitFonts();
  FlushEvents (everyEvent, 0);
  InitWindows();
  InitMenus();
  TEInit();
```

5 A pointer in C is an index to data, an address in the computer memory. In this particular case it pointed to a »structure«, a collection of values that in *Palladio*'s code described the properties of a room.



2. *Split description of the Villa Valmarana*



3. *Split description of the Palazzo Antonini*

6.
Richard Freedman: Recursive subdivision process, from A Computer Recreation of Palladian Villa Plans, *1987.*

```
  InitDialogs (NULL);
  InitCursor();
}

terminate()
{
  unallocplan();
}
```

Divide and Conquer

The interesting thing with **split()** is how it operated on the room description it got as its input: first, it subdivided it into smaller rooms (if possible), and then called **split()** again with each of the new smaller rooms as input, to be further processed and subdivided. This programming technique is called recursion and it produces the programming equivalent to a *mise en abîme*, a matryoshka doll of telescoping code in which a function self referentially calls itself. Recursion enables the defining of a large and complex task – in this case, splitting a large room representing the perimeter of the whole villa into the many smaller ones making up a Palladian plan – as made up of smaller versions of the same task – splitting any room into a couple of smaller ones. This type of procedure is called a »divide and conquer algorithm«, a method for factorizing jobs common in programming. Freedman programmed how **split()** would decide to nest splits into other splits, making up a recursive subdivision process that would generate the room layouts of any possible villa (fig. 6).

The generation of these recursive subdivisions was fundamentally different from the gestalts at the center of Wittkower's analyses. Rather than »principles« in the guise of elemental or ideal forms, the program used a set of »heuristics« or rules of thumb to decide how to split, or not, a room. Their development followed the process explained in *Possible Palladian Villas* of trial and error, of the logic of »debugging« typical of interactive programming and software development. Faced with the production of »improvable« villas by the code, Freedman iteratively ran and tested, and added and tweaked a set of rules that would make the subdivisions of rectangles more probable, more like the plans in *The Four Books*. There was no essential or underlying principle there, just the accumulative result of being able to produce, at the press of a button, an almost limitless number of variations in applying the same rules, and of modifying the code depending on the results.

```
pROOM split (pr)
pROOM pr;
{
  SPLITTYPE stype;
  SPLITRATIO sratio;
  pROOM ptopleft;
  int atts = 0;
  MYBOOLEAN yessplit;

  if (pr == NIL)
    return();
  if (pr->stage == 0)
    roomcount = 1;

  resetcontext();
  if (yessplit = splityn (pr, roomcount)) {
    do {
      if (yessplit = (atts++ < MAXATTEMPTS)) {
        getstype (pr, &stype);
        getsratio (pr, &stype, &sratio);
      }
      else
        break;
    } while (lookahead (pr, &stype, &sratio, _H));

    if (yessplit) {
      ptopleft = dosplit (pr, &stype, &sratio);
      incroomcount (pr, &stype);
      split (ptopleft);
    }
  }
  if (!yessplit) {
    split (pr->right);
    split (pr->down);
    return (pr);
  }
}
```

This cumulative work, »the conversation« between programmer and computer, is still readable in the code of **split()**. The series of steps given in the book, the trials and errors that made its second chapter, entitled »Planmaker«, (the same ones summarized in Freedman's article in the *Journal Architectura*) (Freedman 1987) appear in code as a set of decisions layered on top of each other, different bits of text that modulate the general behavior of previously encoded assumptions. But even if Freedman treated the plans in the *The Four Books* as merely empirical data for his code, his programming could not bypass the influence of the architectural culture of the second half of the 20th century. Thus, the proportional system suggested by Wittkower, and which presented the villas as equivalents of musical compositions, also regulated the subdivision scheme of *Palladio*, now transformed into an operational procedure rather than a symbol of the humanist spirit of the Renaissance.[6] As a computer science major, Freedman had little stake in the ideological investments architects had made in Palladian architecture and *The Four Books*. Since their identification by Wittkower, the absence or presence of equivalent formal »principles« in buildings separated what Rowe described as utilitarian answers to specific problems from concerns with the universal problem of architecture (Rowe 1956). The iconographic and gestaltist understanding of form behind this idea of »principle«, with its emphasis on human conception and perception, had little place in the computational makeover of Palladio. Hersey, the art history professor, was fully aware of Palladio's role in the discipline and the consequences of downgrading his work to a mere mechanical procedure, of the danger to »have

6 The only procedural description of proportional relations in *The Four Books* that could be transcribed as an algorithm refers to the proportions of the rooms, rather than of the whole plan: »By numbers it will thus be found: The length and breadth of the room in feet being known, we'll find a number that has the same proportion to the breadth as the length has to the number sought. This we find by multiplying the lesser extreme with the greater; because the square root of the number which will proceed from the said multiplication, will be the height we seek. As for example, if the place that we intend to vault be nine foot long, and four wide, the height of the vault will be six foot; and the same proportion that nine has to six, six also has to four, that is the sesquialteral« (Palladio 1738: 28). Palladio discussed the proportions of the rooms at length (proportions that he does not always follow, not even in the edited version of the plans of his buildings in the *Four Books*), but not of a system to order these throughout the whole building. This idea, with the abstraction of walls to lines, tracing general geometrical relations in the plan, are Wittkower's invention and discovery.

devalued the originality and genius of this architecture« and »have reduced Palladio to a game«. But he turned the argument around however, presenting instead the idea of *The Four Books* as almost a work of conceptual art. In this view, *The Four Books* consisted in the description of certain rules describing »procedures for assembling given parts into new wholes«, rules that a reader may want to reuse (Hersey/ Freedman 1992: 1). According to Hersey and Freedman these rules were given by Palladio in an applied from, rather than explicitly stated: hidden precepts for the combination or elements rather than the iconographic figures of Wittkower and Rowe. Writing a program to produce the plans in *The Second Book* was simply to accept Palladio's challenge of discovering the rules behind his systematization. Palladio's buildings became reinterpreted after their digital representation by Hersey and Freedman as the first representatives of a type of game-like architecture identified by the recurrent application of a rule-based principle on a corpus of work. Palladio's example would be followed by Claude-Nicolas Ledoux customs houses, Le Corbusier's villas, Frank Lloyd Wright's prairie houses, or even the prefabricated systems and modular kits of the 20th century. Hersey and Freedman called this type of architecture »paradigmatic«. In it, buildings were produced similarly to how sentences are produced following grammatical rules. *The Four Books*, after the precedent set by Francesco di Giorgio and Sebastiano Serlio, were then a set of specimen plans, doorways, windows, or columns that seem to ask to be »conjugated« according to some rules in order to produce architecture (Hersey/Freedman 1992: 8–9). In the challenge of deciphering Palladio's language game, the computer would be taught to design, or rather speak, Palladian villas. According to Hersey and Freedman, the difference with applying the rules by hand was in the computer's ability »to calculate a huge number of possible permutations and combinations based on Palladio's rules«. The computer would do this instantly and straightforwardly, in contrast to how »an unaided human being« would (Hersey/Freedman 1992: 10).

But to teach the computer to speak »Palladian« it was necessary to establish its grammar by a computer that could only dutifully conjugate it, and a programmer who could only tentatively define it and subjectively compare the results against an existing corpus of plans. Rather than an in-depth analysis of the diagrammatic drawings of *The Four Books*, the process of decoding Palladio's game consisted instead of the progressive adjustment of elimination of error, a definition of Palladio's architecture more by what it isn't than what it is. To find what Palladio did, it was necessary to discover »everything

he would not do«. Lacking any personality of its own or any idea of the programmer's intention, the computer would simply do what it was told, making every rule »explicit and unambiguous« (Hersey/Freedman 1992: 10).

This process of iterative and pragmatic approximation, rather than one based on ideal forms, would be the basis of a new type of historiography that Hersey and Freedman proposed more than 30 years ago. A historiography consisting of writing software, and which would remove »architectural connoisseurship from the realm of instinct and sets it within that of the verifiable«, where articulating the immanent rules of architecture would have the advantage to »etch out, with hitherto unexperienced clarity, the procedures and habits that distinguish this great architect from all others« (Hersey/Freedman 1992: 12).

Machine Psychologies

Incidentally, the verifiable representation of what was only the realm of the instinct was one of the ideological foundations of programming. The invention of programming languages during the 1950s was closely linked to the propositions of cognitive psychology, which, during the same period begun explaining the internal mechanisms of thought, including instinct, as computations. Palladio, the software, can be seen then as the recasting of architectural theorizations influenced by the premises of gestalt psychology – the immediacy of visual perception, the subsumption of the parts to an organizing whole – under the logocentric, fragmented, and procedural logic of the computer. This psychological dimension of programming is perhaps best summarized not in works of cognitive psychology, but in the foreword to *Structure and Interpretation of Computer Programs*, the textbook for the introductory course to programming at the Massachusetts Institute of Technology (MIT), where Alan J. Perlis wrote how »Every computer program is a model, hatched in the mind, of a real or mental process« (Abelson/Sussman 1983). This statement encapsulates the equivalences between thought processes, language, and logic that, as the basis of analytical philosophy, were also the starting point for programming languages, artificial intelligence, and cognitive psychology at the end of the 1950s.

The first step in the method put forward by Hersey and Freedman was to create a language to describe Palladian plans, a notation, using the recursive subdivision process discussed earlier, that could account for the room configurations in the villas in *The Second Book*. This was not a description of

geometry, form, or shape, but a notation of the process for generating it. Being a procedure, rather than a figure, permitted the suggestion of a correspondence with Palladio's own cognitive processes through the equivalences between programs and thoughts underpinning programming. To strengthen the possibility of this equivalence, Freedman and Hersey suggested that Palladio himself would have thought in terms of the recursive »divide and conquer« logic of their program, by identifying a similar subdivision technique in Palladio's description of a method to design entablatures in *The Four Books* (Hersey/Freedman 1992: 46).

The representation of Palladio's hypothetical thought processes as programs followed a pragmatic logic of approximation made up of tentative adjustments, patterned by the »conversation« between Freedman and the computer. Whereas Wittkower's analysis of eleven Palladian villas led to the synthesis of an ideal twelfth villa, an imaginary patter underlying all the others, Hersey's and Freedman's approximations were never conclusive. Despite their unambiguous nature, the programs that would »etch out, with hitherto unexperienced clarity« the procedures and habits of Palladio remained a tentative hypothesis that ruled out their consolidation into a final and idealized schema. Rules of thumb, or *heuristics*, were at the heart of this pragmatic approach. Their use as part of an ad-hoc accumulation of adjustments is clear in the structure of the **split()** function, which, to anyone that can read code, shows the process of embedding loops, conditional statements, and functions like **lookahead()** that are ostensibly solutions to problems that occurred as the program was written, rather than as implementations of an algorithm carefully planned in advance.

In the context of operations research, »heuristic« were proposed by artificial intelligence pioneer and Nobel Prize laureate in economics, Herbert Simon, in the 1950s as way of expanding the field's area of applicability. Whereas previous methods in operations research and management science dealt with well-structured problems, »heuristics« would help to address those tackled with judgment and guess (Simon/Newell 1958). »Heuristics« enabled programs to be written that mimicked thinking habits learned from experience, rather than simply implementing mathematical methods for problem-solving. Programs could become »theories in a completely literal sense, of the corresponding human processes«. These would be verified by comparing the behavior of a computer running the program with the behavior of a human performing the same task (Simon/ Newell, 1962). While »heuristics« were not mentioned anywhere by Freedman and Hersey, their

rationale clearly drove their propositions as an ingrained technique in the practice of programming. It is through the ideas behind »heuristics« that the **split()** function above can then be seen as a theory »of the corresponding human processes« of Palladio, as Simon would put it, or as Hersey and Freedman intended of »the procedures and habits that distinguish this great architect from all others« (Hersey/Freedman 1992: 12).

But »heuristics« and the cognitive motivations behind programming were not the only psychological models involved in the writing of *Palladio*. The very »conversation« between Freedman and the computer had also been theorized under ideas from psychology. Besides BASIC, the expansion of the potential user base of computers from the1960s onward demanded other ways to increase computer literacy. Teaching children how to program and the use of computers in teaching became a worthy research pursuit during the 1960s. Seymour Papert, Co-director of the *AI Lab* at MIT, developed a pedagogical framework that had the idea of »debugging« at its center. Papert reimagined the conversational model that was becoming standard in computer-human interaction through the constructivist psychology of Jean Piaget, with whom he had worked in Geneva. In the conversation between a child and a computer, concepts that were initially intangible and abstract would slowly be given concrete form. Errors in the code written by a child would play an important role in the process: they would force the child to understand the reasons behind them to fix them, and in the process improve the concretization of their knowledge, both in their mind and in the unambiguous notation of a computer program (Papert/Solomon 1972; Papert 1980).

Besides the influence that Papert's ideas had in the development of modern graphic user interfaces at Xerox PARC (Kay 1972), later mass-marketed through the Apple Macintosh that also run Palladio, his Piagetian theorization of »debugging« as a way of constructing knowledge fits quite aptly the process followed by Freedman and Hersey. The idea was to first »let loose« a program using »incipient Palladian rules«, which would come up with plenty of mistakes from which to learn (Hersey/Freedman 1992: 53). The code would then be refined, slowly constructing a model of Palladio through this interaction between programmer and computer. The C code of *Palladio* is a trace and record of this conversation and of the iterative concretization of a hypothesis of Palladio's own working process. What both »heuristics« and a constructivist understanding of »debugging« highlight is the lack of a »principle«, of an ideal. This was substituted instead by the deployment of guesses, tests, and experiments that could, but may not, correspond

to Palladio's design process. In their capacity to endlessly produce plans that look somehow »Palladian«, their rhetoric differs importantly from Wittkower's or Rowe's idealism. »Palladio«, the program, is not a representation of a hidden ideal but a hypothetical and pragmatic encoding of Palladio's thought processes.

Possible Palladian Villas takes over many of Wittkower's premises: the formalist isolation of the abstract form of a building, a process started by Palladio himself in *The Four Books*; the proposal of proportion and ratios as an underlying and unifying logic for the geometry of the plans; the reading of these plans as indices of Palladio's cognitive work, and the possibility to reconstruct, or at least speculate on, the mental processes involved in this work. In their effort both showed a psychological leaning that was not necessarily explicit but part of the respective discourses of formalism in art and of programming and software design. Both were invested in the explanation of the villa's plans in *The Four Books* as the play of forms in Palladio's mind, rather than on the material, technical, economic, and social conditions for their production as buildings and their reproduction in print. Both implied a subject, Palladio, as their source. But the writing systems employed in each case are also fundamentally distinct: one graphical and typographical, where Palladio's figure as an architect is presented as one of the first examples of the very humanistic culture it promoted; the other algebraic and mechanical and where subjectivity had been substituted with an objective rationality. Wittkower's logic was fundamentally visual. Form, broadly understood as »gestalt«, worked as a principle unifying both the intention behind artistic work and its perception. Freedman's and Hersey's was instead sequential and algebraic, based on written symbols and their processing by computers. These two forms of writing produced two distinct architectural subjectivities: the first, one in which the production and experience of form create the respective subjects of the author and of the reader. The second, one where human capacities and actions become disembodied mechanical procedures and fragmented actions that can be indistinctly performed by humans or machines. They thus propose two Palladios: one, the humanist artist, the prototypical Western architect with all its baggage; the other, a name that is a placeholder for a set of procedures, of »inventions«, that are anything but inalienable.

Conclusion

Etymologically, collaboration means to work together; in the writing of the software *Palladio* and of the book *Possible Palladian Villas* that supplemented it, work was distributed between two humans and the computer. As the non-human partner in the collaboration, the computer brought the capacity to work relentlessly, to carry out Freedman's commands unquestioningly and to present back the results of their execution. Their relation was patterned by the explicit and implicit discourses, the technologies, and the practices shaping and regulating their interactions. The relationship between Hersey and Freedman was framed on the other hand by its institutional settings – the difference in roles at a prestigious *Ivy League* university – their generational difference – Freedman a young computer science student, Hersey a humanist scholar born before computers had even been invented – but more importantly by the two literacies represented by each of them: a humanistic and academic, one funded on the reading and understanding of texts and artistic artifacts as human products; and a literacy of algebraic writing where letters, symbols, and signs lost any reference to a human voice. The collaboration comprised then a double translation, each of them carried out mainly by Freedman and Hersey respectively: a first one of the plans for *The Four Books* which, seen as data, became indices of a hidden program to generate them, a set of tacit rules used by Palladio that could be transcribed into the algebras of programming; and a second translation of these technical inscriptions into their meaning to the historical conditions of the Renaissance and the historiographical context of the second half of the 20th century, to which analyses of Palladio had been key. The conclusions reached by this double translation could not escape the discursive practices and ideological formations of its two writing systems and their inherent contradictions. In their common identification of Palladio as the principal source of the plans for *The Second Book*, their contrasting ideas of subjectivity, one human and the other mechanical, reproduced the very conflict between system and authority that the villas in *The Four Books* present to modernist thought.

But the lack of followers of their software-based historiography also points to the distance between these two forms of writing and their conflicting ideologies, and emphasizes the serendipitous circumstances behind the improvable encounter between Freedman and Hersey. More than 30 years later, and in the midst of a renewed interest in AI, we see many of the

themes that played out in the writing of the software and the book. Rather than the playfulness behind their propositions, today the mechanical reproduction of what were believed to be exclusively human performances conjure instead apocalyptic futures without any nuance or critical distance, perhaps because the gap between these two literacies, despite their current inextricable interdependence, is at least as large as it was three decades ago.

References

Abelson, Harold/Sussman, Gerald Jay (1983): *Structure and Interpretation of Computer Programs*. Cambridge, MA: MIT Press.

James S., Ackerman (1966): *Palladio*, Harmondsworth: Penguin.

Apple Computer, Inc. (1994): »Imaging with QuickDraw«, Apple Computer, Inc.

Cohen, Matthew A. (2018): »Proportional Systems in the History of Architecture: A Conversation with James S. Ackerman«, in Cohen, Matthew A./Delbeke, Marteen (eds), *Proportional Systems in the History of Architecture*, Leiden: Leiden University Press, 511–522.

Eisenman, Peter/Stanford Anderson (2000): »Virtual Palladio: Two Views«, MIT Department of Architecture Lectures Series. Cambridge, MA: Massachusetts Institute of Technology, Department of Architecture records.

Frankl, Paul (1968): *Principles of Architectural History: The Four Phases of Architectural Style, 1420–1900*, Cambridge, MA: MIT Press.

Freedman, Richard (1987): »A Computer Recreation of Palladian Villa Plans«, in: *Architectura. Zeitschrift für Geschichte der Baukunst*, 17/1, 58.

Hersey, George/ Freedman, Richard (1992): P*ossible Palladian Villas: Software*, Cambridge, MA: The MIT Press.

Hersey, George/ Freedman, Richard (1992): *Possible Palladian Villas: (Plus a Few Instructively Impossible Ones)*, Cambridge, MA: The MIT Press.

Jameson, Frederic (1972): *The Prison–House of Language: A Critical Account of Structuralism and Russian Formalism*, Princeton, NJ: Princeton University Press.

Jarzombek, Mark (1999): *The Psychologizing of Modernity: Art, Architecture and History*. Cambridge: Cambridge University Press.

Kay, Alan (1972): »A Personal Computer for Children of All Ages«, *Proceedings of the ACM Annual Conference: Volume 1* .

Mitchell, William (1990): *The Logic of Architecture: Design, Computation, and Cognition*, Cambridge, MA: The MIT Press.

Miranda Carranza, Pablo (2022): *Interview with Richard Freedman, Author of Possible Palladian Villas (Plus a Few Instructively Impossible Ones)*, unpublished interview.

Palladio, Andrea (1570): *I Quattro Libri dell'Architettura* – English translation: *The Four Books of Architecture*, transl. from the the first edition by Isaac Ware, London: Printed for R. Ware, at the Bible and Sun, on Ludgate-Hill, 1738.

Papert, Seymour (1980): *Mindstorms: Children, Computers, and Powerful Ideas*, New York: Basic Books.

Papert, Seymour/ Solomon, Cynthia (1972): »Twenty Things to Do with a Computer«, in: *Educational Technology*, 12/4, 9–18.

Puppi, Lionello (1975): *Andrea Palladio*, Boston: New York Graphic Society.

Rowe, Colin (1947): »The Mathematics of the Ideal Villa: Palladio and Le Corbusier Compared«, in: *Architectural Review* 101, 101–104.

Rowe, Colin (1956): »Chicago Frame«, in: *Architectural Review* 120, 285–289.

Herbert, A. Simon/Allen, Newell (1958): »Heuristic Problem Solving: The Next Advance in Operations Research«, in: *Operations Research*, 6/1, 1–10.

Herbert, A. Simon/Allen, Newell (1962): »Computer Simulation of Human Thinking and Problem Solving», in: *Monographs of the Society for Research in Child Development*, 27/2, 137–150.

Stiny, George/Gips, James (1971): »Shape Grammars and the Generative Specification of Painting and Sculpture«, in: *IFIP Congress* 2.

Stiny, George/Mitchell, William (1978a): »Counting Palladian Plans«, in: *Environment and Planning B: Planning and Design* 5/2, 189–198.

Stiny, George/Mitchell, William (1978b): »The Palladian Grammar«, *Environment and Planning B: Planning and Design* 5/1, 5–18.

Wittkower, Rudolf (1944): »Principles of Palladio's Architecture«, in: *Journal of the Warburg and Courtauld Institutes* 7, 102–122.

Wittkower, Rudolf (1949): *Architectural Principles in the Age of Humanism*, London: Warburg Institute, University of London.