

**Exploring the Technics of Computer Representation:
Three 'Prepared' Instruments for Plotting, Meshing and Rendering.**

by

Jonah Marrs

B.A. History
McGill University, 2010

M.Arch
University of Toronto, 2016

SUBMITTED TO THE DEPARTMENT OF ARCHITECTURE IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN ARCHITECTURE STUDIES
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

JUNE 2018

©Jonah Marrs. All Rights Reserved.

The author hereby grants to MIT permission to reproduce and to
distribute publicly paper and electronic
copies of this thesis document in whole or in part in any medium now
known or hereafter created.

Signature redacted

Signature of Author:.....

.....

Department of Architecture
May 24th, 2018

Signature redacted

Certified by:.....

.....

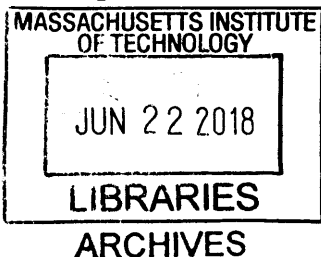
Terry Knight
Professor of Design and Computation
Thesis Advisor

Signature redacted

Accepted by:.....

.....

Sheila Kennedy
Professor of Architecture
Chair of the Department



THESIS COMMITTEE

Terry Knight

Professor of Architecture
Department of Architecture, MIT
Thesis Supervisor

Andrew Witt

Assistant Professor in Practice of Architecture
Graduate School of Design, Harvard University
Thesis Reader

**Exploring the Technics of Computer Representation:
Three 'Prepared' Instruments for Plotting, Meshing and Rendering.**

by
Jonah Marrs

Submitted to the Department of Architecture
on May 24, 2018 in Partial Fulfillment of the
Requirements for the Degree of Master of Science in Architecture
Studies

ABSTRACT

In *Translations from Drawing to Building*, Historian Robin Evans draws attention to the "blind spot between the drawing and its object" as a productive site in architecture. Instead of a "uniform space through which meaning may glide without modulation", the "substratum" of this "gap" is uneven and unpredictable, a space of "entropy" and a "locale of subterfuges and evasions." Through their passage, "things can get bent, broken or lost on the way." Evans sees architects exploring "deviations" and "potentialities" here by "maintaining sufficient control in transit so that more remote destinations may be reached."

Today we manipulate drawings and 3D models in software programs and visualize them at human-computer interface moments like the display, the plotter and the 3D printer. Meanwhile, at internal machine-to-machine interfaces, algorithms regularly carry out translations, their "labor behind or below the threshold of perception" (John May, "Field Notes from the Instruments Project" in *JAE*). Theorist Paul Virilio describes virtual drawings and models here as "signals in the electronic currents of a closed-system, readable by machines but neither visible nor legible to humans" (*The Vision Machine*).

My thesis gives form to these unseen computer translations between different representations of our drawings and objects. The three sections: Plotting, Meshing and Rendering, materialize the underlying infrastructure of the computer representation technology we interact with daily as designers, exploiting a form of "time-lapse" image to freeze moments that happen within an instant inside the computer. The thesis asks architects to question our relationship with our tools and invites us to explore Evans' "blind spot" and claim this fertile territory for the designer, leveraging our visual-spatial and interdisciplinary knowledge culture for new experimental ends.

Thesis Supervisor: Terry Knight
Title: Professor of Design and Computation

ACKNOWLEDGMENTS

I wish to express my most sincere gratitude and appreciation to the many people who have played a significant role in my education and work. My mother, Julia, and father, Iain, I thank most of all. You are the most supportive and encouraging family one could ever hope for.

From MIT I want to thank my thesis advisor Terry Knight who was a tireless resource, with incredible patience and generosity. I want to thank Irina Chernyakova for putting up with me and curating my exhibit at the Keller Gallery. Thank you Daniel Cardoso Llach who featured my work at his Computation Conference at CMU. Theodora Vardouli kindly invited me to speak at McGill. Jake Read of the CBA helped me plan complex mechanical things and connected me with parts.

From Harvard, my Reader Andrew Witt offered many useful ideas for my project through conversations and his Mechatronics Optics class. Rachel Vogel familiarized me with many important works of New Media Theory, in part through Laura Frahm's New Media Course and Rendering Conference, and also brought her brilliant mind to thinking through the implications my project. I'm thankful to have had Matthew Allen to bounce ideas off of, he pointed me towards important references like Ivan Sutherland's "twinkling display" and kindly set up an exhibit at the GSD with me. Tom Haye's Physics 123 Class at Harvard was helpful in developing my electronics skills and was a valuable hands-on history of technology primer.

From the Canadian Center for Architecture, Tim Walsh and the Digital Archive team were a great help understanding how software has evolved since the 1990's and how obsolete media can be digitized.

From the University of Toronto, my M.Arch Thesis Advisor Laura Miller gave me the space and resources to explore. An independent study with John May had a big impact on my work, my project is indebted to the forthcoming Instruments Project with Zeynep Çelik Alexander. Mauricio Quiros pointed me towards Media Archaeology and the CCA and was a moral anchor for me at the school.

I want to thank the following people for productive conversations: Eytan Mann, Alex Bodkin, Lucy Lui, Amanda Ghassaei, Emily Watlington, Pedro Zylbersztajn, Sam Ghantous,

ACKNOWLEDGMENTS

Nick Pecula, Nick Montfort, Joseph Choma, Axel Kilian, Takehiko Nagakura, Scott Donaldson, Dave Reeves, Paul Christian, Matthew Blunderfeild, Skanda Lin, Skylar Tibbits, Timothy Hyde, William O'Brien Jr., Daniel Rosenberg, Tobias Putrih, Caitlin Mueller, Carrie McKnelly, Mark Jarzombek, Caroline A. Jones, Daniel Marshall, Elizabeth Galvez, Mason White and Peter Osborne.

I thank the Autodesk Pier 9 San Francisco staff, especially Noah Weinstein and Vanessa Sigurdson. Autodesk Engineers very generously shared their insights with me into how their software functioned:

Scott Conover - Autodesk Boston
Sankarappan Gopalsamy - Autodesk Boston
Ryan Schmidt - Autodesk Toronto
Cameron Westland - Autodesk San Fransisco
Ravi Krishnaswamy - Autodesk San Fransisco
Aaron Bartholomew - Autodesk San Fransisco
Kyle Machulis - Autodesk San Fransisco
Eric Haines - Autodesk San Fransisco

I want to recognize the funding I received from MIT to travel and conduct research in 2017:

MIT GSC Travel Grant
Schlossman Research Fellowship
Bill Mitchell ++ Research Award
Harold Horowitz Research Award
Council for the Arts Grant
Council for the Arts Director's Grant

Thank you to the extra-MIT departments which supported my research through lecture invitations:

Oxford University's Internet Institute
Harvard's Department of Visual and Environmental Studies
The University of Southern California's Art History Department
Carnegie Mellon School of Architecture
McGill University School of Architecture

To anyone I may have left out I apologize and thank you for your support as well.

TABLE OF CONTENTS

i. Introduction	7
Method	11
Precedents	14
ii. Plotting	22
Background	23
Prepared Plotter.....	25
Plotting Series No.1: Degree of Time-Lapse.....	28
Plotting Series No.2: Type of Input.....	32
Plotting Series No.3: CNC/CAM Sequencing.....	39
Conclusions	55
iii. Meshing	60
Background	61
Prepared Mesher.....	62
Meshing Series No.1: Type of Input.....	69
Conclusions	75
iv. Rendering	78
Background Pt.I.....	79
Background Pt.II.....	88
Prepared CRTC.....	93
Rendering Series No.1: Degree of Time-Lapse & Type of Input...	103
Rendering Series No.2: Randomizing Algorithm.....	106
Conclusions	110
v. Thesis Conclusions	111
Appendix	121
Bibliography	124

I. INTRODUCTION

I. INTRODUCTION

In *Translations from Drawing to Building*, Historian Robin Evans draws attention to the "blind spot between the drawing and its object" as a productive site in architecture. Instead of a "uniform space through which meaning may glide without modulation", the "substratum" of this "gap" is uneven and unpredictable, a space of "entropy" and a "locale of subterfuges and evasions." Through their passage, "things can get bent, broken or lost on the way." Evans sees architects exploring "deviations" and "potentialities" here by "maintaining sufficient control in transit so that more remote destinations may be reached."¹

Today we manipulate drawings and 3D models in software programs and visualize them at human-computer interface moments like the display, the plotter and the 3D printer. Meanwhile, at internal machine-to-machine interfaces, algorithms regularly carry out translations, their "labor behind or below the threshold of perception."² Theorist Paul Virilio describes virtual drawings and models here as "signals in the electronic currents of a closed-system, readable by machines but neither visible nor legible to humans."³

In *Builders of the Vision*, Daniel Cardoso Llach traces the engineers and technologists who worked on the Computer-Aided Design Project, a research operation funded by the US Air Force, active between 1959 and 1967, at MIT. Llach shows how the engineers "reconfigured discourses of design and design representation" in the "language of the computer", leading to a "new representation tradition expressed by indexical combination, in the computer, of Albertian perspectivalism and data processing." Llach then traces the renewed expression of the tropes of "neutrality, centrality and universality" of CAD promoted by the CAD Project team, along with their view of CAD "as a cybernetic 'man-machine' problem-solving engineer", into the current Building Information Modeling (BIM) environment.⁴ In fact, BIM does not offer a "universal space" where meaning

1 This paragraph: Robin Evans, "Translations from Drawing to Building" in *Translations From Drawing to Building and Other Essays* (1997).

2 John May, "Field Notes from the Instruments Project" (*JAE* 69:1 Spring 2015).

3 Paul Virilio, *The Vision Machine* (Cambridge: MIT Press, 1995).

4 This paragraph: Daniel Cardoso Llach, *Builders of the Vision : Technology and The Imagination of Design*. (Cambridge: MIT Press, 2012).

glides in a frictionless architectural *lingua franca*.⁵ Anthropologist of Science and Technology Diana E. Forsythe, outlines the specific bias in the engineering of knowledge-based systems not unrelated to CAD. Forsythe's research into the culture of the technical reveals a series of assumptions made by engineers which manifest themselves in the expert systems they design. David J. Hess' Introduction of Forsythe's *Studying Those Who Study Us* sums up some of her findings. He writes that Forsythe: "insists that the technical is itself cultural", and that that culture is one of "technical bias; decontextualized thinking; quantitative, formal bias (seen especially in the use of clinical trails for evaluation methods); a preference for explicit models; and a tendency to believe that there is only one correct interpretation (or reality) of events." Forsythe shows how the tools knowledge-engineers developed reinforced their world-view, along with ideas around gender, and "deleted the social".⁶

Architecture theorists Zeynep Çelik Alexander and John May's provocative Instruments Project is a series of historical and philosophical meditations on contemporary instrumental processes (rendering, scanning, modeling, etc.) designed by engineers like those Forsythe studied. Subtly drawing on German Media Theory starting with Friedrich Kittler, the Instruments Project investigates the implications of representation technologies on the culture of architectural thought specifically. Echoing Robin Evans' view that the site of representation is central to the architect's thinking, they see architecture's critical and conceptual frameworks and language as rooted in representation, the "preconceptual substrate from which the contemporary architectural mind fashions and understands itself." They argue that technical imaging has "rapidly and thoroughly displaced our entire history of representation as the primary site of reasoning and experimentation." In place, technical imaging has brought its own world-view, "silently posit[ing] an entire cosmological theory of life". The situation of forms of visualization specific to scientific knowledge dominating architectural discourse has led architects to find themselves in a state of "breathless amateurism", their intellectual grounding pulled out from

⁵ Daniel Cardoso Llach, *Builders of the Vision : Technology and The Imagination of Design*. (Cambridge: MIT Press, 2012).

⁶ This paragraph: David J. Hess, "Introduction", in Diana E. Forsythe, *Studying Those who Study Us: An Anthropologist in the World of Artificial Intelligence*. (Stanford: 2001).

underneath them.⁷

The following section outlines an experimental method for architects to analyze and critically engage contemporary mediating technologies and thereby move towards reclaiming Evans' productive translation spaces. Unlike the authors cited above, this thesis will attempt to engage in research through a graphic and spatial form of inquiry, drawing especially on the epistemological concepts of Johana Drucker's graphesis and Andrew Witt's Design Hacking.

⁷ This paragraph: John May, "Field Notes from the Instruments Project" (*JAE* 69:1 Spring 2015).

I. METHODS

In *Design Hacking: The Machinery of Visual Combinatorics*, Andrew Witt defines Design Hacking as the “intervening in established sequences of operation” of “corporate technological systems” to “divert[] computation from deterministic conclusions” and “industrial and institutional ends”, towards more “independent” and “experimental aims.” Witt traces the cultural heritage of the practice from early experimental drawing machines of the 1700’s through to avant-garde formal experiments of the 20th C and contemporary experiments with custom CNC tooling and Arduino. Witt describes the different kinds of “hacks” that are required by artists and designers as the technological medium evolved from mechanical to analog electronic and eventually digital code. The core methodology behind design hacking, however, remains consistent: “Through experimentation, designers interrogate a combinatorial range of visual possibilities latent in a system, finding the truth in the system itself.” Witt shows that the value of such experiments are largely epistemological; they open up architecture to new fields of knowledge. Witt writes:

“[T]hrough this backchannel of practical experimentation with technology, design gains a new range of activity and freedom. In the direct physical trace of the machine, the hack enables synthetic possibilities for both design and the knowledge culture of architecture.”¹

The concept of Design Hacking appears to fit well within Johana Drucker’s view of the distinction between the epistemological frames of *mathesis* and *graphesis*. In *SpecLab: Digital Aesthetics and Projects in Speculative Computing*, Drucker expels the myth of code as a pure, ideal and direct representation of human knowledge (“mathesis”) by force of its connection with the ideology of mathematics. Drucker shows that code is in fact an embodied, specifically-instantiated, material in itself, arguing that “no constituted expression exists independent of the circumstances of its production and reception.”

Drucker finds a “gap” similar to the one identified by Robin Evans: “There is always a space between expressed idea and expression of an idea, procedures capable of generating any number of material instantiations.” Drucker goes on to explore forms of representing knowledge in a materially embedded form (“graphesis”) which are

¹ This and last paragraph: Andrew Witt, *Design Hacking The Machinery of Visual Combinatorics*.

often visual and therefore ambiguous and multi-functional.²

Acknowledging the problem of the displacement of traditional architectural representation as outlined by Alexander, May and Llach, and with the goal to reclaim Evan's "gap" as a site for architectural mediation, I bring to bear Drucker's concept of *graphesis* and Witt's concept of *Design Hacking* to several representation technologies which architects interact with today. Through a series of hands-on experiments which reach into the nuts-and-bolts of very specific and elemental computer processes, this thesis will give visual and spatial form to near instantaneous computer processes created by engineers and reify them in material objects. These materializations will serve to emphasize the specificity of mediation technologies, in the face of their purported universality, and highlight the transformational influence of technological mediation in the face of it's purported model as a neutral vessel.

The experiments in this thesis are organized into three chapters. The first section, Plotting, will examine how computers prepare CAD drawings for plotting machines. The second section, Meshing, is a foray into how computers prepare forms for three-dimensional printing. The third section, Rendering, is an investigation into how a computer generates a display list for a monitor from a database of user-instantiated elements.

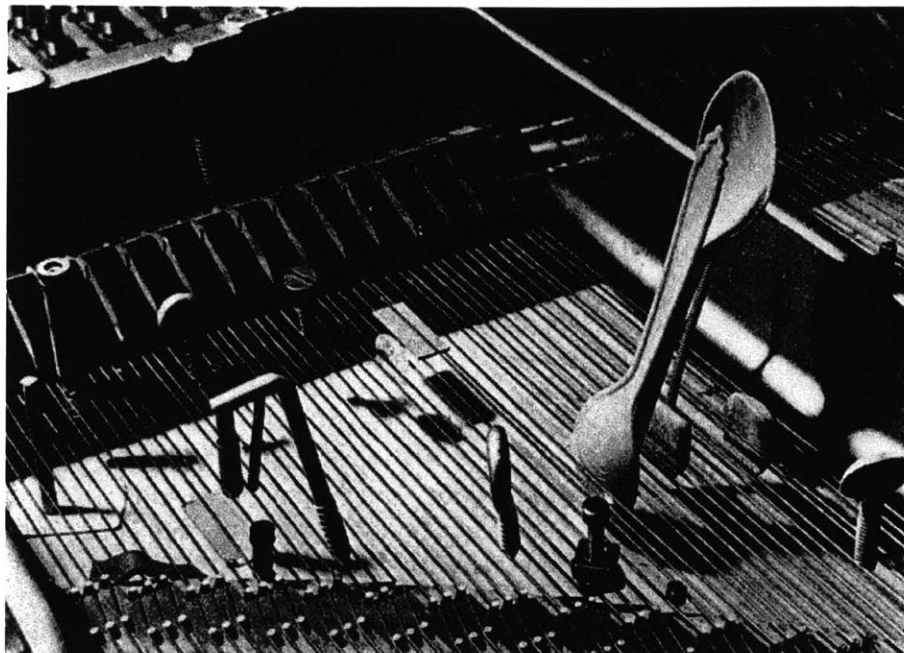
In focusing on three highly specific moments within the sea of machine to machine junctures which exist in the contemporary workflow, this project seeks to draw on the strength of extremely site-specific media histories of technical systems such as Sean Cubitt's study of the politics of Youtube's H.264 MPEG compression algorithms or Noah Wardrip-Fruin archaeology of Allen Turing's Love Letter Generator program. In both cases, a marriage of deep technical analysis is combined with historical study of the issues of class, race and gender; media technologies become a lens to study the culture which created them. In the case of this thesis, the analysis of technical systems and an attempt to reach deeper issues relevant to our society is conducted through the visual and spatial medium one could argue is native to the artist and architect.

The following section, Precedents, examines artistic projects which visually and spatially engage with representation

² This and last paragraph: Johana Drucker. *SpecLab: Digital Aesthetics and Projects in Speculative Computing*. (Chicago: 2009)

instruments such as the piano, the mobile phone app, the bound book format, Google Earth, the CRT television, and Deep Neural Nets. This thesis aims to conduct similar studies with representation instruments specific to the culture and practice of architecture.

I. PRECEDENTS: PREPARED INSTRUMENTS



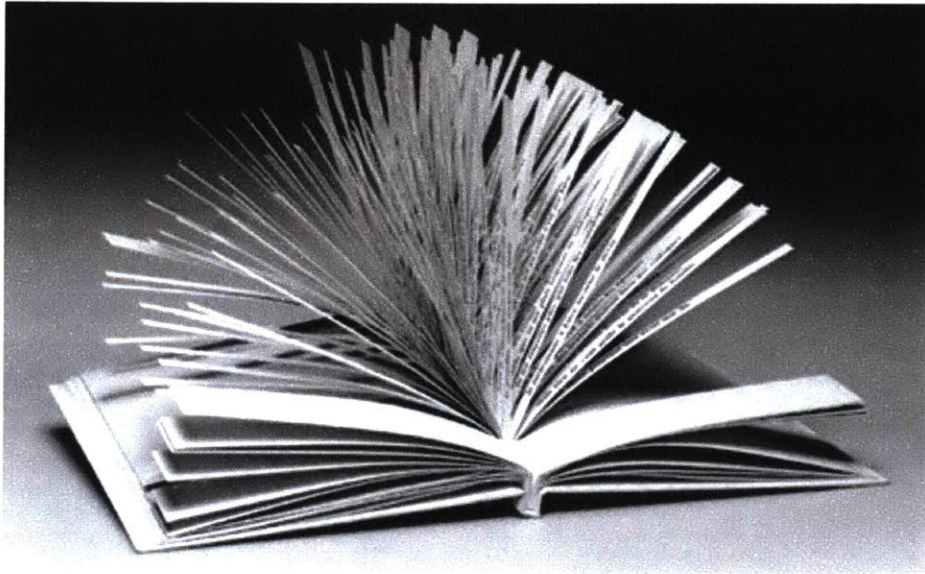
John Cage, Prepared Piano

By placing objects (often screws and bolts but also wool, bamboo, coins and weather stripping), known as "preparations", which interfere with the vibration of certain strings in the piano, new kinds of sounds (percussive, dampened notes, etc.) are available to the composer.¹

Cage's preparations not only allowed for different sounds, they also reframed the instrument itself as a space of exploration for the musician that invited a more intimate involvement with the physical working of musical tools. Physically laying hands on the inside of the instrument and manipulating the innards was the precursor to the modern day home-brew synthesizer with knobs for every kind of sonic parameter.

¹ Photo from <http://wmich.edu/mus-gened/mus150/preppiano.gif>

I. PRECEDENTS: PREPARED INSTRUMENTS



Oulipo, Cent Mille Milliards de Poems

Raymond Queneau of the collective L'Ouvroir de Littérature Potentielle (Oulipo) created Cent mille milliard de poèmes¹ along with mathematician Francois Le Lionnais in 1961. The book is collection of sonnets that has been "prepared" with a guillotine so that each line of text exists on a separate strip of paper, now free to be turned irrespective of the others. The prepared book creates the possibility for a nearly endless recombination of different lines from the group of sonnets, inviting the reader to participate in the creating of their own recombinant works of poetry.

This work demonstrates the generative possibilities that exist "between the lines" of traditional formats like the sonnet and the printed book. Here, the assumption that the page is the atom (the indivisible base unit) of the printed book is questioned, and as a result, a kind of latent energy which has accumulated in these formats over the years is suddenly released. The result brings the active role of the medium itself into focus, making the argument that the medium itself is a tool which can exploited as a source of creative inspiration.

¹ Photo from http://next.liberation.fr/livres/2014/11/21/1-oulipo-dans-une-forme-oulipique_1147974

I. PRECEDENTS: GOOGLE VISION PT.I



Postcards from Google Earth, Clement Valla

Clement Valla¹ discovered strange moments where any illusion of a seamless and self-same representation of the Earth appears to break down within Google Earth. Instead of representing moments of failure in the algorithms (glitches), his "Postcards from Google Earth" in fact represent "the absolute logical result of the system" which created them. While purporting to represent the surface of the planet, the images Valla captures and frames in fact bring attention to the way in which Google Earth generates the illusion of the seamless representation of the Earth.

This artistic method is insightful because it uses the tool itself to generate evidence of the working of the tool. Nothing is added to the system to make it "express" itself, examples are merely curated in a collection to produce a result simultaneously didactic and absurdly irrational. The resulting series of images creates the sensation that the everyday can easily tip into the uncanny.

¹ Photo from: <http://clementvalla.com/work/postcards-from-google-earth/>

I. PRECEDENTS: GOOGLE VISION PT.2



Along the Resolution Frontier, Erin Besler

Along the Resolution Frontier¹ maps the threshold at which Google Earth's resolution for urban areas gives way to lower resolution scanning used for less dense human settlements. Besler's camera captures absurd moments where houses plunge into the ground, billboards float in midair, apartment blocks are severed diagonally and where streets and cars are stretched vertically or truncated abruptly. More subtle juxtapositions also exist between different colors and textures of open ocean or dessert, often marked by the raised scar that is the blending of the two representations.

Erin Besler discovers that the "resolution frontier" of cities like L.A. and Mexico City differ from that of Chicago, for instance, where more waypoints were used to divide the city from non-city and more care taken to follow natural boundaries. Besler's work uncovers a social boundary, that between urban and non-urban, meeting a technical reality, the limited memory and resources for scanning and storing models. It draws attention to the construction of the representation itself and its provisional nature, the map of the earth as minimum viable product. The result undermines any pretension which could exist of an objective map of the world.

¹ Screenshot from: <https://www.erinbesler.com/along-the-resolution-frontier/>

I. PRECEDENTS: IMAGE RECOGNITION



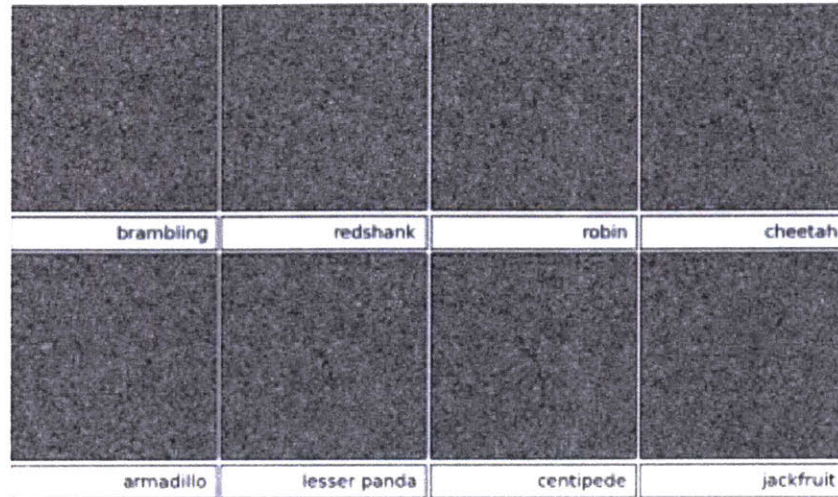
Faceswap

Faceswapping¹ is a practice that became popular in 2015-2016 when several apps were released to detect people's facial features and then map them on to one another on a mobile device. Users typically swap faces of women and men, children and adults, owners and their pets but also humans and inanimate objects susceptible to pareidolia-like interpretation (like wall plugs, patterns in wood grain, etc.).

The results are absurd, troubling and at times nightmarish: children holding miniature versions of their parents, and Dalmatians with outlandish (human) bodies. The faceswap exploits the fact that the reading of the face depends greatly on context (such as the size and width of the head, along with the rest of the body). However, the operation of faceswapping also reveals the algorithm's indifference to the human significance of the swapping operation, the uncanny ease with which a piece of software can imagine and describe such surreal situations as a pumpkin-headed man holding a pumpkin with his own face. As a result of its aptitude in dealing in the absurd, the algorithm can begin to feel native to the surreal in the public imagination.

¹ Photo from "Most Horrifying face swap EVER!" IMGUR, 2012.

I. PRECEDENTS: IMAGE RECOGNITION



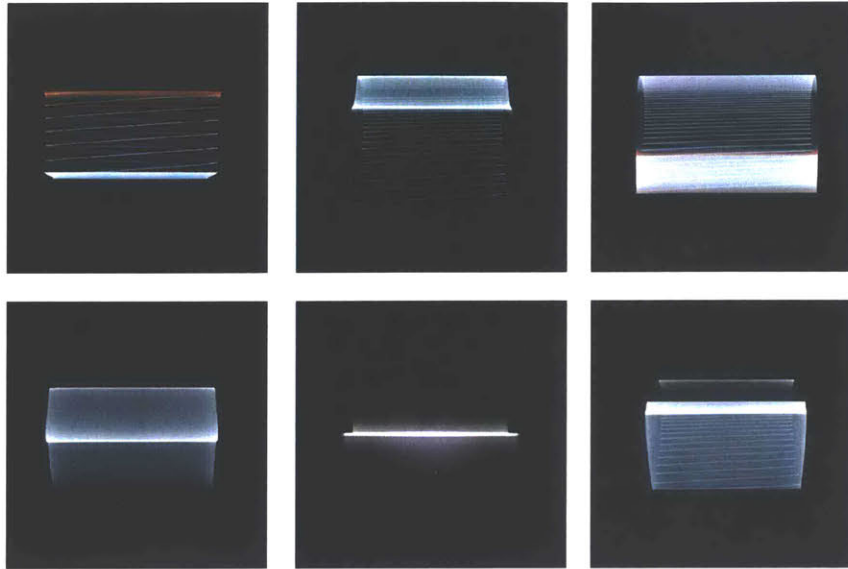
“[...]High confidence predictions for unrecognizable images”

Recently Deep Neural Networks (DNNs) have been achieving high degrees of performance in pattern-recognition tasks, especially in what are known as visual classification problems, and are approaching human degrees of discernibility. However, computer scientists at Cornell and U of Virginia have shown that it is easy to fool a DNN with simple image modification techniques. For example, the image of a school bus can be altered in a way which is imperceptible to humans to trick a DNN into misclassifying that image with 99.99% confidence.¹ Related studies show that images which appear completely unrecognizable to humans (such as white noise) can be manipulated to be recognized by DNNs as various species of animals.

The blatant misclassifications made by state-of-the-art DNNs reveal the degree to which human and DNN perception are fundamentally different, and therefore susceptible to different kinds of manipulation.

¹ Photo from Anh Nguyen, Jason Yosinski, Jeff Clune. *Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images* (Cornell: 2015).

I. PRECEDENTS: VIRTUAL ATMOSPHERE



Leuchtpunktordnungen , Stephan Tillmans

Stephan Tillmans¹ creates photo series of old color Cathode Ray Tube (CRT) televisions at the moment they are turned off. All manufactured products that leave a factory meet basic criteria but they exist within a range of acceptable dimensions and weights. Tillmans seizes a moment when mass-produced machines reveal their idiosyncrasy, contrasting the understanding of a single line of products being essentially homogeneous.

Tillmans hones in on a meaningful moment of the CRT's working: the precise moment when it ceases to create an illusion of an image. This moment was not designed nor intended to be seen. The moment becomes a window into the inner workings of the machine; the photographs expose how the CRT raster scans left to right, then moves to the next row from top to bottom.

By studying how something breaks down, like smashing particles together in a collider or studying crime to learn about society, one can learn much about how something is held together. By focusing in on a single moment in time, and comparing instances of these thin temporal slices, the range of behaviors of manufactured systems is revealed.

¹ Photo assembled from <http://www.stephantillmans.com/luminant-point-arrays/>

I. PRECEDENTS: VIRTUAL ATMOSPHERE



David O'Reilly's XYZ RGB

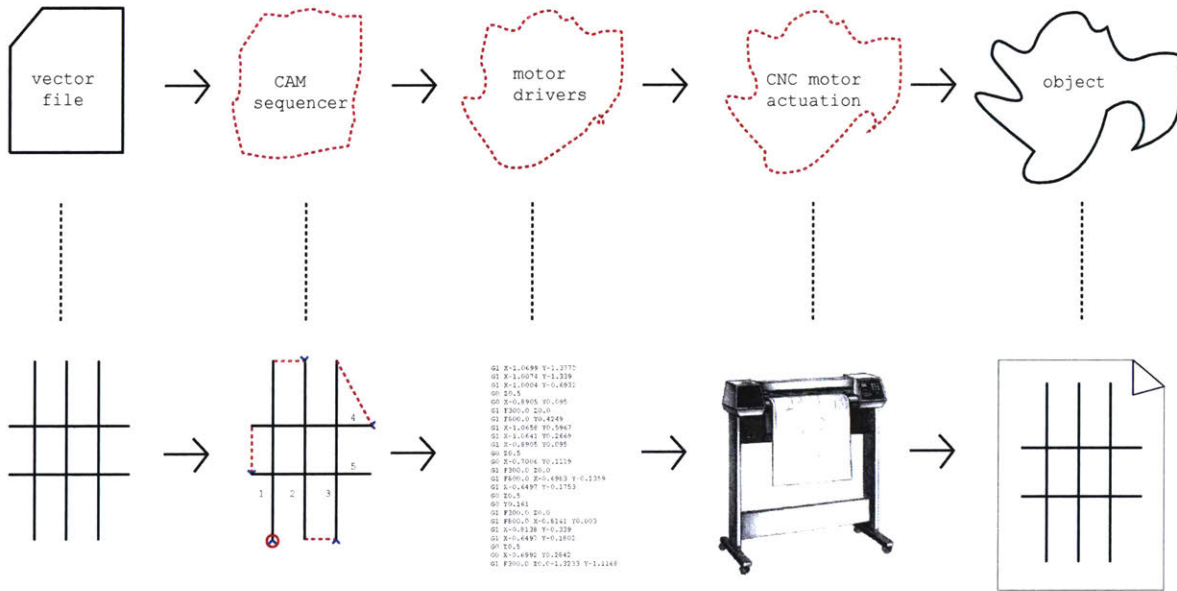
Irish filmmaker David O'Reilly¹ uses the technical artifacts of early 3D modeling software to tell the story of a boy leaving the small town for a job in the big city, along with a not so subtle meta-narrative concerning the nature of creativity and constraints in digital tools. To convey the instability of the main character's world in RGB XYZ, the camera orbits beyond the horizon line vertiginously, regularly flipping the entire scene upside down while the action continues. Camera singularity moments, which take place when the perspective approaches the origin of the 3D model, serve as scene transitions. Characters follow the non-Newtonian rules of the digital world, sliding through closed doors and turning around by rotating about arbitrarily-defined axes. Z-fighting, the coplanarity of two surfaces which present a quandary for rendering algorithms, and conflicts between object location constraints that create oscillating forms, are exploited to create rich, dynamic textures and environments.

Instead of trying to erase their mark, O'Reilly highlights the traces of the 3D modeling tools themselves to narrative effect. O'Reilly thereby pushes against the trend towards artists employing increasingly sophisticated digital tools that abstract the user further away from low-level animation operations and finds the ludic in a tightly-constrained minimalist creative process.

¹ Screenshot from <http://www.davidoreilly.com/#/rgb-xyz/>

II. PLOTTING

II. PLOTTING: BACKGROUND



Plotter Toolchain

What happens in the moments after we click print and transform a drawing on our screen into a drawing output by a plotting machine?

In the traditional Computer Aided Machining (CAM) / Computer Numerical Control (CNC) plotter or laser-cutter tool-chain, a CAM algorithm will first convert image vectors in a file into a sequence of tool paths, a series of lines and arcs transcribed in a language called G-Code. This sequence of tool paths is determined by an algorithm to be the most efficient based on minimizing distance traveled by the tool. (This path finding task may be referred to as the "traveling salesman problem" in the science of optimization.) Once a sequence of tool operations is optimized, a piece of software will send tool path commands to an external CNC machine which will receive the commands and execute them one-by-one, as predetermined by the machine's firmware settings, by sending signals to activate appropriate motor drivers in sequence.

These moments of translation, between a 2D vector image on our screen and a machine-mediated object, are the topic of the first set of experiments.

II. PLOTTING: BACKGROUND



Traveling Salesman Solving Algorithm

The traveling salesman solver¹ (or shortest path solver) looks for the shortest path between a series of destinations. In other words, given a series of points it must visit once, and only once, the algorithm optimizes for distance. In the context of a CNC plot, this algorithm minimizes the amount of needless travel (pen "lift-off") between drawn lines. Regardless of the order in which the lines in a drawings were made, the traveling salesman solver reshuffles them to make a drawing which will be executed in the least amount of time.

Many Shortest Path Solver (SPS) algorithms exist today such as Dijkstra, Bellman-Ford, A* search, Floyd-Warshall, Johnson, Floyd-Warshall and Viterbi. The SPS that I work with in this section come from proprietary software programs like CAMBAM and RhinoCAM.

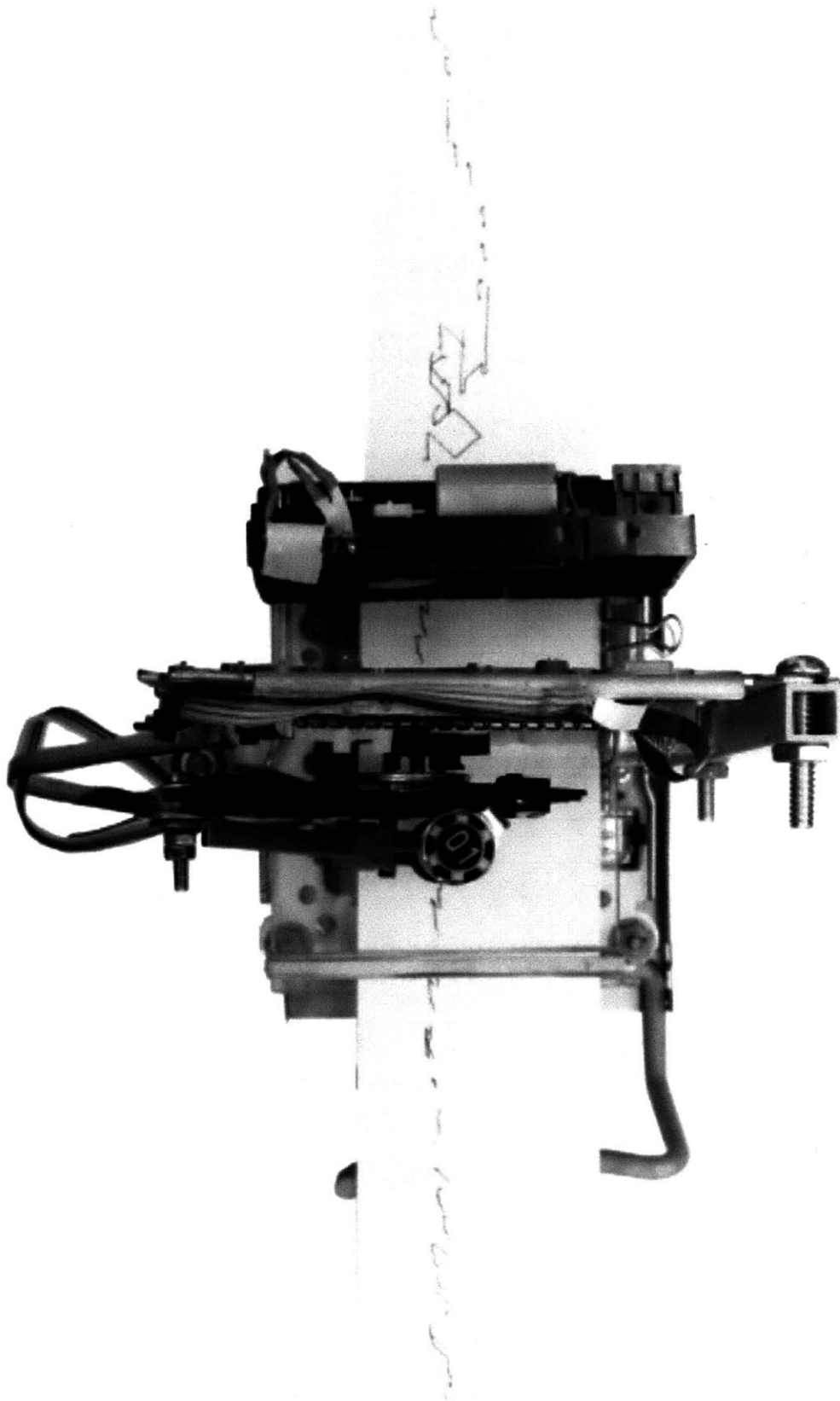
¹ Edited diagram from Jessica Yu, 48StatesTSP.png. Accessed at: https://optimization.mccormick.northwestern.edu/index.php/Traveling_salesman_problems

II. PLOTTING: PREPARED PLOTTER

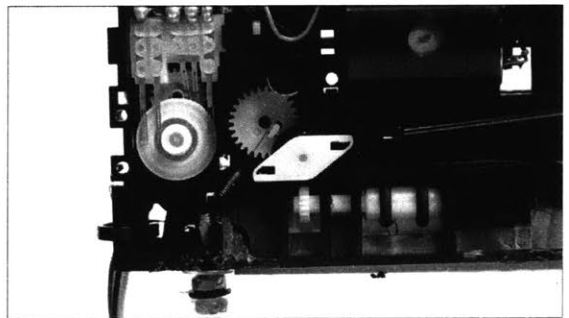
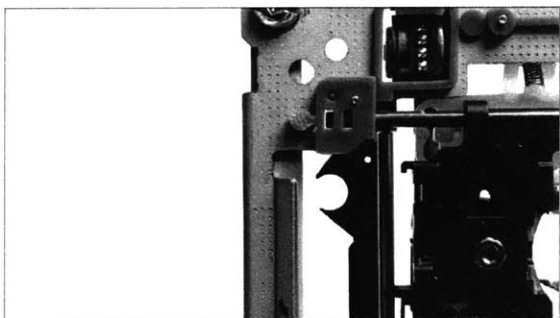
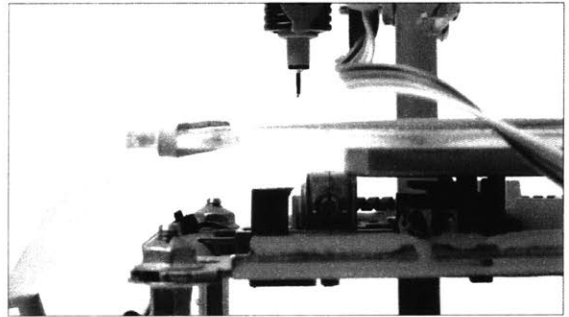
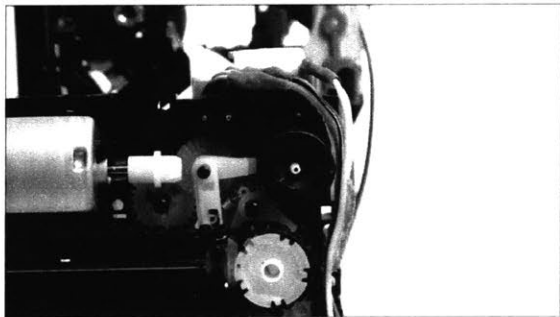
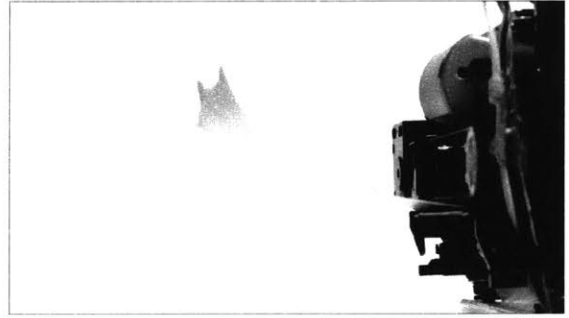
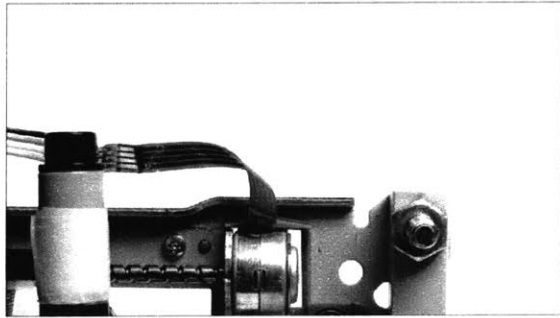
To investigate the "gap" that exists in the plotter tool-chain between a drawing and its optimized sequence of machine commands, a custom CNC drawing machine was devised to record the sequence of CNC machine movements as choreographed by the CAM software. In the spirit of John Cage's prepared piano, which allowed the composer to explore a new range of sonic possibilities inside the body of the piano, the prepared plotter allows for manipulation of the normal working of the plotter to explore new visual possibilities. This 'prepared' plotter was a bricolage of found objects: a compact photo printer, CD and DVD drives, and a printing calculator. Each component was designed for a different purpose (such as reading CDs, drawing circles, or printing receipts) and therefore each brings its own prescriptions to the assembly.

With this custom device, the algorithmically-optimized drawing sequences can be "stretched" in time to varying degrees to reveal the sequence of operations choreographed for the CNC plotting machine. This method, of visually stretching a process in time and recording it in a vertical strip where it can be juxtaposed with others, will be the main representation tool used for the experiments in this thesis and will be referred to as a "time-lapse drawing" in this section.

The original prepared plotter prototype was completed for my M.Arch thesis at the University of Toronto in May 2015.



Prepared Plotter (Top View)



Prepared Plotter Details

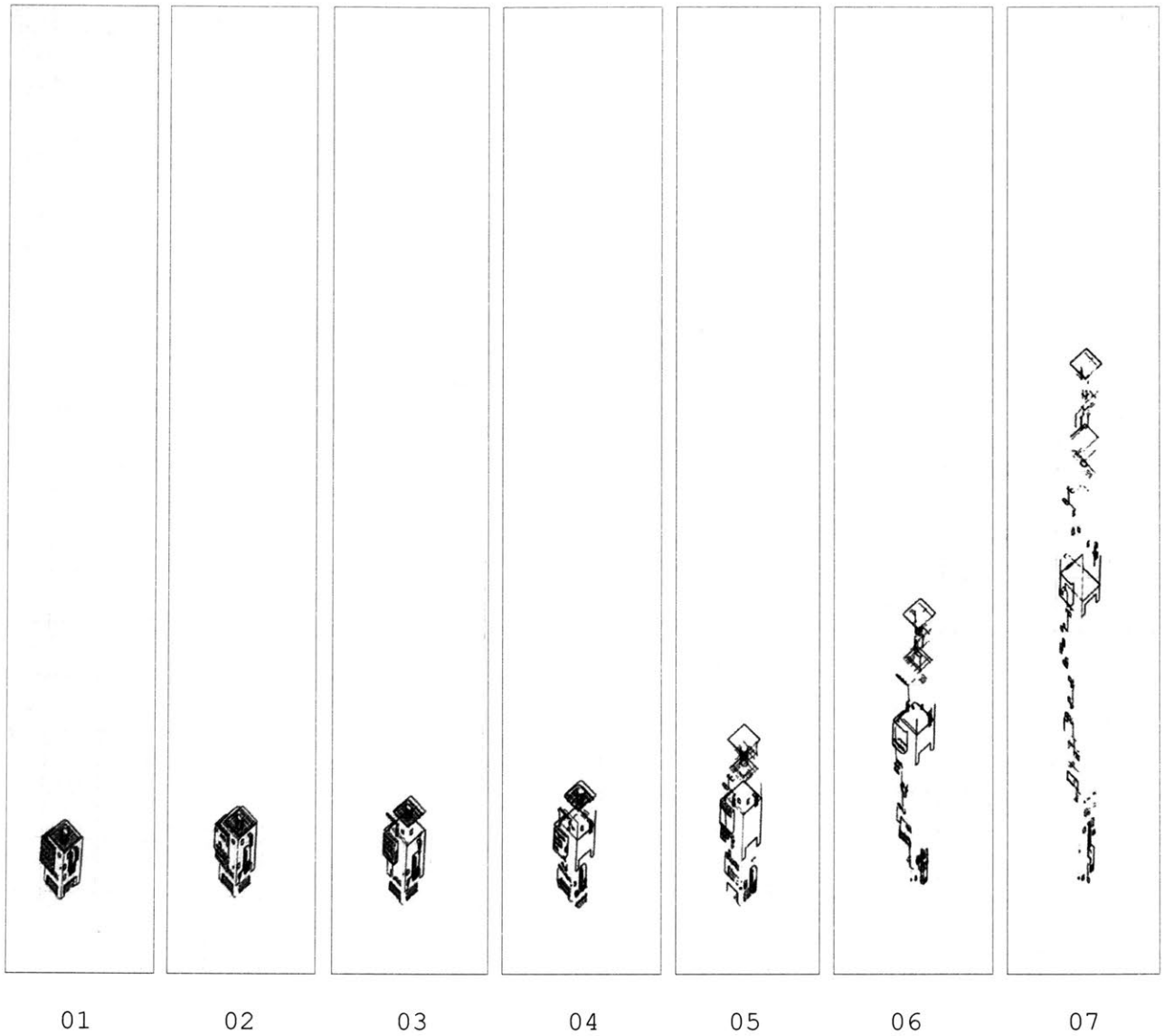
II. PLOTTING: DRAWING SERIES No.1

The 'time-lapse' drawings are created by pulling the paper away from a specially-designed plotter at regular intervals to interrupt a drawing process and mark the sequence of drawing over time. Instead of being interrupted continuously, the drawing machine is interrupted after a set number of machine operations are undertaken. This leads to drawing "clusters" separated by space, with the vertical direction representing the passage of time.

On the following page, the drawings on the far left are output by the prepared plotter without interruption. The drawing appears on paper as it appears on the computer screen, all lines existing at one and the same time. Moving to the right, the same drawing is produced but the drawing process has been interrupted (the paper has been pulled away from the machine) at increasing frequencies. Moving rightward, this creates an increasingly "time-lapsed" original drawing.

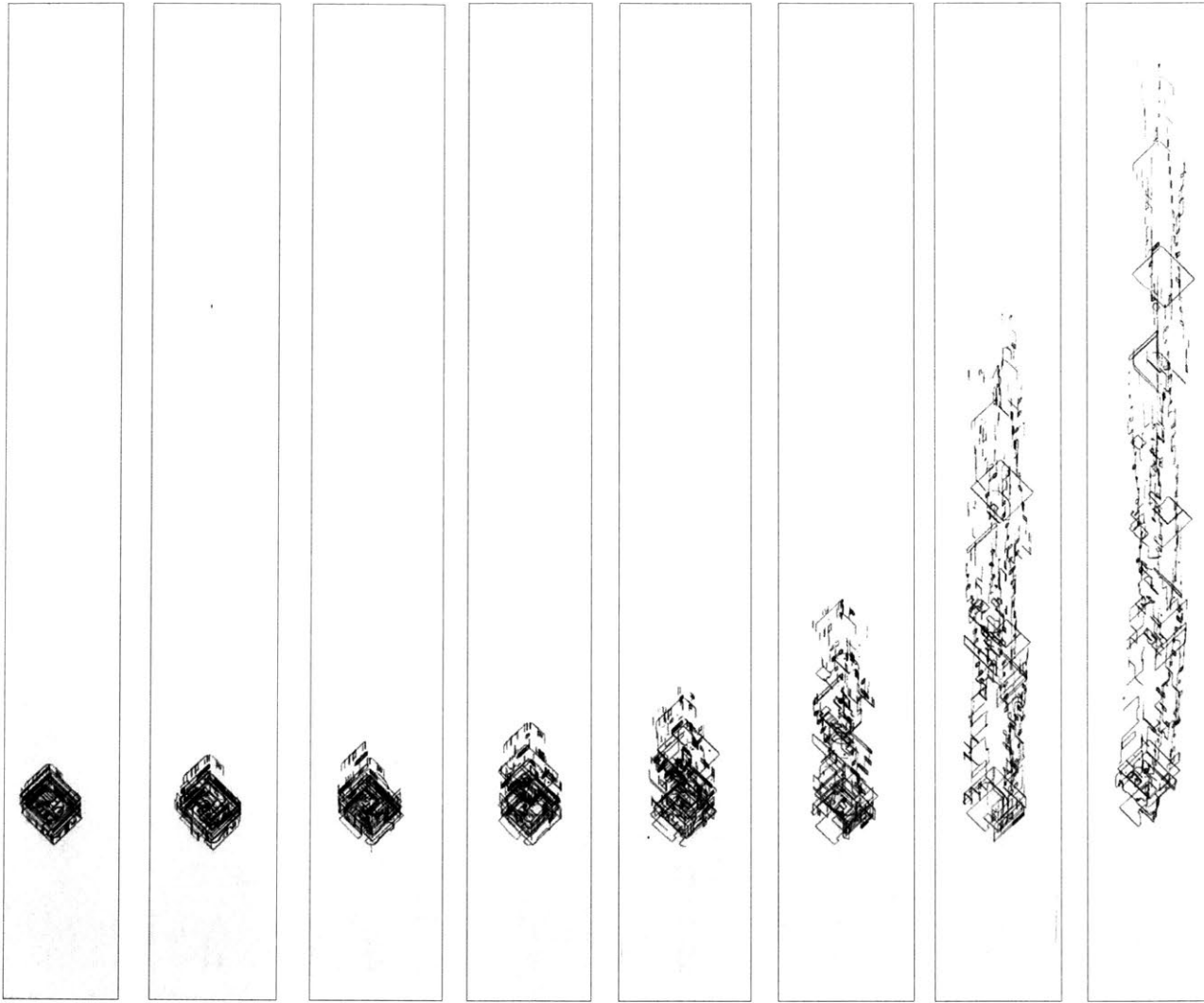
The Drawing Series No.1 has the goal of describing the "time-lapse" process through graphic means. Future series use this method to compare different ways of drawing the same drawing, or to compare different drawings time-lapsed with the same settings.

This drawing series was completed for my M.Arch thesis at the University of Toronto in May 2015.



01 0 interruptions
02 2 interruptions
03 4 interruptions
04 8 interruptions
05 16 interruptions
06 32 interruptions
07 64 interruptions

1A: O.M. Ungers Axonometric



01

02

03

04

05

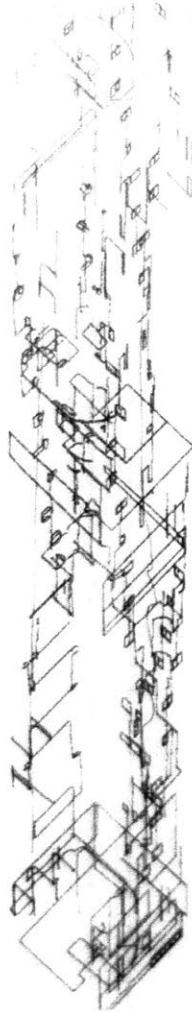
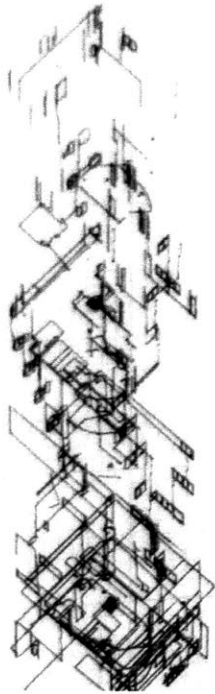
06

07

08

01 0 interruptions
02 2 interruptions
03 4 interruptions
04 8 interruptions
05 16 interruptions
06 32 interruptions
07 64 interruptions
08 128 interruptions

1B: Villa Savoye Axonometric



1B: Villa Savoye Axonometric Detail

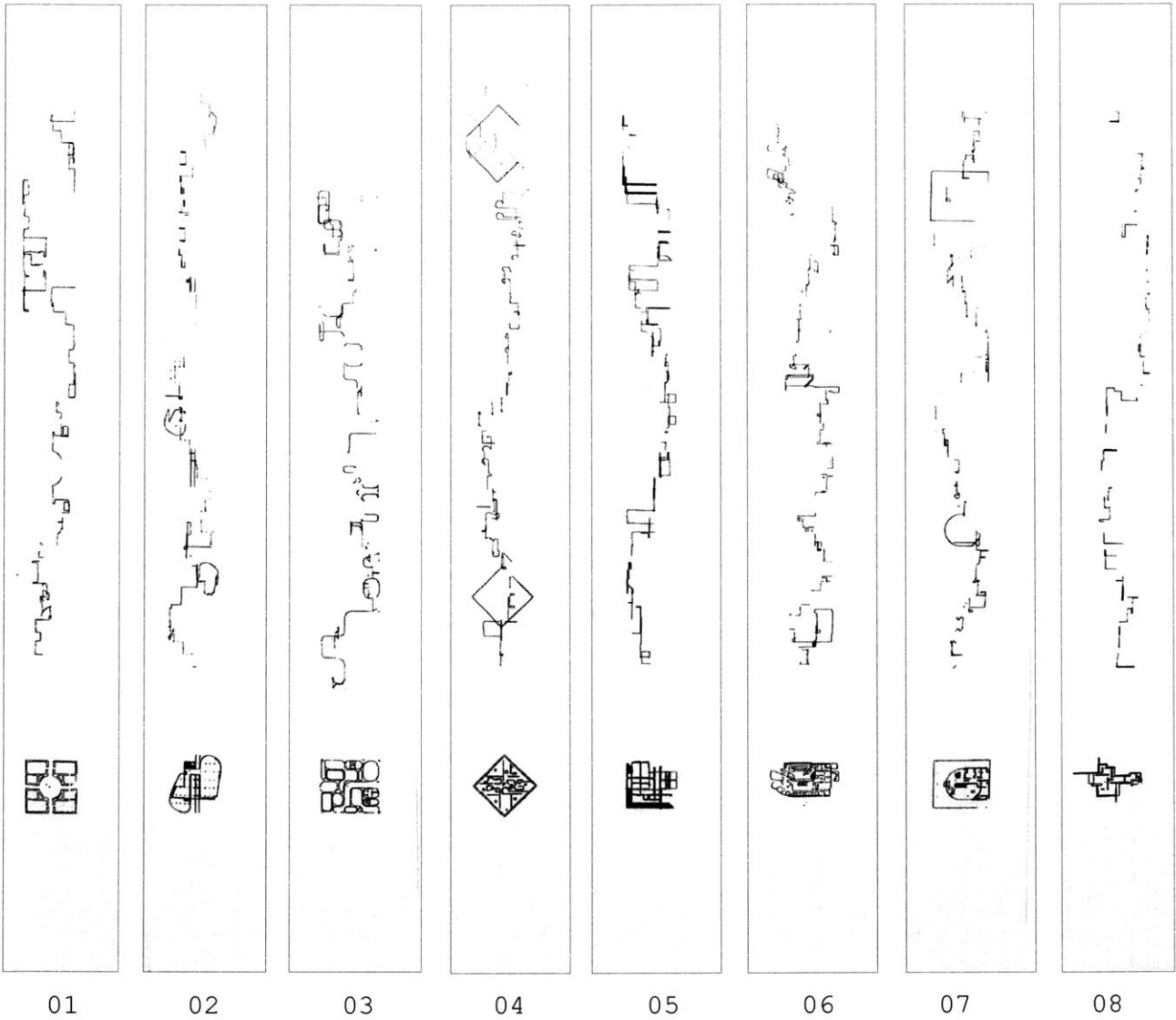
Photo: Sarah Wagner

II. PLOTTING: DRAWING SERIES No.2

The "time-lapse" drawing process invites rereading of iconic images by seeing them "through the eyes" of the traveling salesman solver algorithm. This series takes a selection of line drawings, which are shown in their original form at the base of each print, and makes a time-lapse drawing above each. The goal in this series is not to illustrate the time-lapsing process itself but instead to employ it as a technique to compare similar drawings sequenced by the same algorithm.

In this series, familiar buildings are transformed through the time-lapse process to invite novel readings: Some transformations appear to accentuate the original parties, while others scuttle it; Symmetrical plans are pulled apart in uneven clusters; Drawing elements which represent construction components become abstracted and compositional; Sinuous curves are fragmented into infelicitous shards; Modernist free plans are set loose to swim freely on the page and new patterns emerge from lines which were not originally in dialog.

This drawing series was completed for my M.Arch thesis at the University of Toronto in May 2015.



01

02

03

04

05

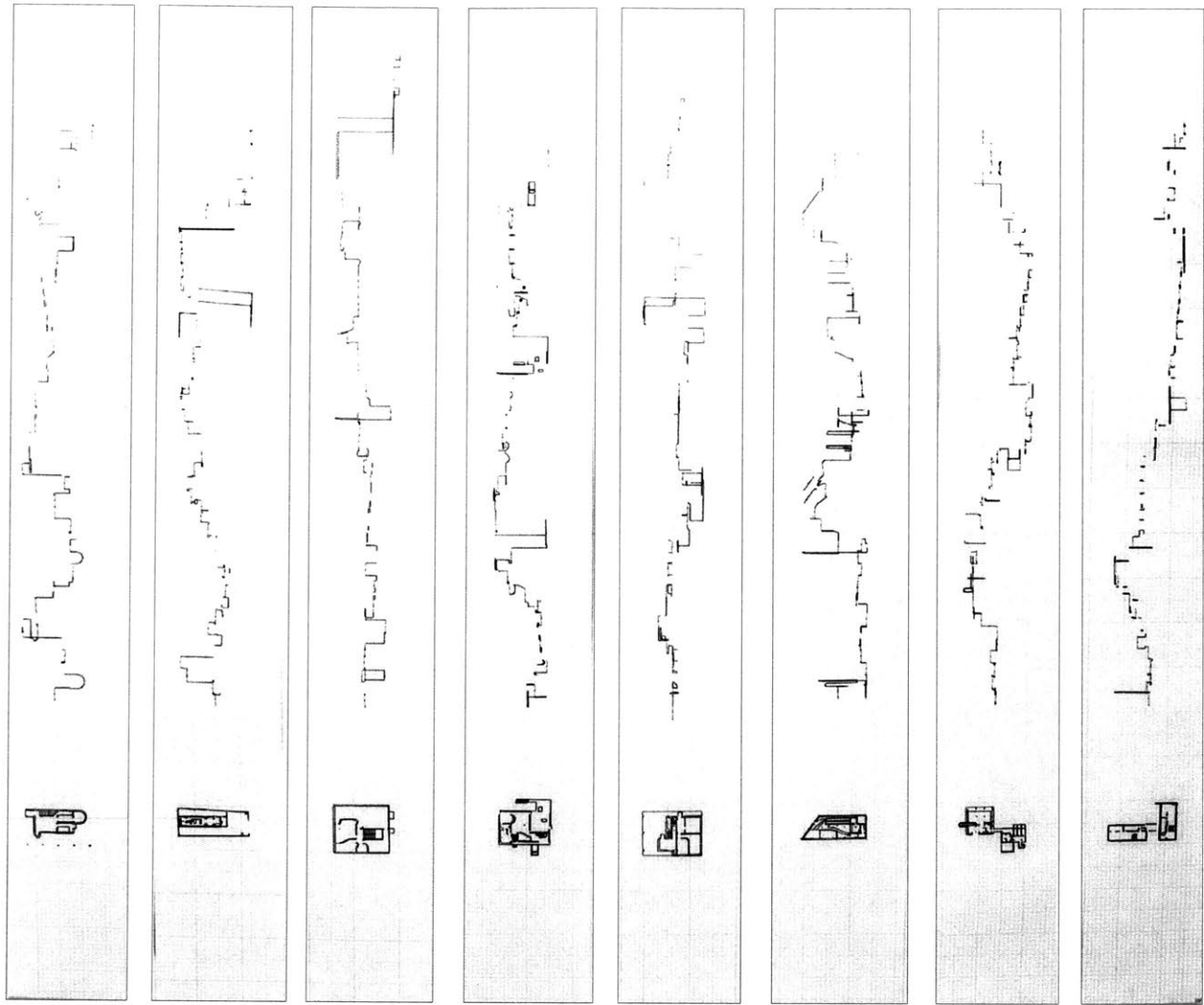
06

07

08

- 01 Villa Rotunda
- 02 Carpenter Center
- 03 Glass Blowing Museum
- 04 Diamond House A
- 05 Frank House
- 06 Wolfsburg CC
- 07 Villa Savoye
- 08 Brick House

2A: Iconic Plans



01

02

03

04

05

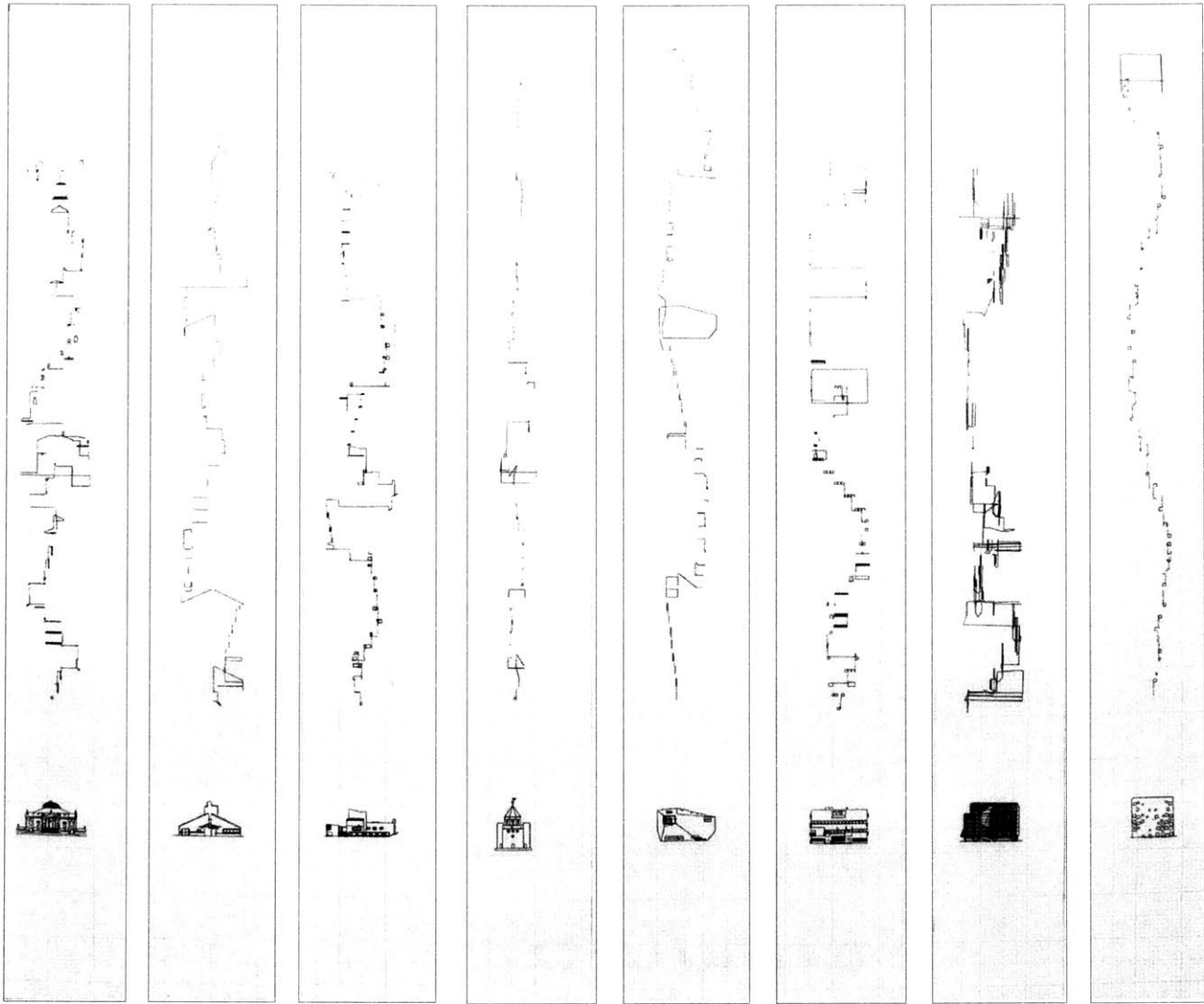
06

07

08

- 01 1905
- 02 1920
- 03 1932
- 04 1932
- 05 1940
- 06 1943
- 07 1949
- 08 1955

2B: Chronological Le Corbusier Plans



01

02

03

04

05

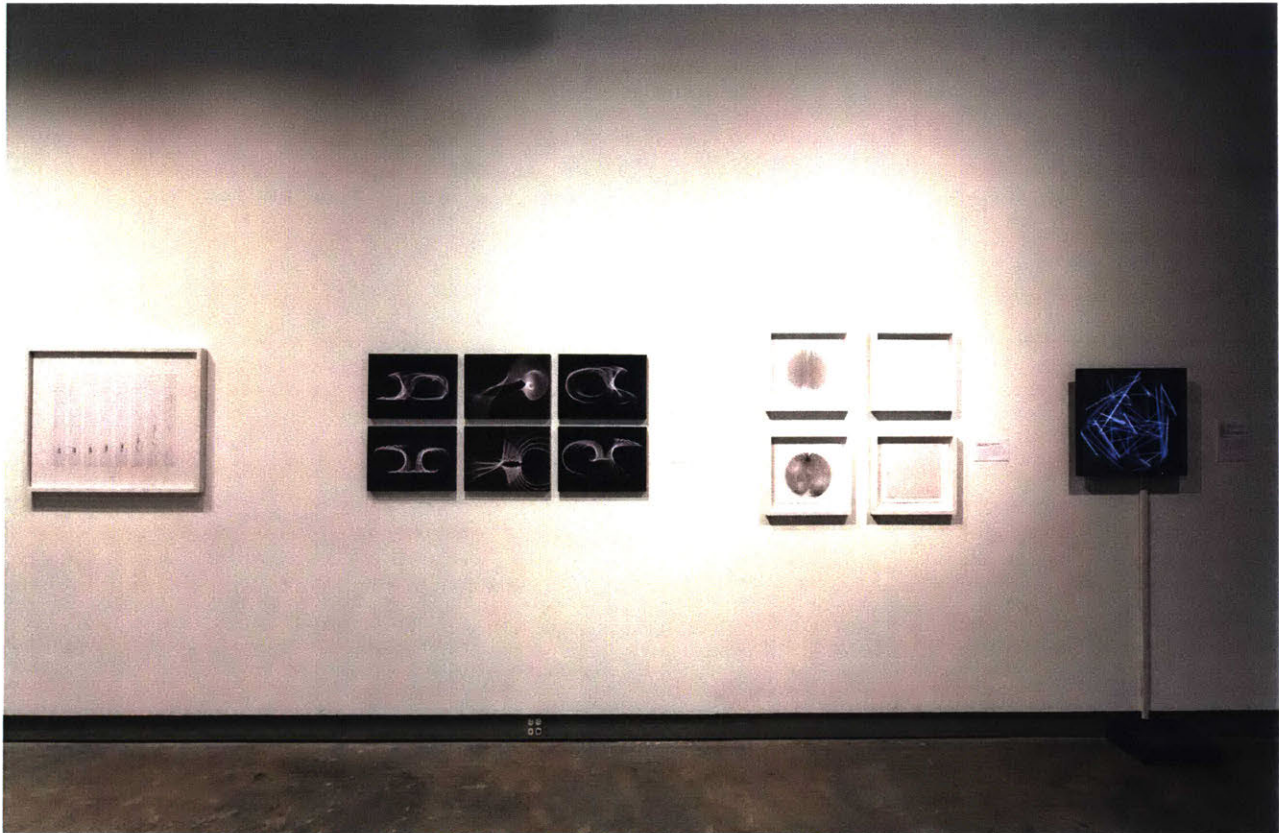
06

07

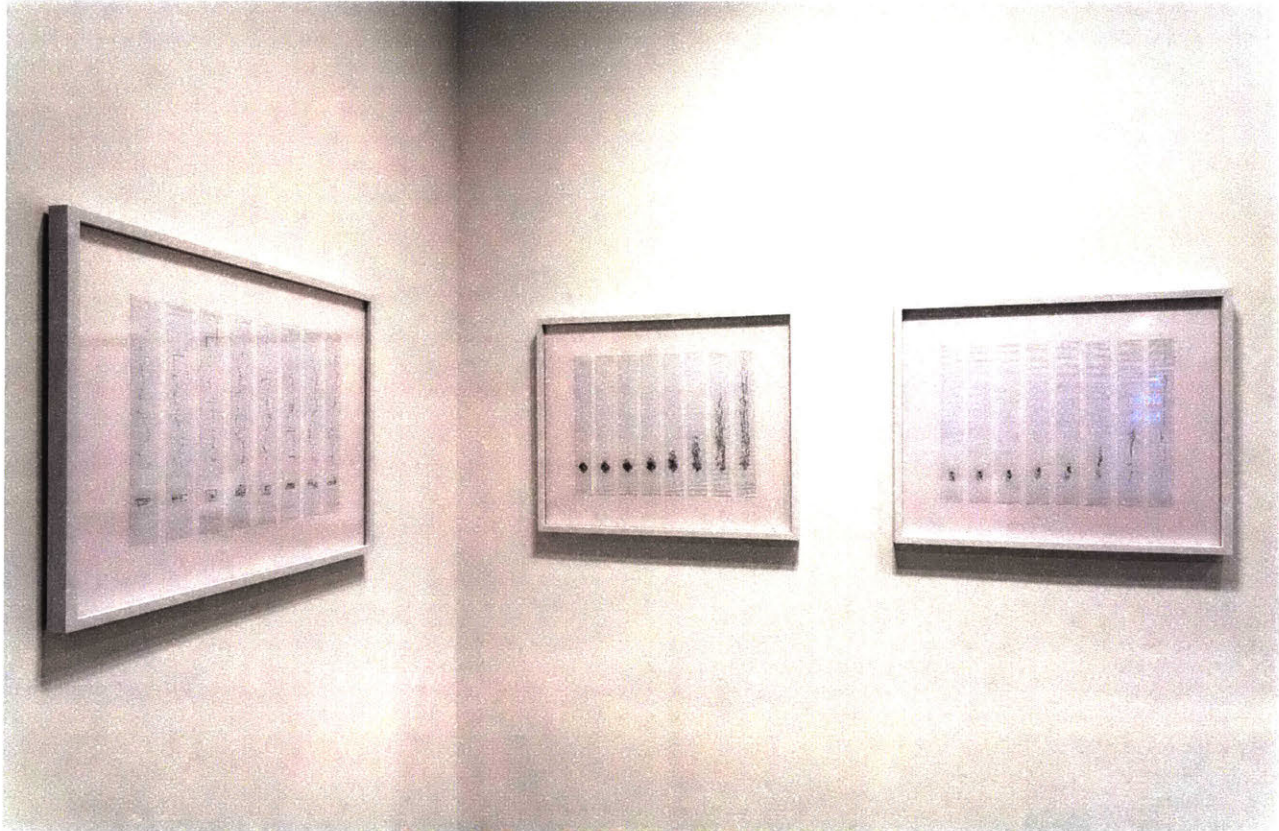
08

- 01 Villa Rotunda
- 02 Venturi House
- 03 Säynätsalo Town Hall
- 04 Teatro del Mundo
- 05 Case de Musica
- 06 Villa Stein
- 07 Cooper Union
- 08 Zollverein School

2C: Iconic Elevations



CMU's SOA DCI-IDC Exhibit September 2017. Photos: Tom Little



CMU's SOA DCI-DCI Exhibit September 2017. Photos: Smokey Dyar



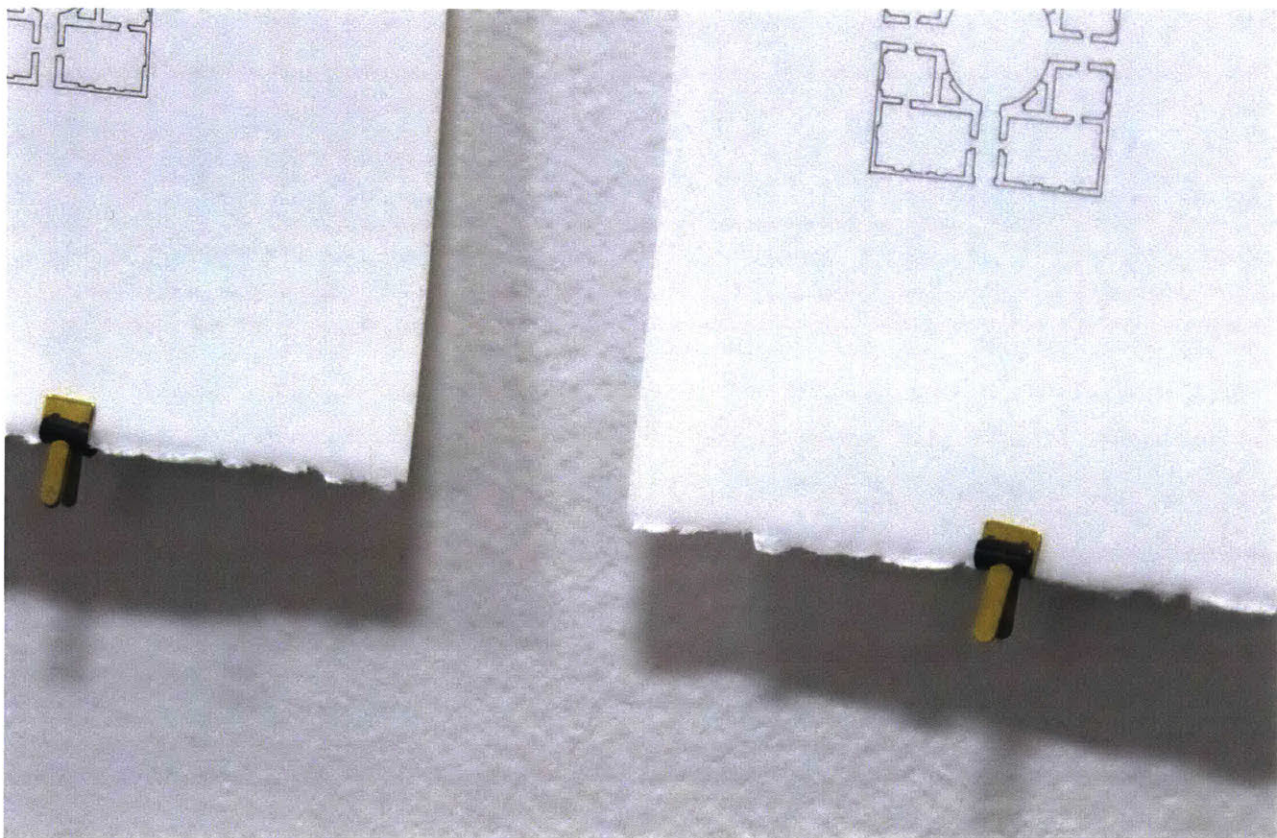
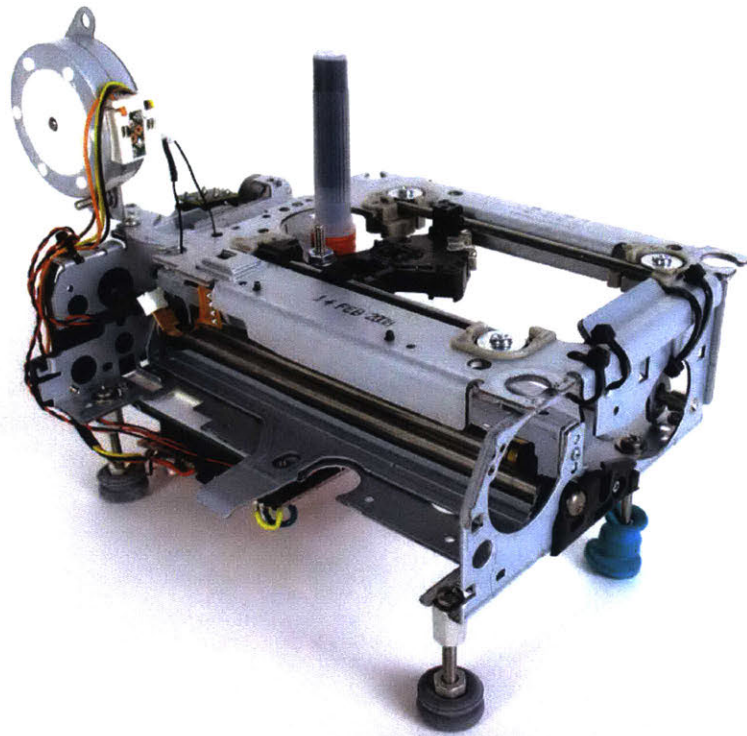
CMU's SOA DCI-IDC Exhibit September 2017. Photos: Smokey Dyar

II. PLOTTING: DRAWING SERIES No.3

The "time-lapse" drawings in this series focus on the parameters of the CNC plotting process itself. This series compares different CNC/CAM drawing settings of the same image, such as different starting points, different tool settings, and different orientations of the input drawing.

In order to conduct these experiments, a second prepared plotter was built to work with thicker paper stock, finer Rotring Rapidograph technical pens and higher resolution movement. The machine is more stable and reliable to work with and produces more controlled drawings. A new technique of hanging the drawings to the wall used miniature bulldog clips affixed to pins.

Plotter No.2 and Drawing Series No.3 were produced for two exhibits: At Carnegie Mellon's Miller Gallery, *Computational Design: Practices, Histories and Infrastructures Symposium* in September-October 2017, curated by Daniel Cardoso Llach, and MIT's Keller Gallery for a solo exhibit entitled *Exploring the Mechanics of Representation* in February 2018, curated by Irina Chernyakova.



Prepared Plotter No.2 (top), fastening detail (bottom).

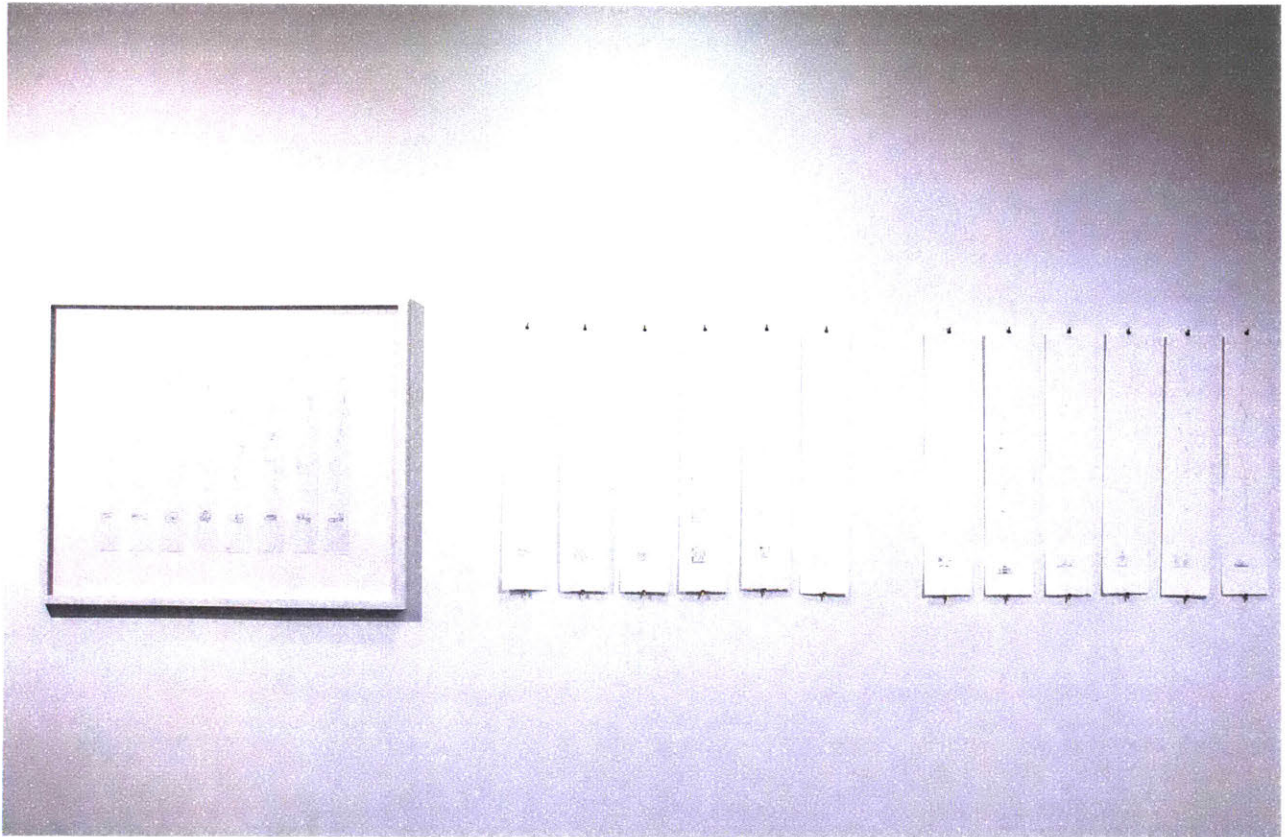


Exhibit at MIT's Keller Gallery curated by Irina Chernyakova.
Photos by Sarah Wagner.

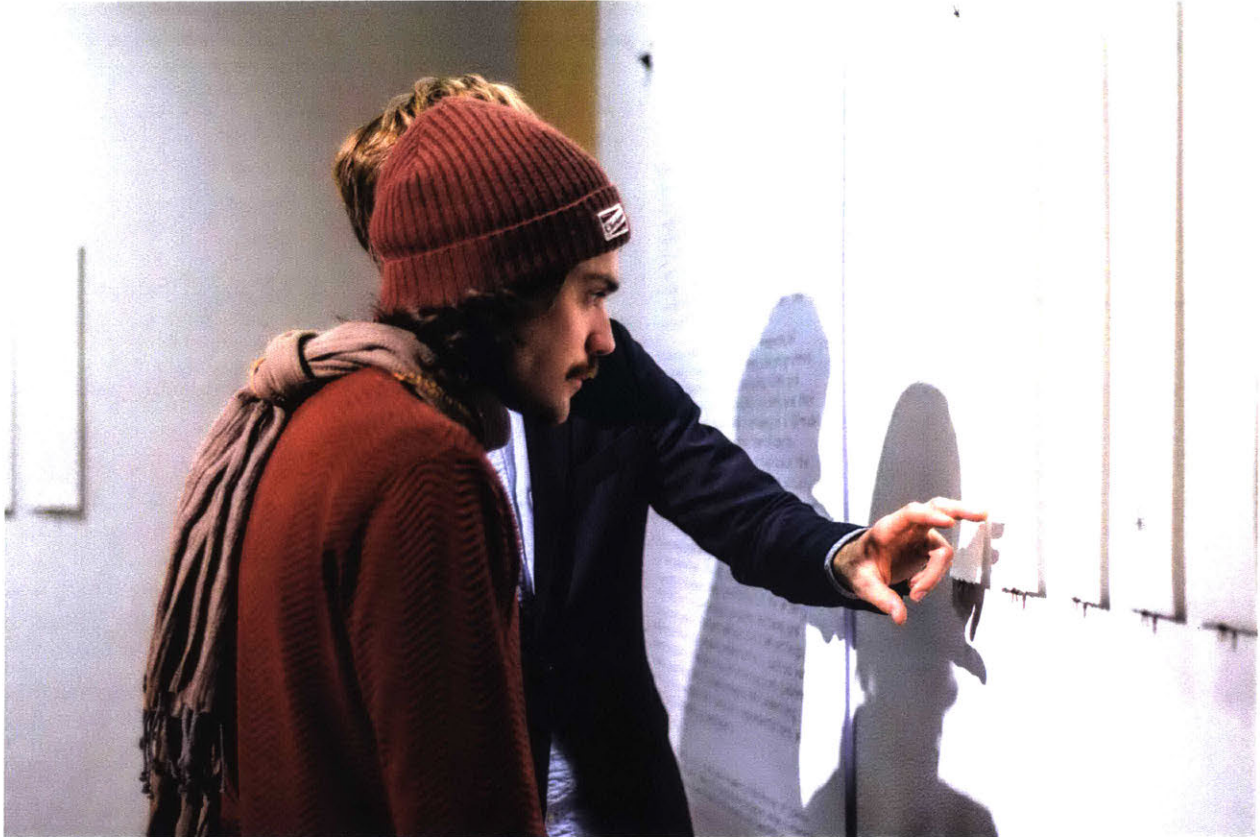


Exhibit at MIT's Keller Gallery curated by Irina Chernyakova.
Photos by Sarah Wagner.

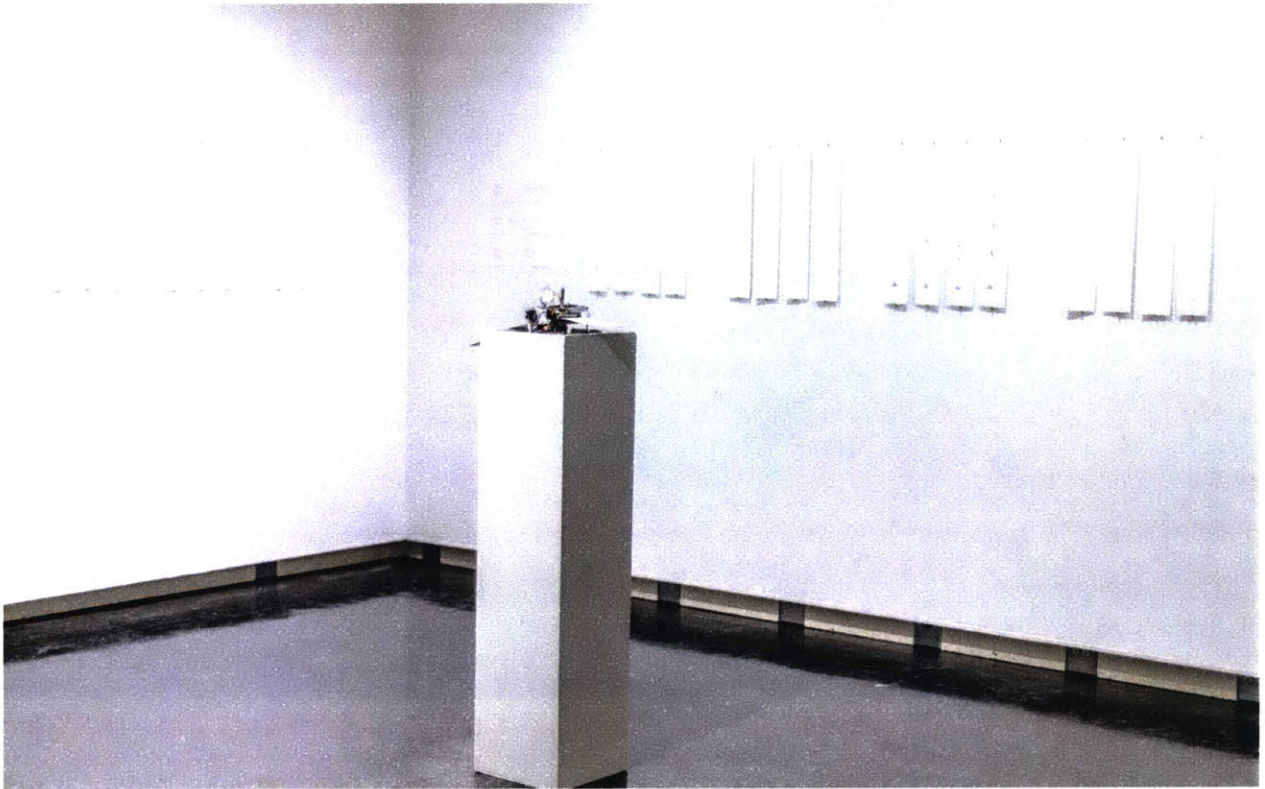
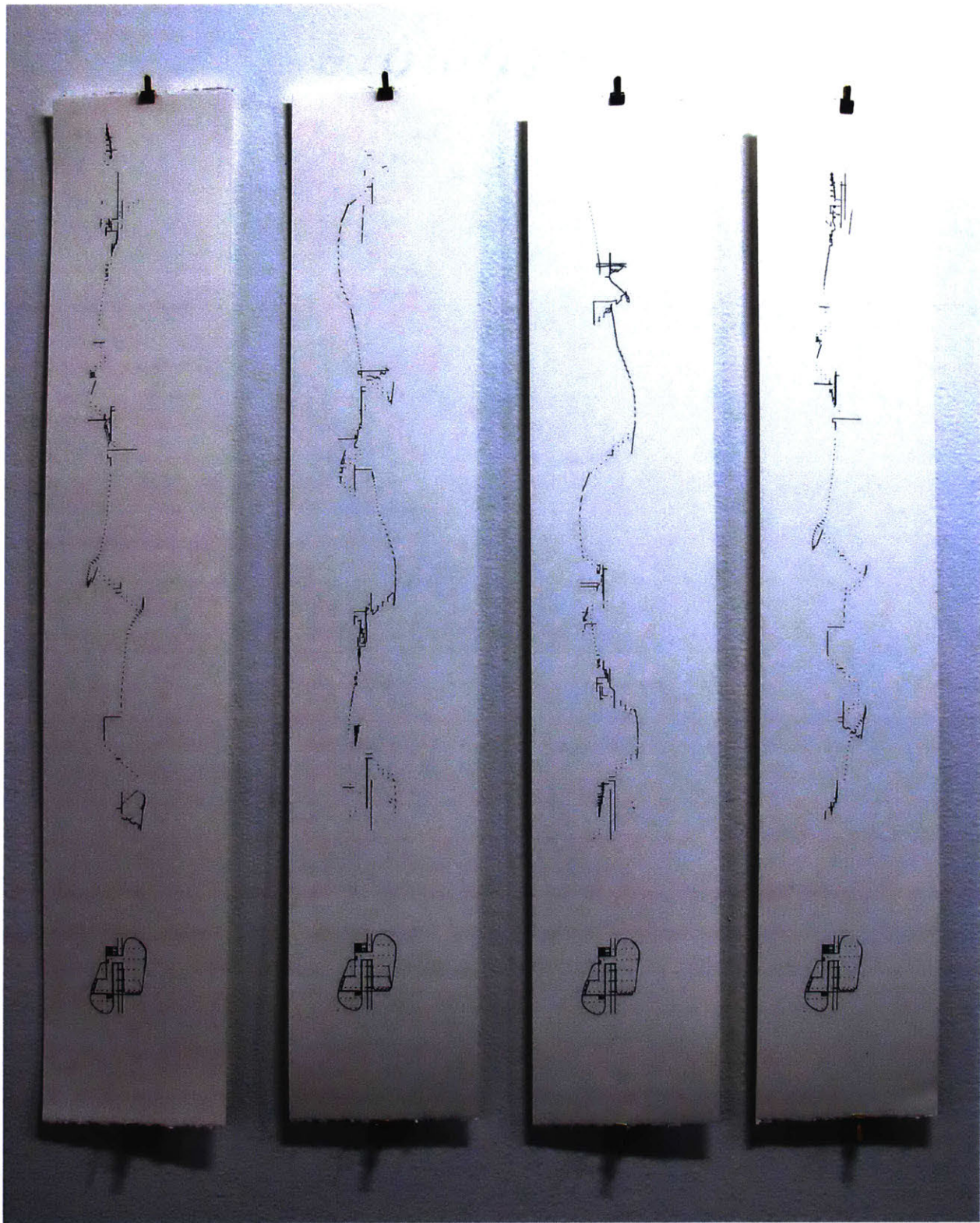
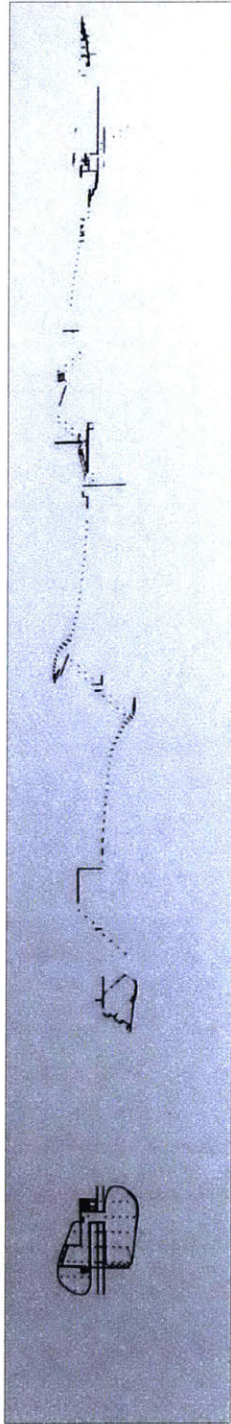


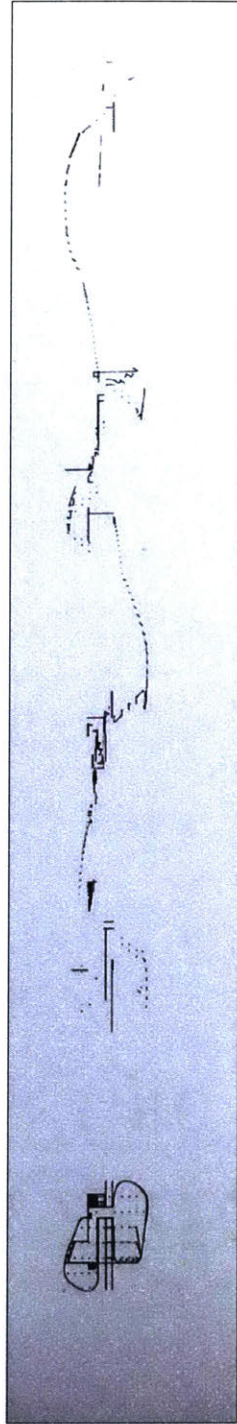
Exhibit at MIT's Keller Gallery curated by Irina Chernyakova.
Photos by Sarah Wagner.



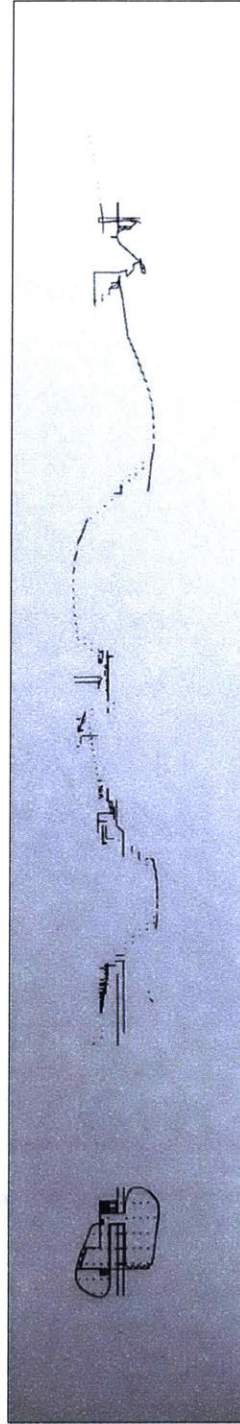
3A: Carpenter Center Plan



01



02



03



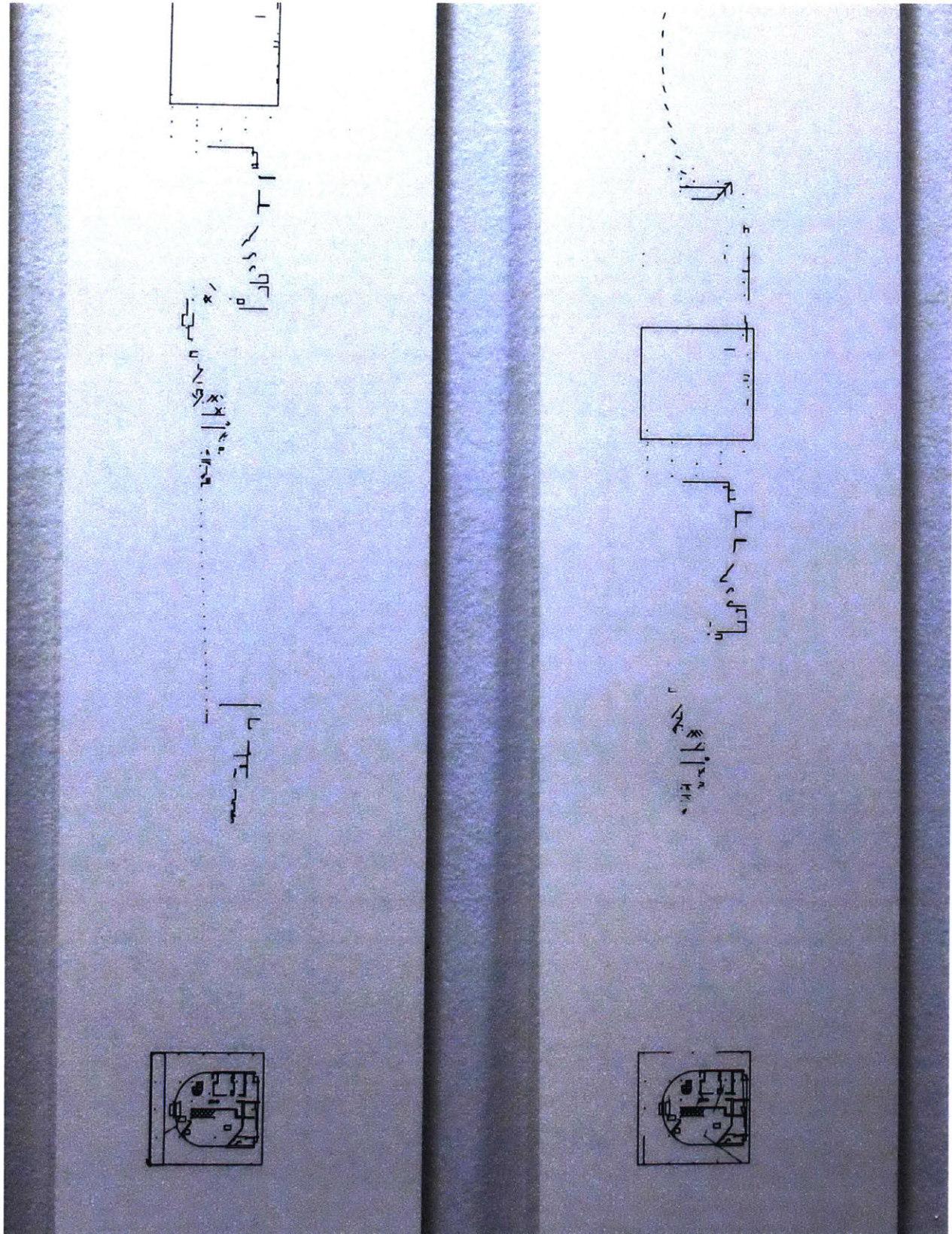
04

- 01 SW Start Point
- 02 NW Start Point
- 03 NE Start Point
- 04 NW Start Point

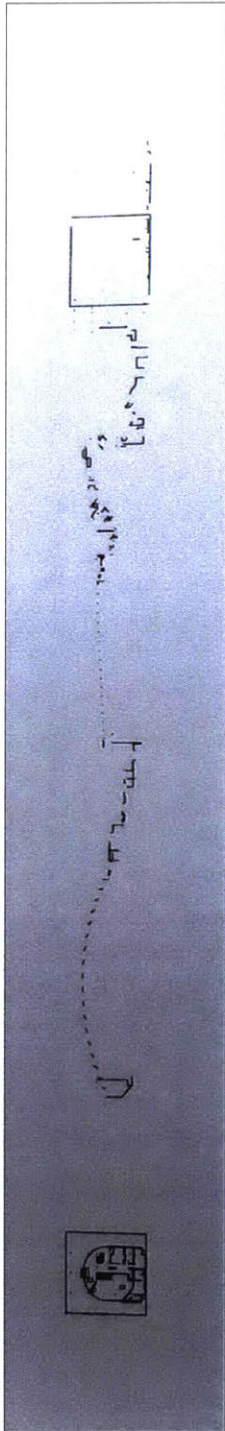
3A: Carpenter Center Plan



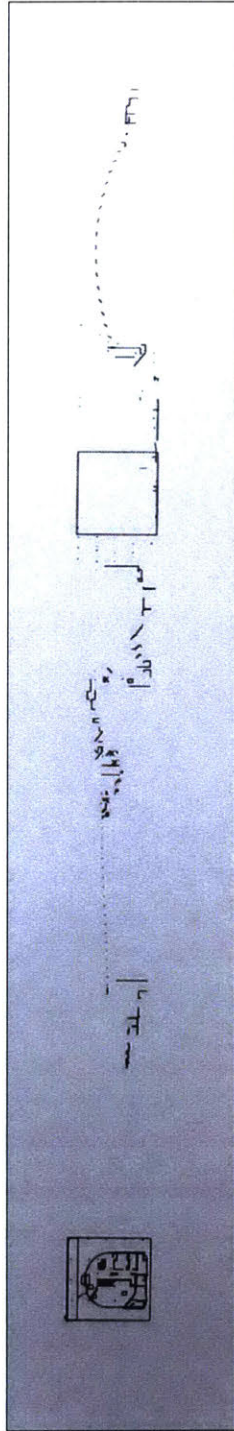
3B: Villa Savoye Plan



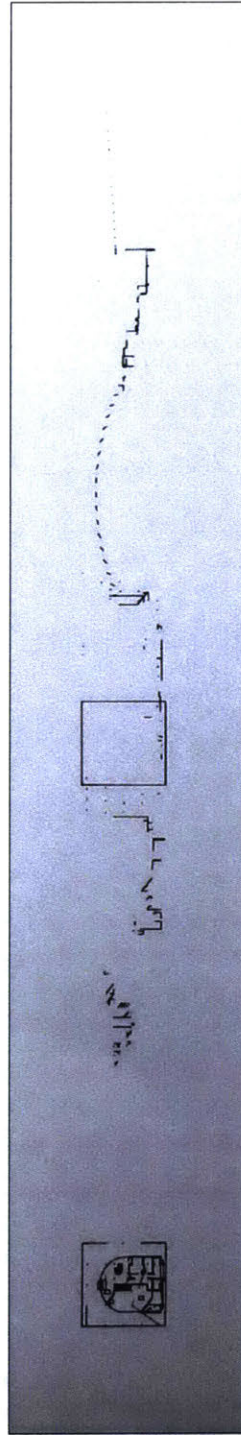
3B: Villa Savoye Plan Detail



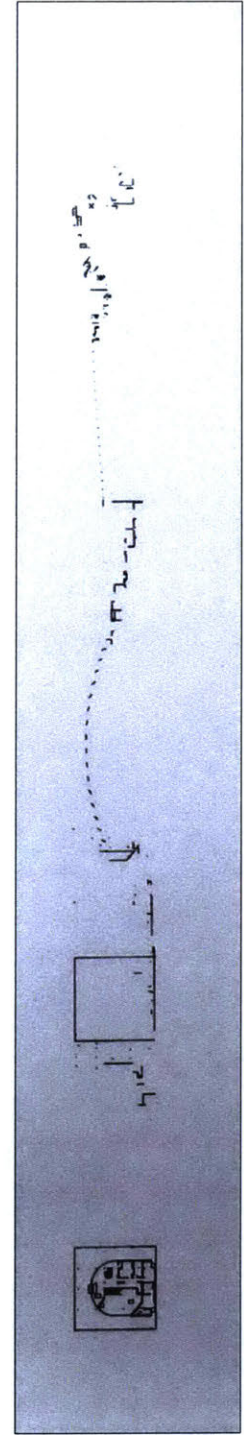
01



02



03



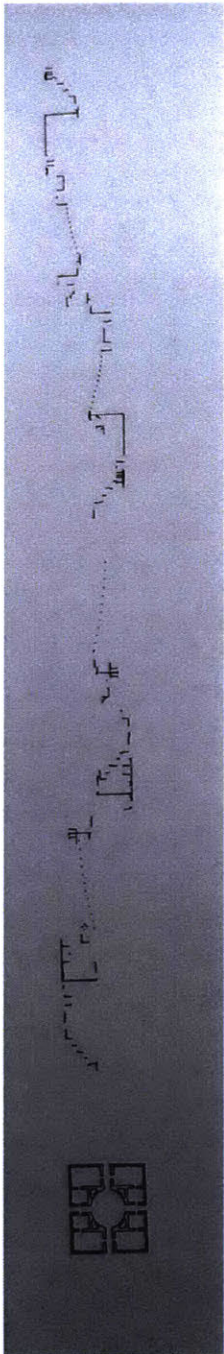
04

3B: Villa Savoye Plan

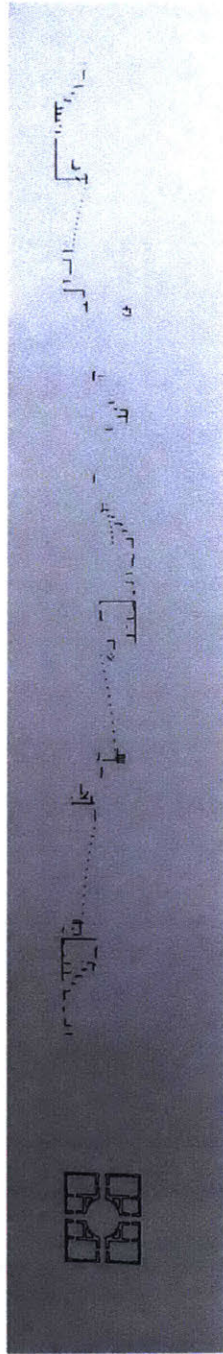
- 01 Original Index
- 02 Index Shifted by 1/4
- 03 Index Shifted by 2/4
- 04 Index Shifted by 3/4



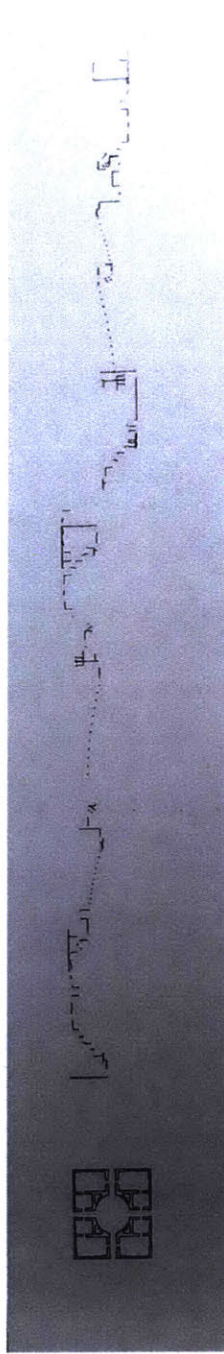
3C: Villa Rotunda Plan



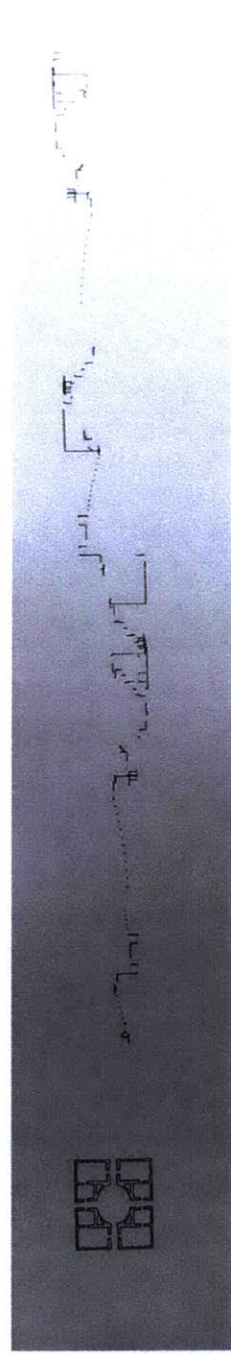
01



02



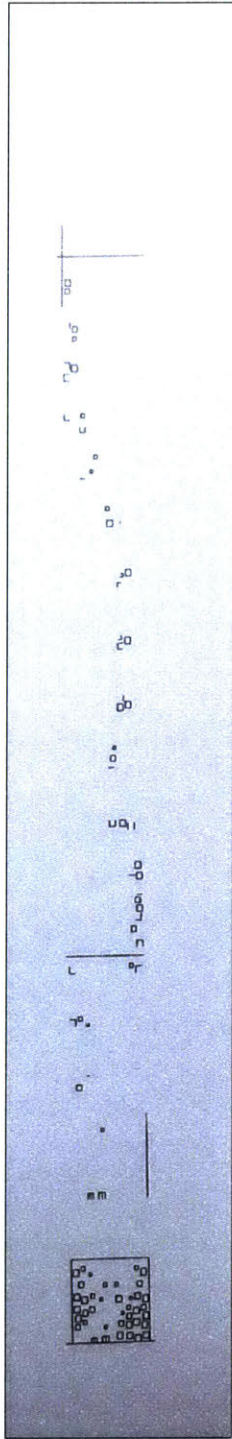
03



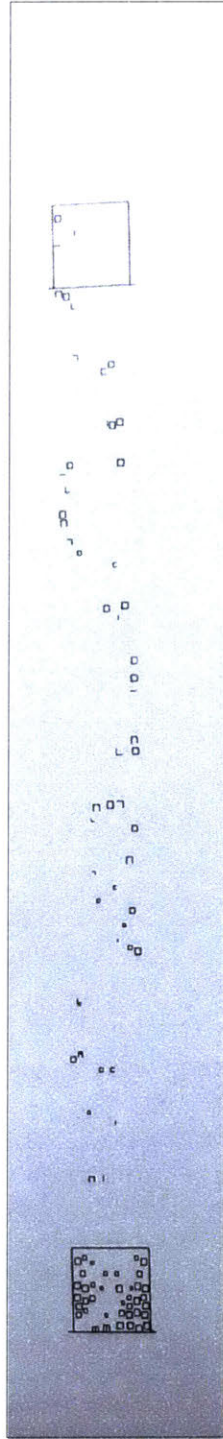
04

01 FlatCAM
02 RhinoCAM
03 Inkscape
04 CAMBAM

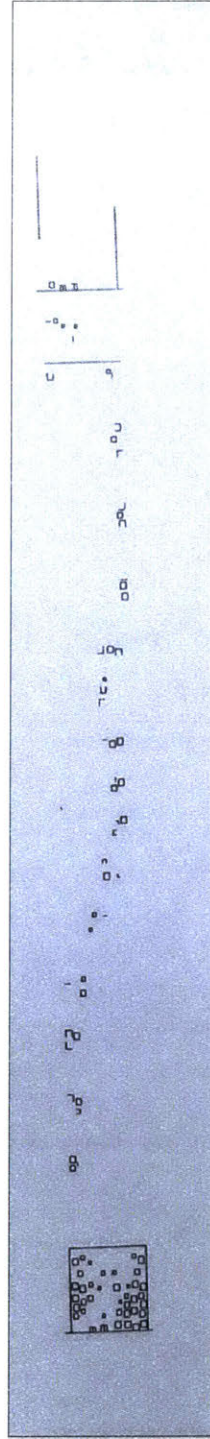
3C: Villa Rotunda Plan



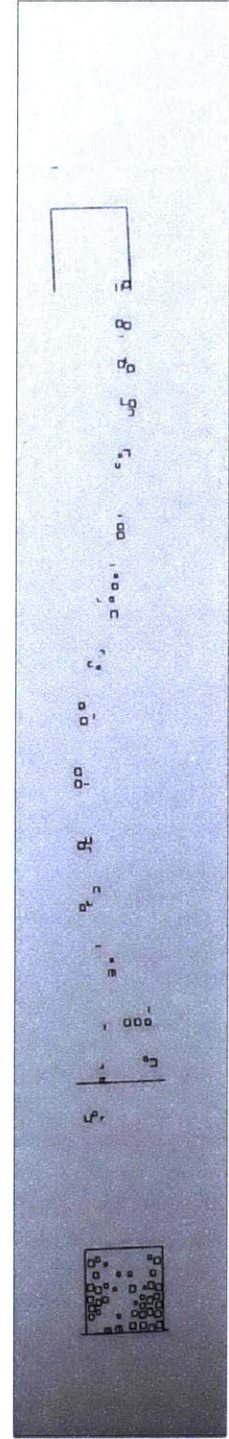
01



02



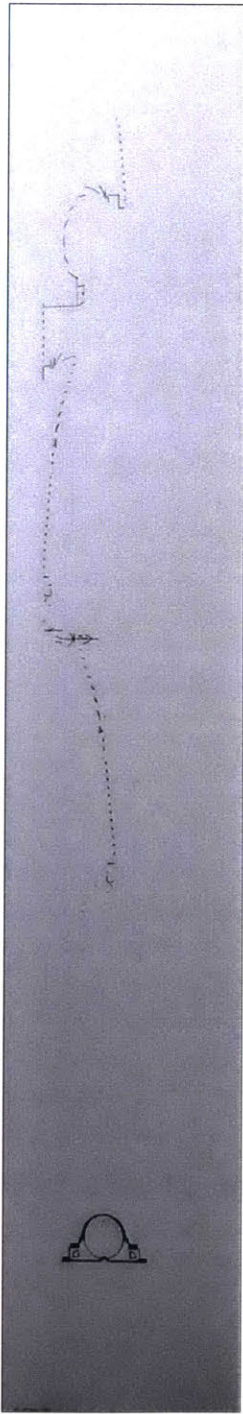
03



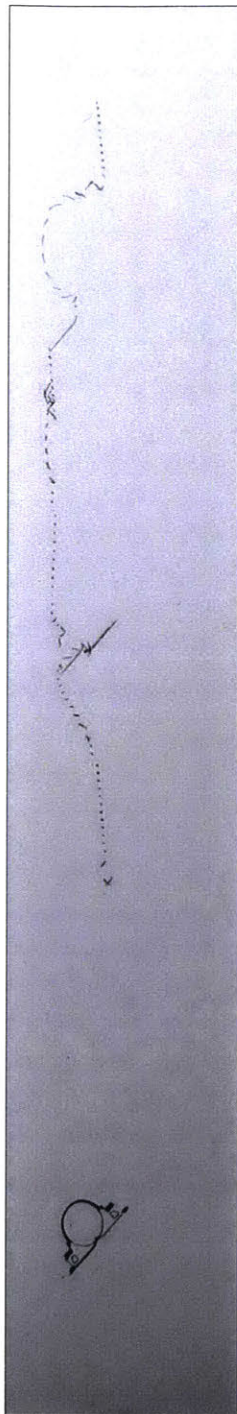
04

3D: SANAA Elevation

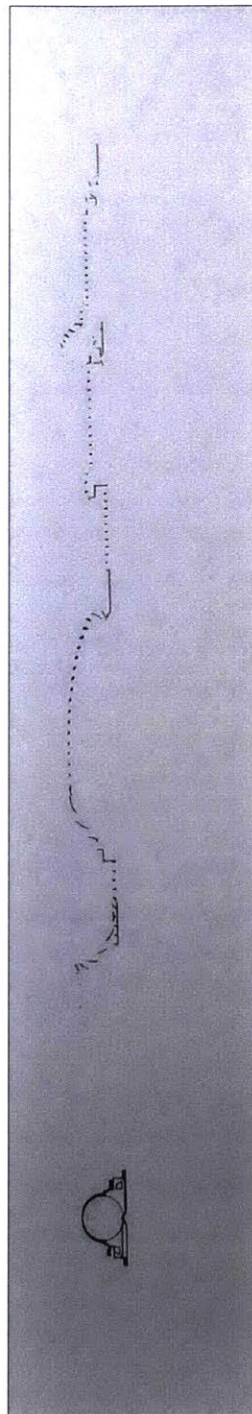
- 01 Counterclockwise milling
- 02 Engrave milling
- 03 Inside milling
- 04 Default milling settings



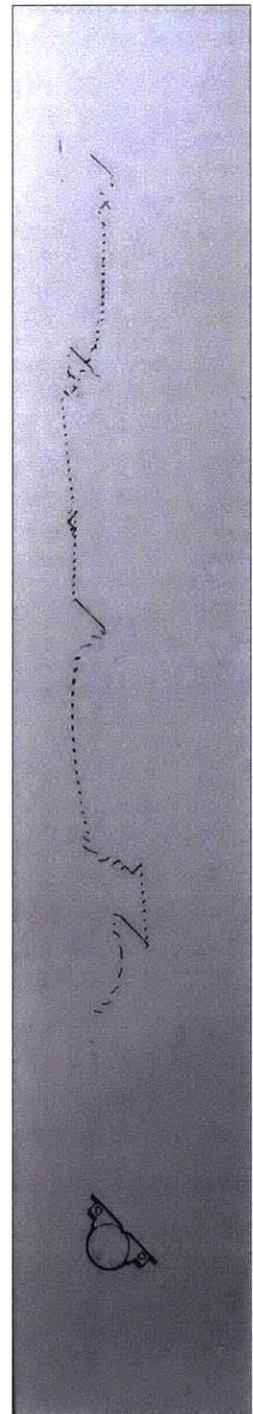
01



02



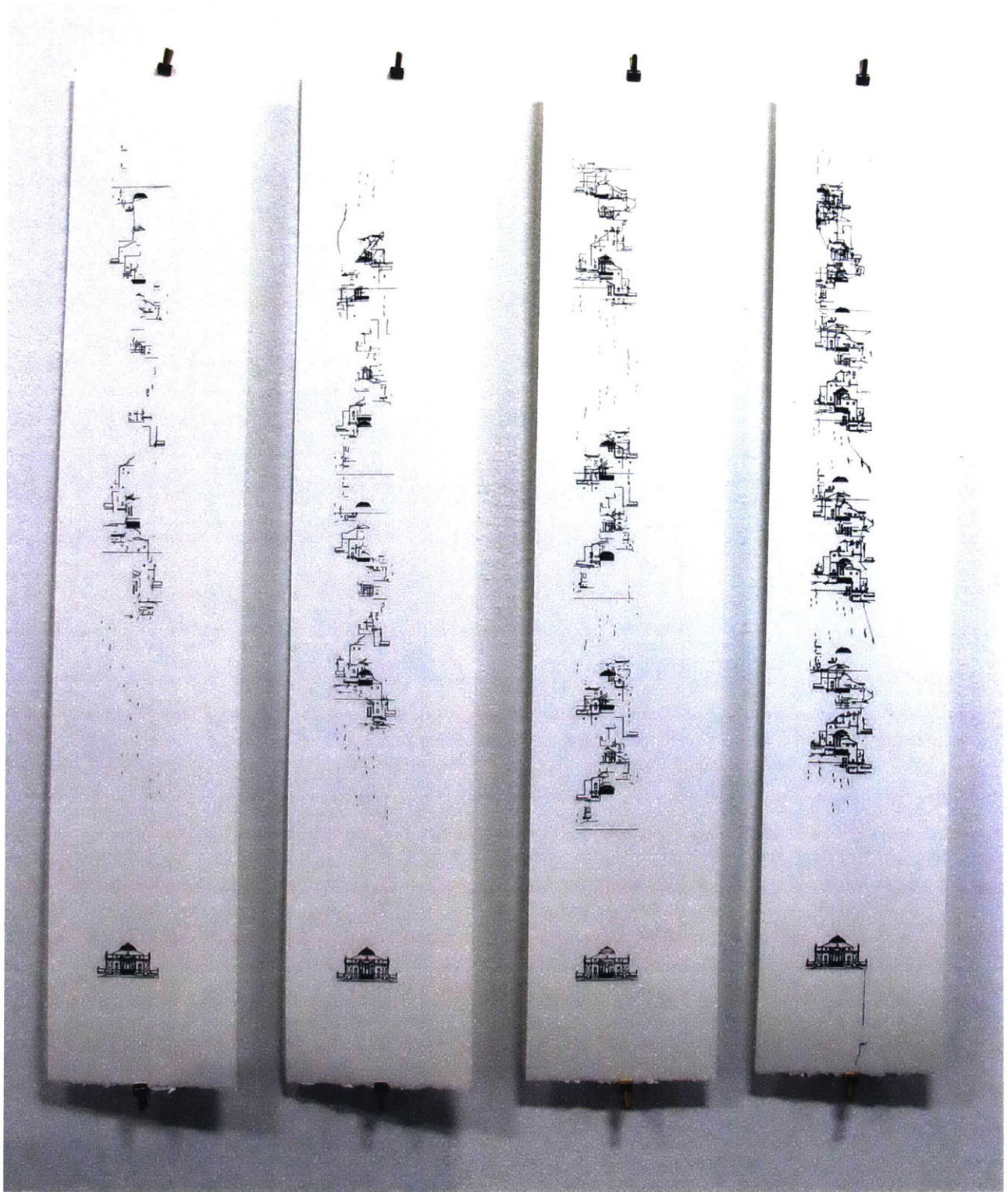
03



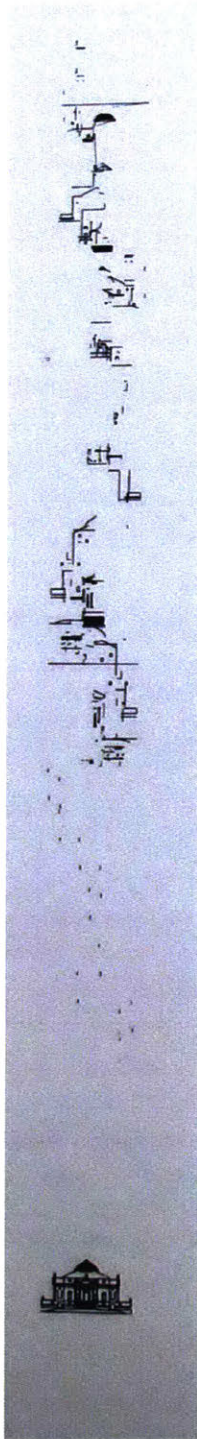
04

- 01 0 degrees
- 02 45 degrees
- 03 90 degrees
- 04 135 degrees

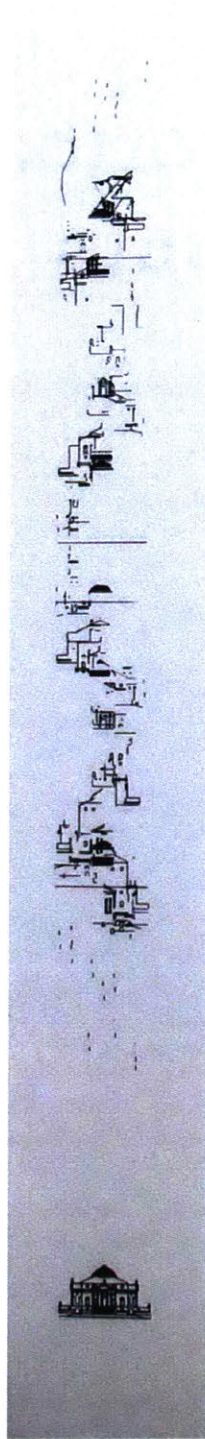
3E: Boullée Section



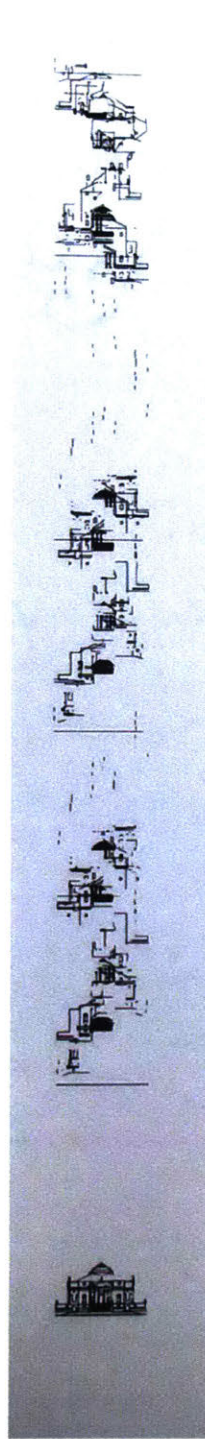
3F: Villa Rotunda Section



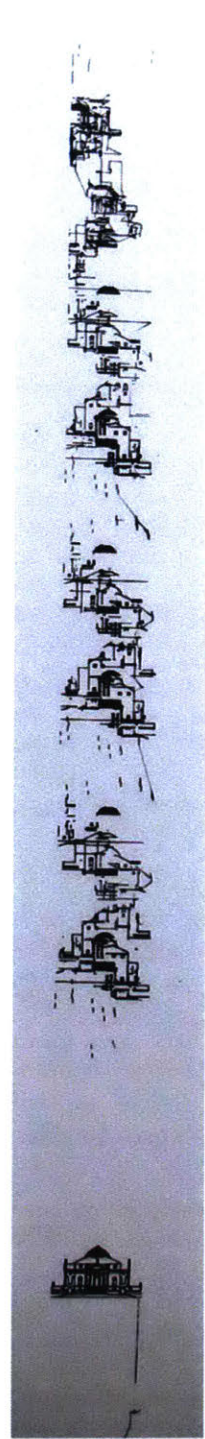
01



02



03



04

3F: Villa Rotunda Section

01 Single Copy
02 Double Copy
03 Triple Copy
04 Quadruple Copy

II. PLOTTING: CONCLUSIONS

Revealing time-lapse, or "exploded" drawings, of input vector images provides a medium for designers to observe and record the transformations that take place within a plotter CAM/CNC tool-chain. Translating input images into the time-lapse notation invites: i.) rereading of the original input drawings, ii.) comparative analysis of similarly transformed input drawings, and iii.) comparative analysis of differently sequenced vector images.

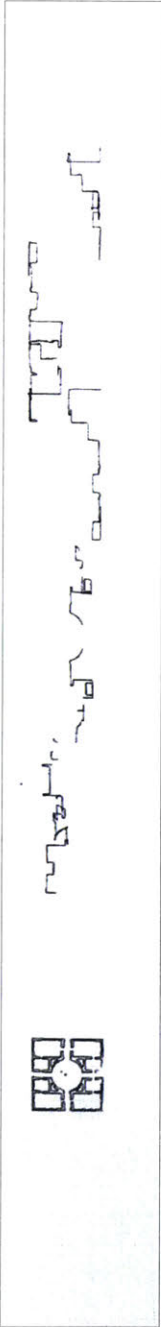
i.) Rereading

When input drawings are translated into time-lapse recordings, the formal composition of the input image is altered. The formal logic of some iconic images appears to be preserved during the transformation process while others (often more curvilinear vectors) became estranged from their original formal composition. In both cases the time-lapse images allow us to see a familiar image in a new light "through the eyes" of a computer sequencing algorithm.

ii.) Comparative analysis

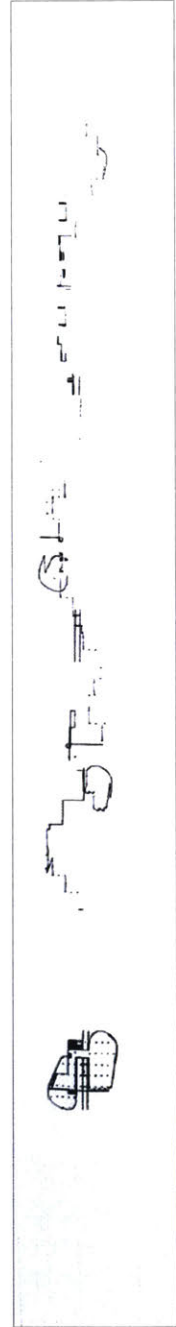
Sets of input images (square plans, work by the same architect, etc.) can be converted into time-lapse images and compared side by side. In this way, one can explore the possibility of a series of Le Corbusier plans sharing certain characteristics that were retained through the transformation process, for example. Often these comparisons reveal characteristics about the transformation process itself rather than the input images. For instance, drawings which are denser exhibit far more intricate "explosions" while minimalist representations create at times elegant signatures.

II. PLOTTING: CONCLUSIONS

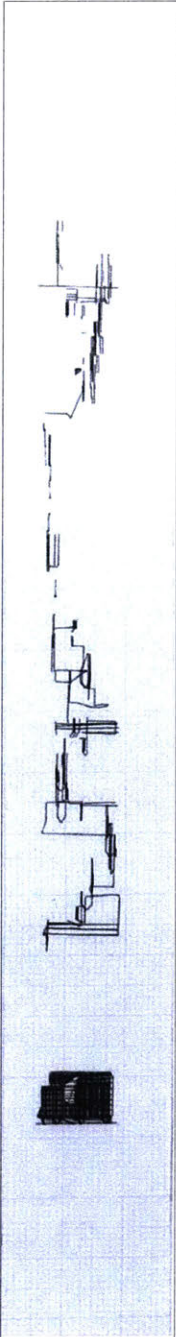


◀ The Palladian Villa Rotunda First Floor Plan is symmetrical in X and Y axes. The shortest path algorithm doesn't appear to exploit the image's symmetry in its optimization process, however, and instead produces a sequence of pen movements which appear haphazard and aleatory to the human eye. The contrast between the input and output is that between a work of classic architecture based on order, and a contemporary algorithmic re-reading producing a seemingly chaotic sequencing of drawing elements.

The different construction elements of Le Corbusier's Carpenter Plan (columns, ramp, windows, etc.) are re-read as graphical and compositional (dots, lines, oddly elliptical forms) by the shortest path finder and the time-lapse representation technique. The resultant composition contains new relationships between drawing components, making emergent forms when parts of the original drawing find themselves juxtaposed and overlapped with new neighbors.

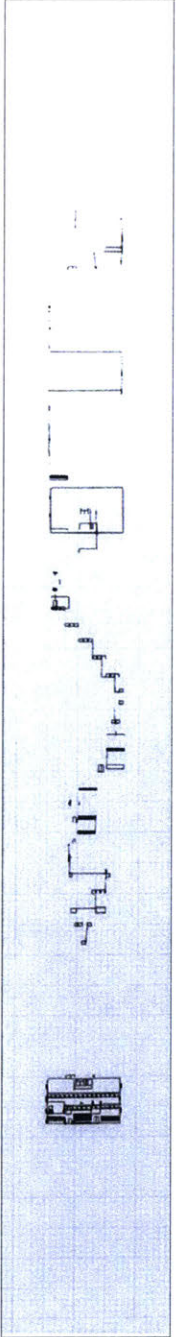


II. PLOTTING: CONCLUSIONS

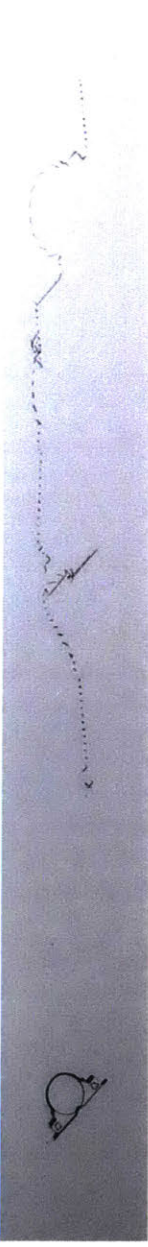


◀ Morphosis is a well known L.A.-based firm which explores striking forms in industrial materials. Because all the lines are equally weighted in the eyes of the optimizer, the grill pattern on the facade of the Cooper Union elevation drawing becomes the most graphically dense element in the time-lapse drawing. Despite the regularity of the grid pattern, the output drawing is mysteriously clustered, like spaghetti falling off of a plate. This time-lapse optimized version contrasts the measured tightness of the original composition of the drawing.

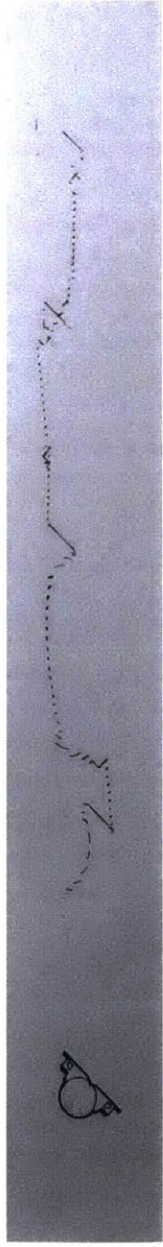
The Villa Stein elevation was the subject of much formal analysis by the likes of Colin Rowe and Peter Eisenman. The time-lapse of the optimized drawing is reminiscent of text in places and almost resembles a work of concrete poetry or the indentation of a piece of Python code. The curvilinear, spiraling path taken by the optimizer was a common feature across many optimized drawings.



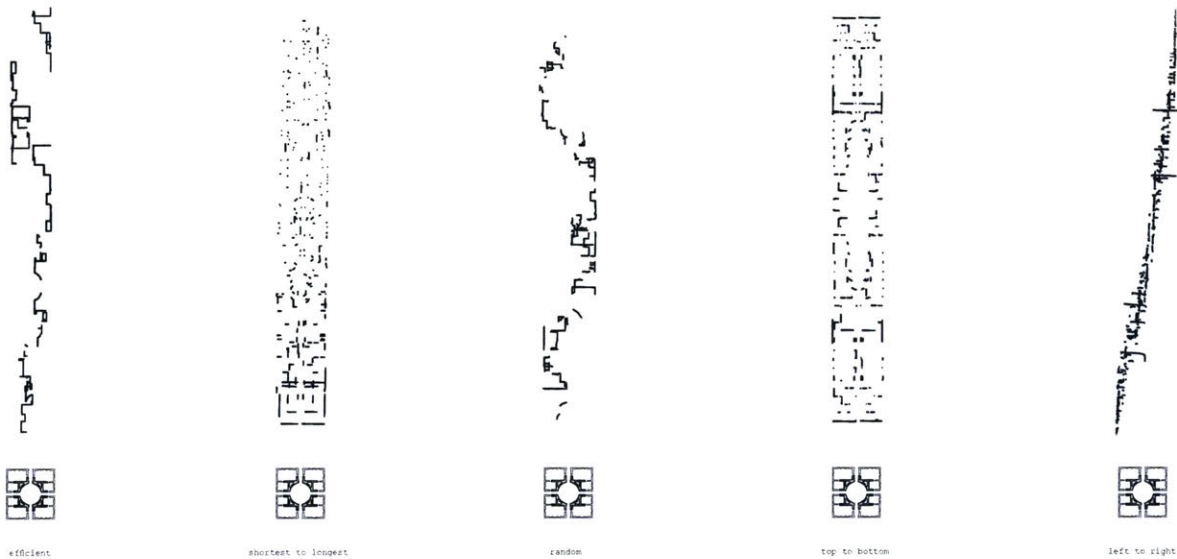
II. PLOTTING: CONCLUSIONS



◀ This series compares the same optimizer sequencing the same image at different degrees of rotation. To the human eye, each of the input drawings are the same, simply rotated. To the optimizer, indifferent to content of the drawing beyond the coordinates of the lines, each input drawing calls for a completely different sequence of line movements. The series calls attention to the importance of image "polarity" in the plotting process and highlights a difference in the ways in which machines and humans can interpret images.



II. PLOTTING: CONCLUSIONS



Palladian Villa sequenced by different algorithms

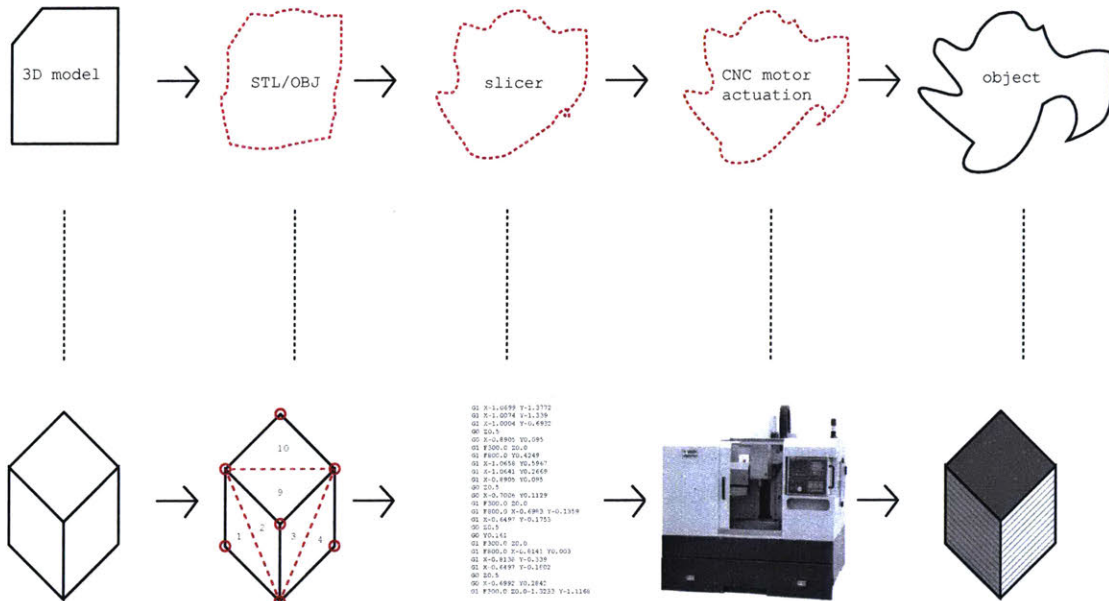
iii.) Custom Tool Path Sequencers

By creating custom vector sequencing algorithms in Grasshopper and passing the resultant G-code output to the time-lapse drawing machine, it is possible to compare different ways of drawing the same input image stretched in time. The above series of time-lapse drawings reveals graphic representations of different algorithmic sequencing priorities. It provides a means of representing a complex process in a kind of visual shorthand. This means of visually describing the priorities of an algorithm touch on Johana Drucker's notion of graphesis, or the graphic expression of a form of knowledge.

In the following section, Meshing, the strategy of placing different algorithmic sequences of the same input will be explored in three-dimensional space.

III. MESHING

III. MESHING: BACKGROUND



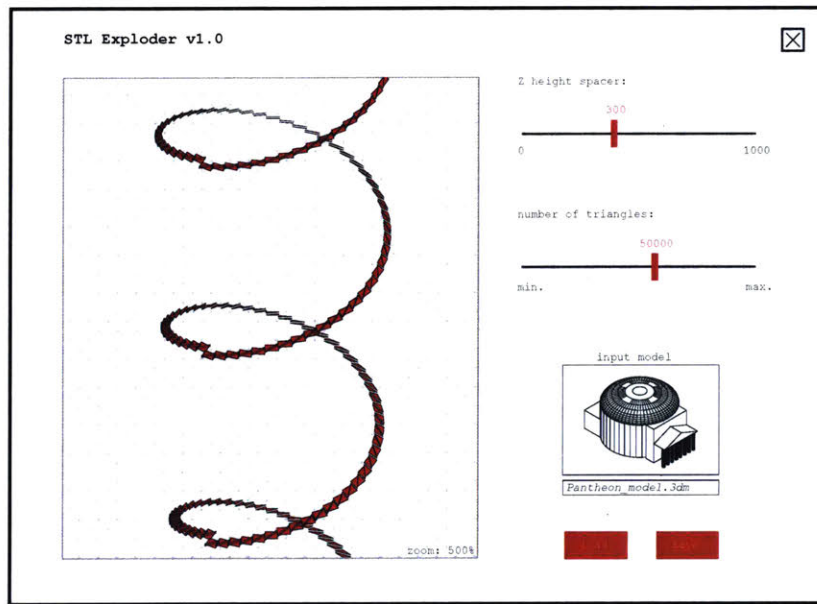
Meshing Toolchain

What happens in the moments after we click print and transform a 3D model on our screen into input for a 3D printer? This second foray attempts to materialize the processes undertaken by meshing algorithms, which translate our models into triangles so they can be 3D printed.

In order for a 3D object to be converted into a series of layers ("sliced") for a 3D printer to extrude, it first needs to be translated into triangles ("meshed").¹ Meshes are the most fundamental way to describe 3D form inside the computer and are also used for rendering, simulation and transferring files between different modeling programs. By exploring the ways in which meshes are created, we can reveal a variety of different meshing strategies and occasionally learn something about the original input model before it was meshed.

¹ "Stereolithography Interface Specification," 3D Systems, Inc., October 1989.

III. MESHING: PREPARED MESHER



STL Exploder v1.0

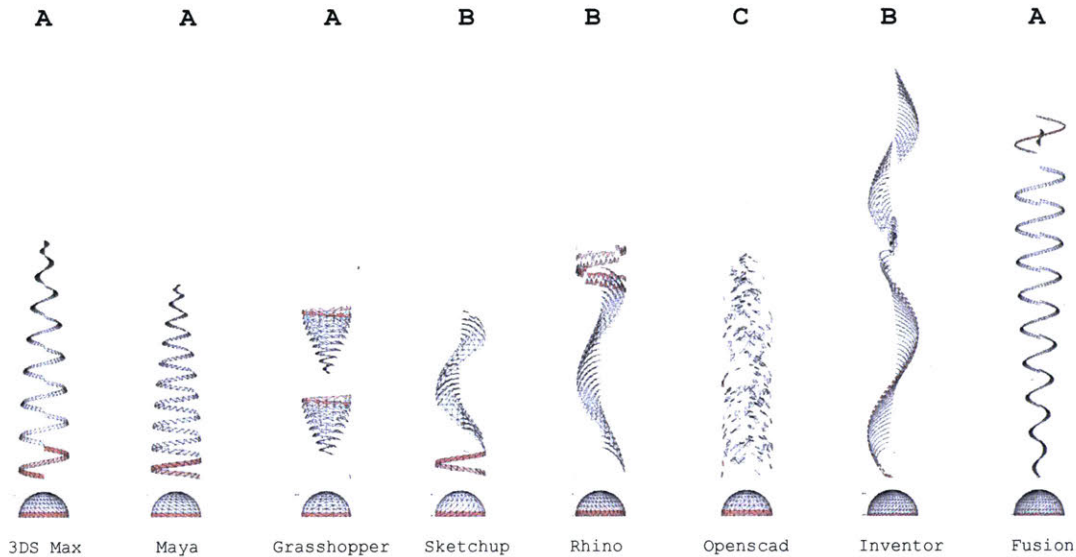
Algorithms which generate STL files and convert curved surfaces like NURBS into polygonal meshes follow these steps:

1. Place points on all of the shared edges of all surfaces of the input model (often depending on the user defined settings for the number of triangles to be used).
2. Create triangles on each surface while ensuring that each corner is coincident with at least one other corner.¹

To represent the sequence in which the STL algorithm has translated a 3D object into a series of triangles, a simple Processing script was devised. The script adds a successively larger Z-height to triangle vertices in the sequence in which they have been written to an STL file by the STL creating algorithm. The result is a new STL file which has been "stretched" upwards, revealing the order in which the triangles have been created by the algorithm while maintaining their X and Y positions. This "exploding" process reveals similar patterns in 3D to the 2D investigations undertaken with vectors.

¹ "Stereolithography Interface Specification," 3D Systems, Inc., October 1989.

III. MESHING: PREPARED MESHER



"Index model" of triangles (first at the bottom, last at the top) describing hemispheres from files output by various 3D modeling programs.

By manipulating the objects in a file to reflect the order in which the file's elements are found in the internal index, we can create reifications of a file's index. This process can be applied to 3D files to create "index models". The file components can be stretched upwards like a scroll in virtual space to reveal the sequence in which the elements are stored in a file.

These index or time-lapse representations reveal different patterns which can be easily distinguished from one another. An analysis of files output from eight different software modeling programs (above) reveals a variety of strategies in the sequencing of the elements of a triangulated hemisphere. Some programs (Type A) construct a hemisphere as a spiraling surface (like an orange peel) while others (Type B) sequence this object as series of slices (like quarters of an orange). Programs of Type C appear to have no logical order in their triangle indices. Each index drawing represents a kind of fingerprint, or DNA code, stored in a file which identifies a particular sequencing priority (whether human or computer-derived).

III. MESHING: PREPARED MESHER

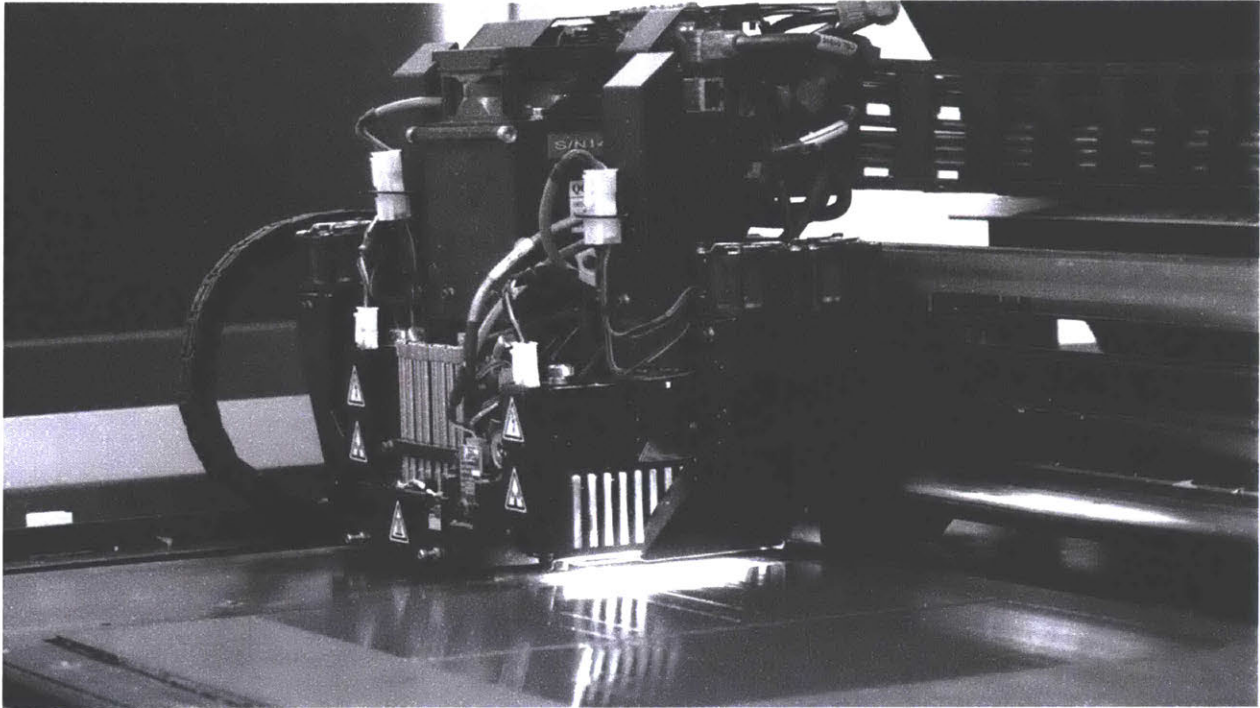
The variety of approaches to meshing a hemisphere offer a window into software engineering culture. Some programs appear to work primarily with quadrilaterals (which are subsequently divided into triangles) while others (such as Grasshopper) appear to work natively with triangles; some work clockwise, others counterclockwise; some build top to bottom, others bottom to top. Some build parts of the sphere and then mirror them while others do not exploit symmetry in any way. The programs also differ in the minimum number of triangles they will use to describe a hemisphere (this corresponds to the height of the "time-lapse" mesh.) Despite the fact that Autodesk now owns many of these software programs, programs which evolved as distinct pieces of software, they do not all work in the same way.

These variations are, as one Autodesk software engineer pointed out, "artifacts of arbitrary coding decisions". Many of these artifacts appear indifferent to optimization, they represent instead the whims of the engineer when faced with a decision which does not appear to have a right or wrong way of proceeding. Does he or she think of peeling an orange clockwise or counterclockwise?

In a more poetic reading, however, what is represented here is a moment of expression, idiosyncrasy and diversity in the software world most of us understand of as monolithic, machinic, and banal. The time-lapse meshes appear to reveal a kind of organic behavior coming from an ostensibly inanimate and unremarkable process. Any impulse to assume a neutral, objective character to the process of meshing is disproven.

We can understand this diagram to reflect the labor conditions under which this software was produced: these piece of code were made by individuals dividing up tasks and then stitching back together their work behind the scenes to produce the illusion of a monolithic software package.

III. MESHING: PREPARED MESHER

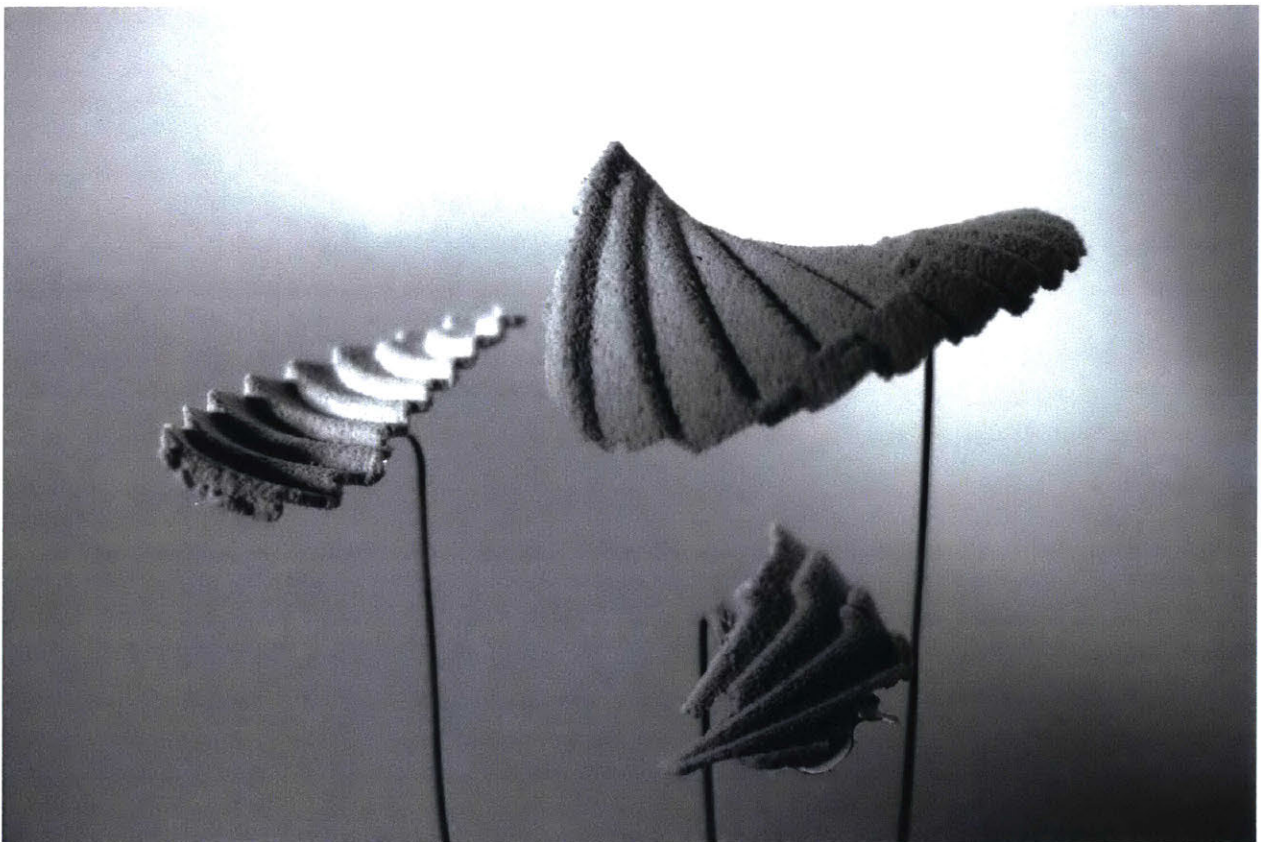
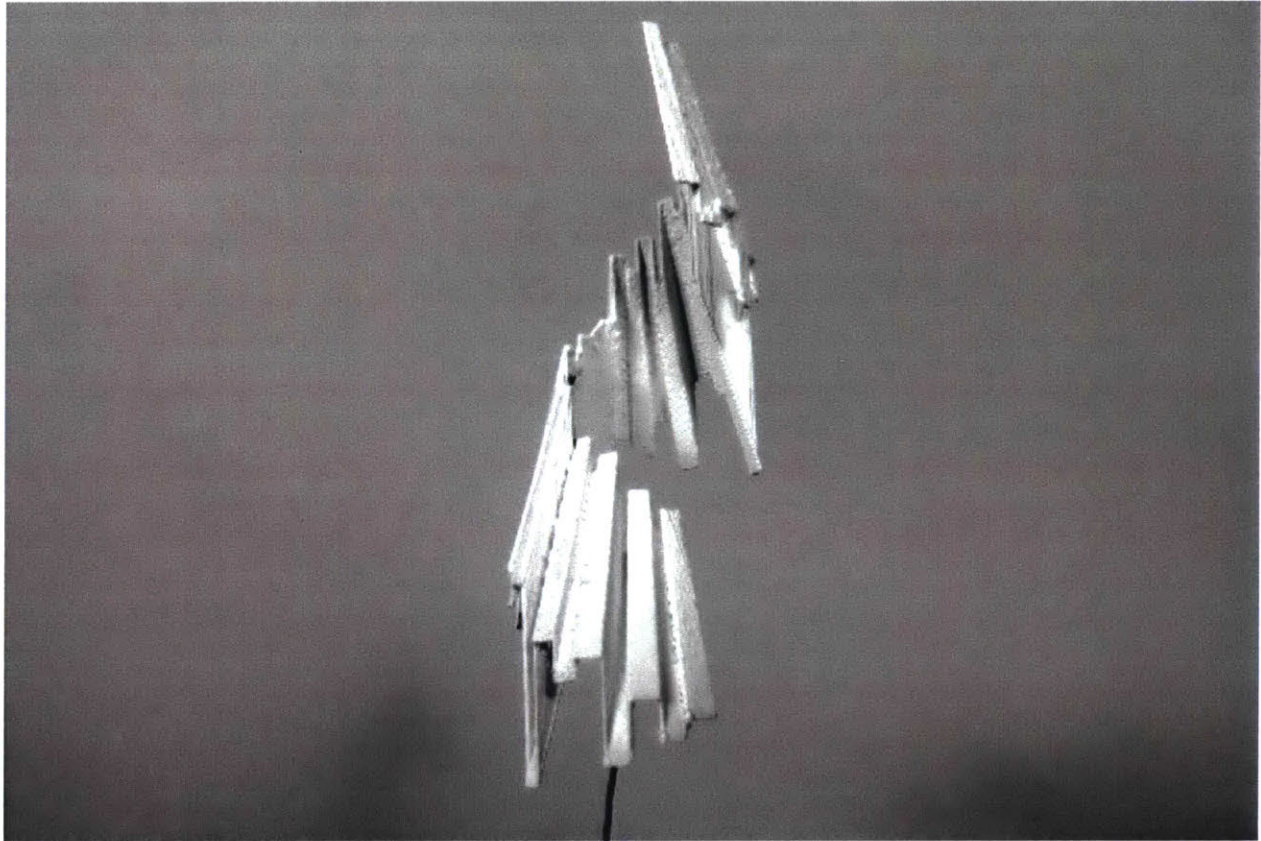


Connex Objet 500 Resin Printer

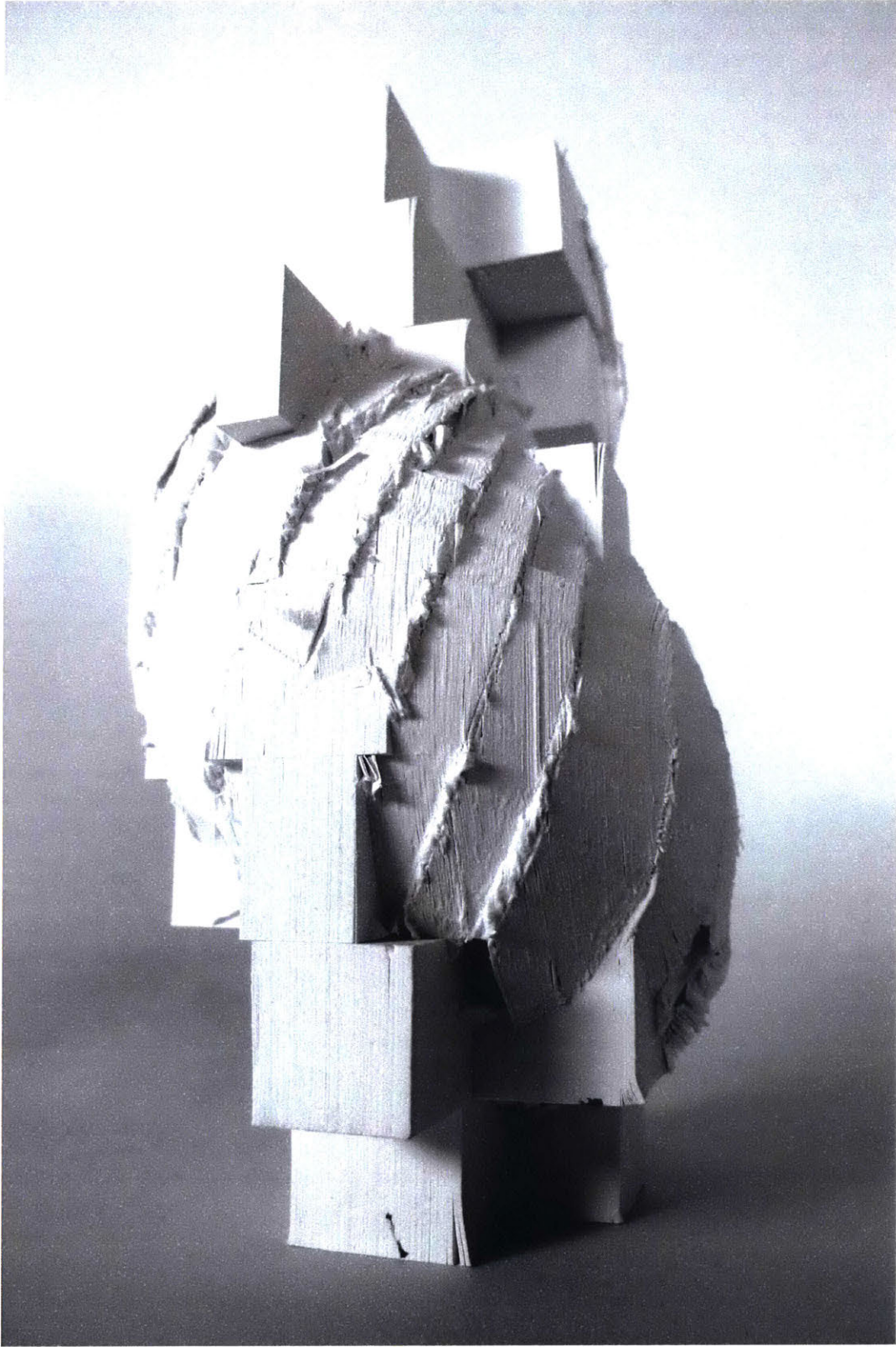
Once a method was developed for creating time-lapse mesh files, the difficulty of materializing them became immediately apparent. Several experiments were made with various 3D printing technologies including starch 3D printing, Laminate Object Manufacturing (LOM) printers using paper, and Fused Deposition Modeling (FDM) printing. Finally, the Connex Objet 500's resin printer was chosen. The Objet 500 can print in multiple materials, including transparent and opaque. Due to its high resolution, it can render complex exploded models in full detail, giving the illusion of triangles "suspended" in a clear resin.

In contrast to the drawing machines from the previous section, this prepared meshing machine toolchain involved a highly sophisticated industrial tool in place of a DIY assembly.

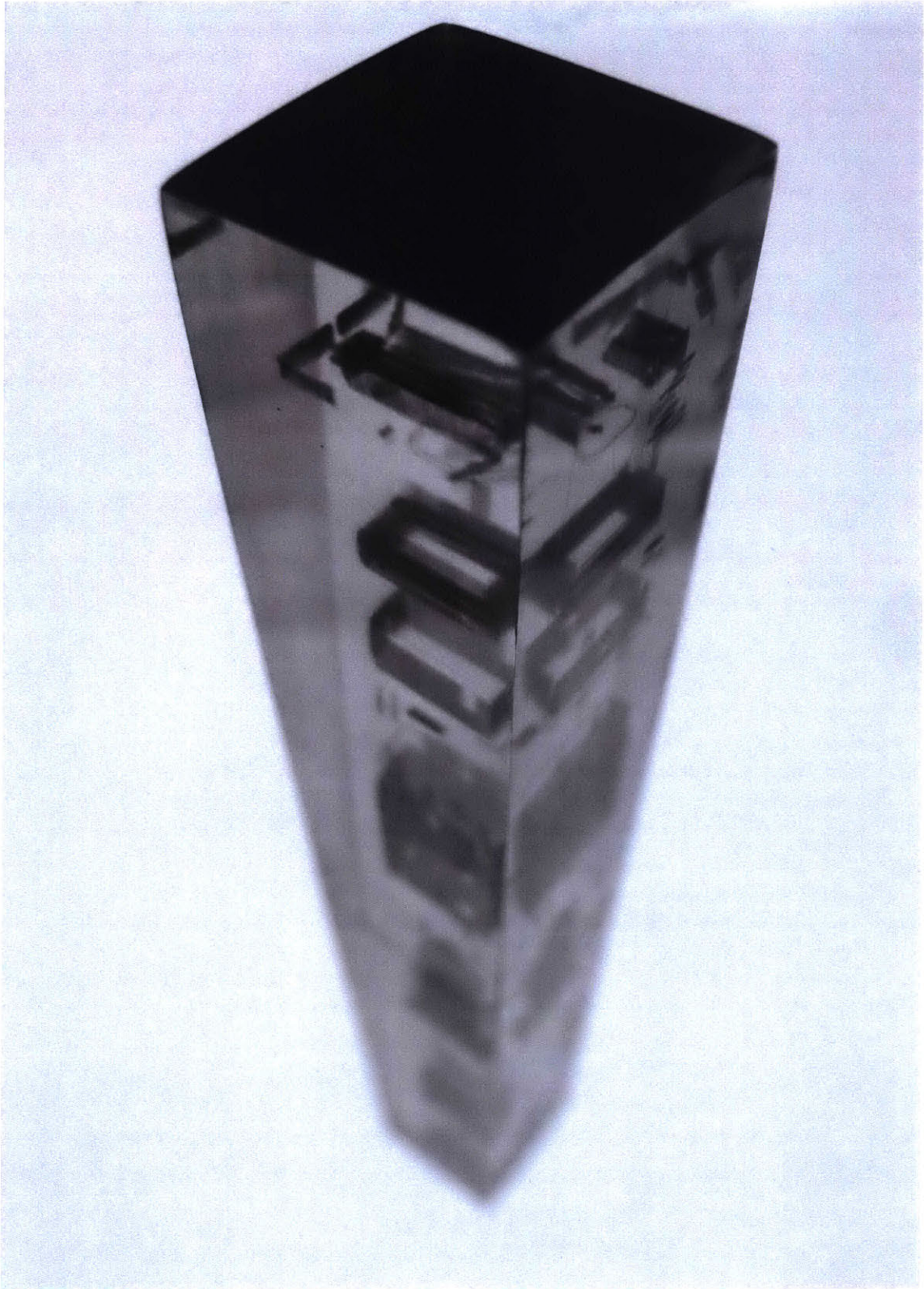
The following experiments were conducted at Autodesk's Pier 9 workshop in the spring of 2016 as part of the Artists in Residence (AIR) Program. The research was used to produce a series of prints commissioned by Autodesk for the December 2016 Pier 9 Year End exhibit. This project received funding from MIT's Council for the Arts Grant.



Time Lapse Cone in FDM (above), Time Lapse Sphere in starch (below)



Time Lapse Sphere from LOM MCor Iris 3D Print



Timelapse Theatro del Mundo Objet 500 resin print detail

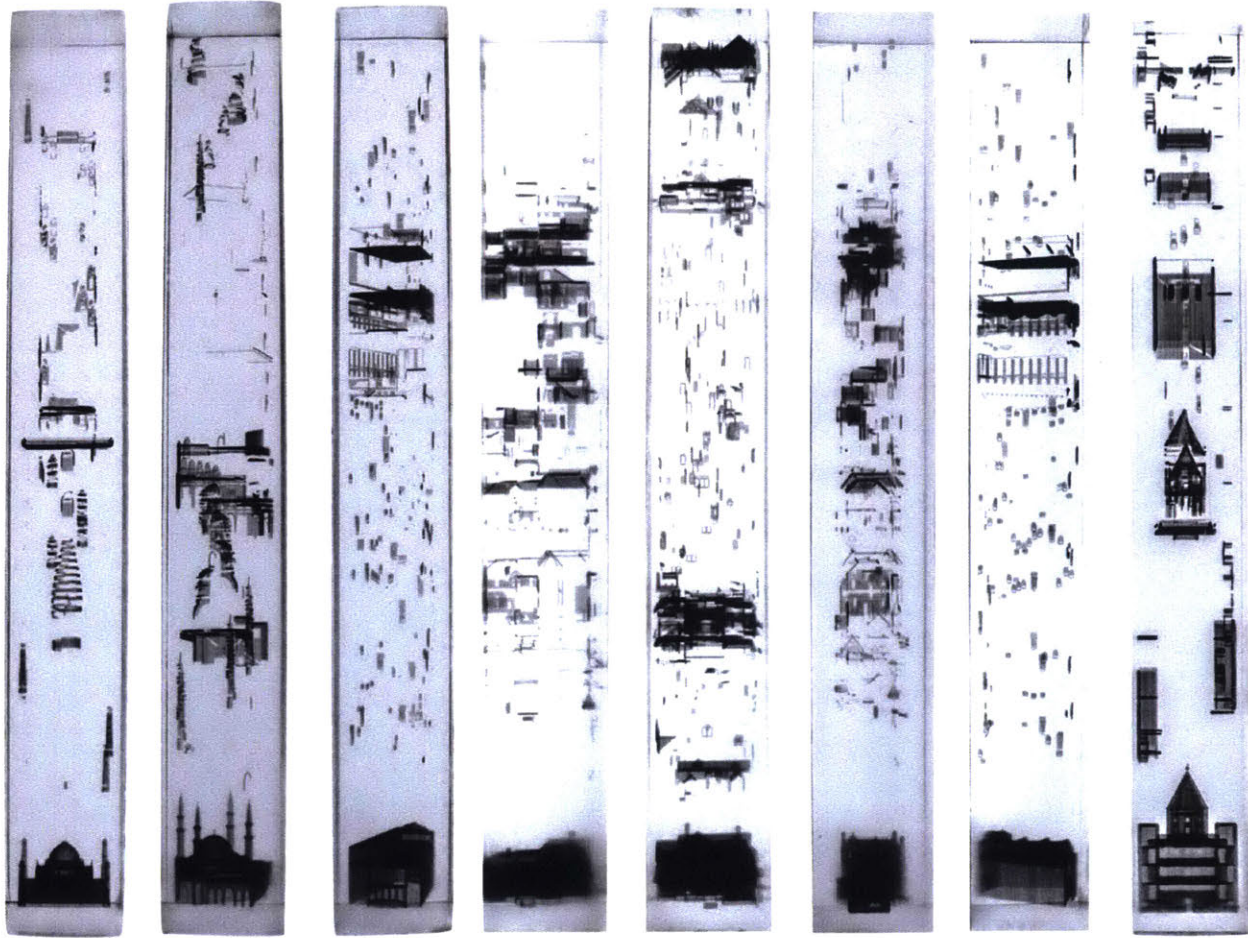
III. MESHING: MESHING SERIES No.1



Various Time Lapse 3D Prints

The 3D objects chosen for printing were sourced online from sites like GrabCAD and BiblioCAD. The files were taken as "found objects," converted into triangles (by exporting the files as Stereolithography files from Rhino 3D), and then manipulated with the STL Exploder processing script to create time-lapse index representations.

Each print is a rectangular prism approximately 6" tall, 3/4" wide in each direction. The prints are made from Tango Black and Vero Clear resins, and require a rigorous sanding and polishing process before they are sufficiently transparent to see through.



01

02

03

04

05

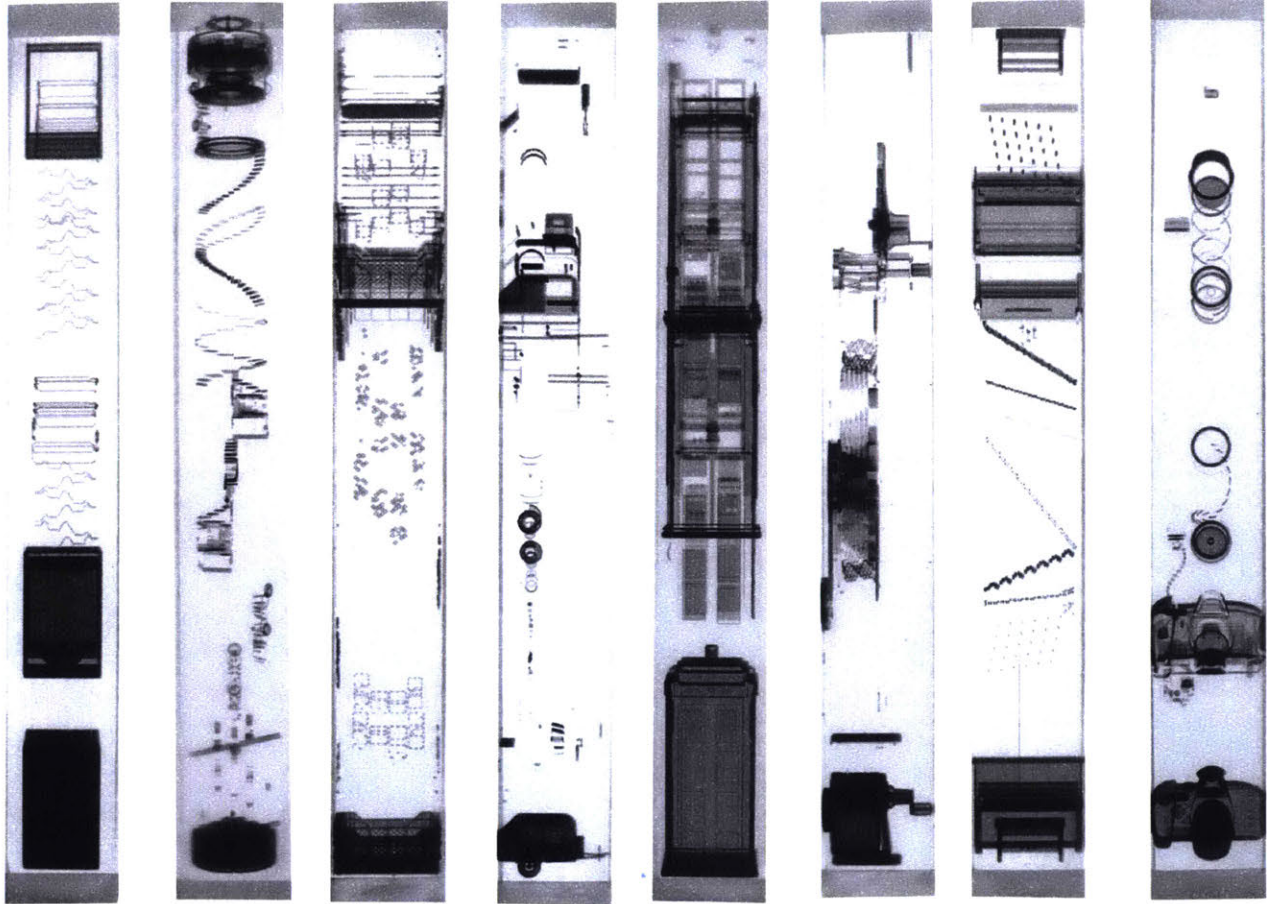
06

07

08

- 01 Taj Mahal
- 02 Hagia Sophia
- 03 Building I
- 04 House I
- 05 House II
- 06 House III
- 07 Building II
- 08 Teatro del Mundo

1A: Assorted 3D Files (Elevation View)



01

02

03

04

05

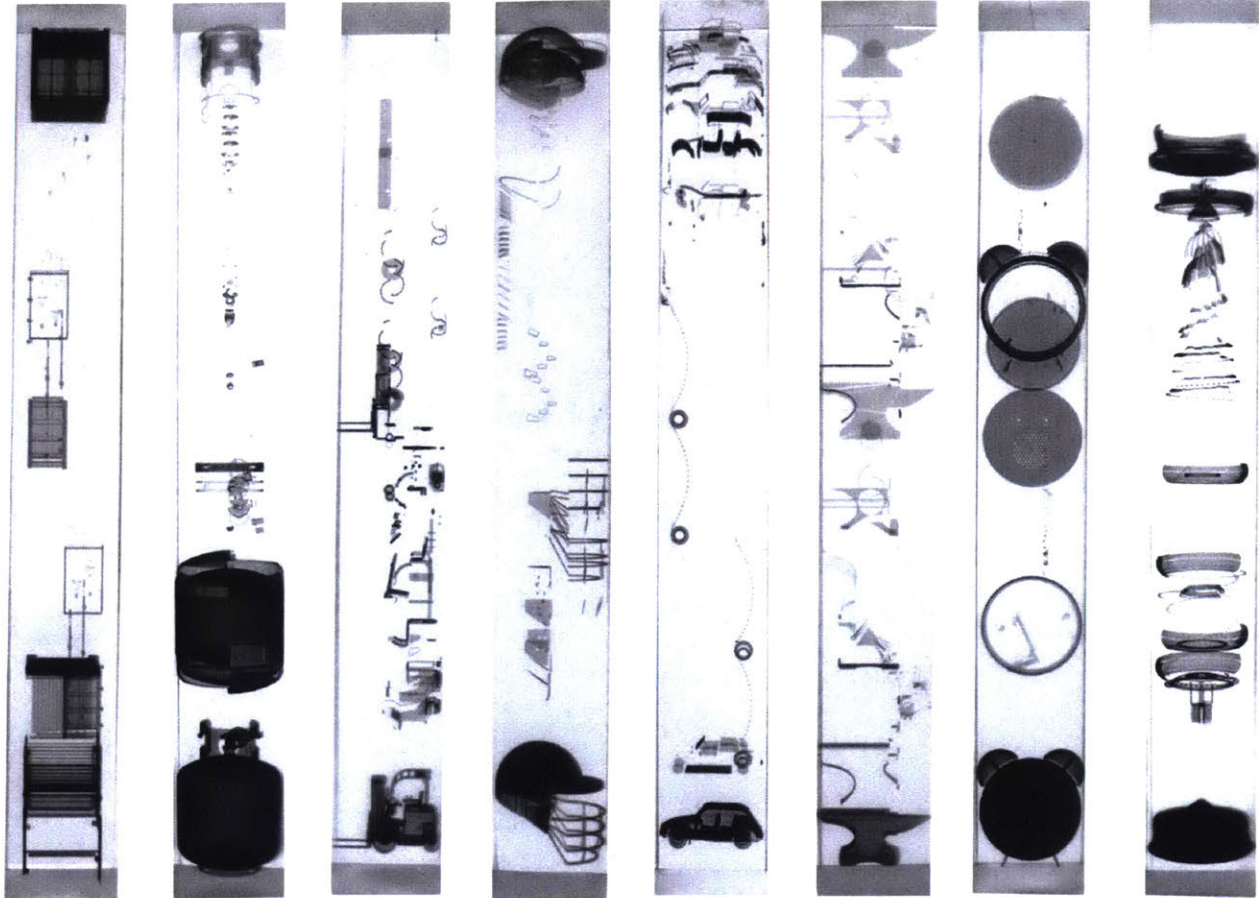
06

07

08

- 01 Dishwasher
- 02 Bit lock
- 03 Milk crate
- 04 Trailer
- 05 Phone booth
- 06 Pencil Sharpener
- 07 Piano
- 08 DSLR Camera

1B: Assorted 3D Files (Elevation View)



01

02

03

04

05

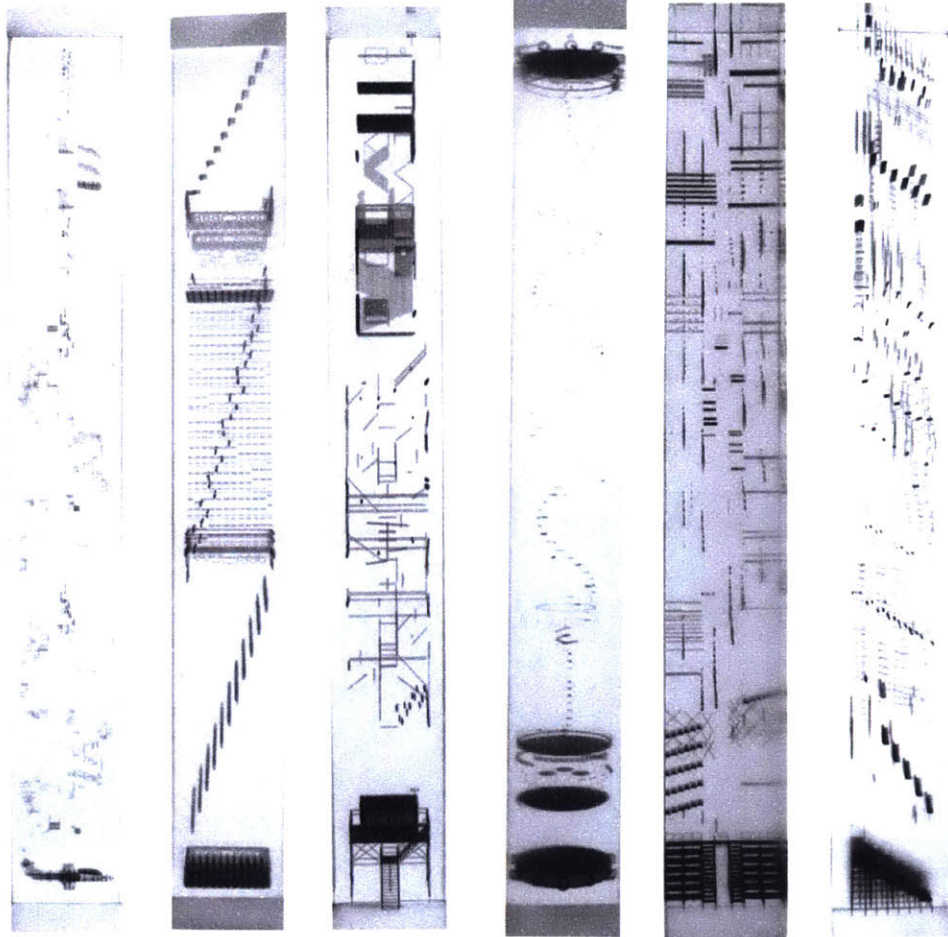
06

07

08

- 01 ISO Container
- 02 Propane tank
- 03 Forklift
- 04 Cricket Helmet
- 05 Citroen 2CV
- 06 Anvil
- 07 Alarm Clock
- 08 Lemon Squeezer

1C: Assorted 3D Files (Elevation View)



01

02

03

04

05

06

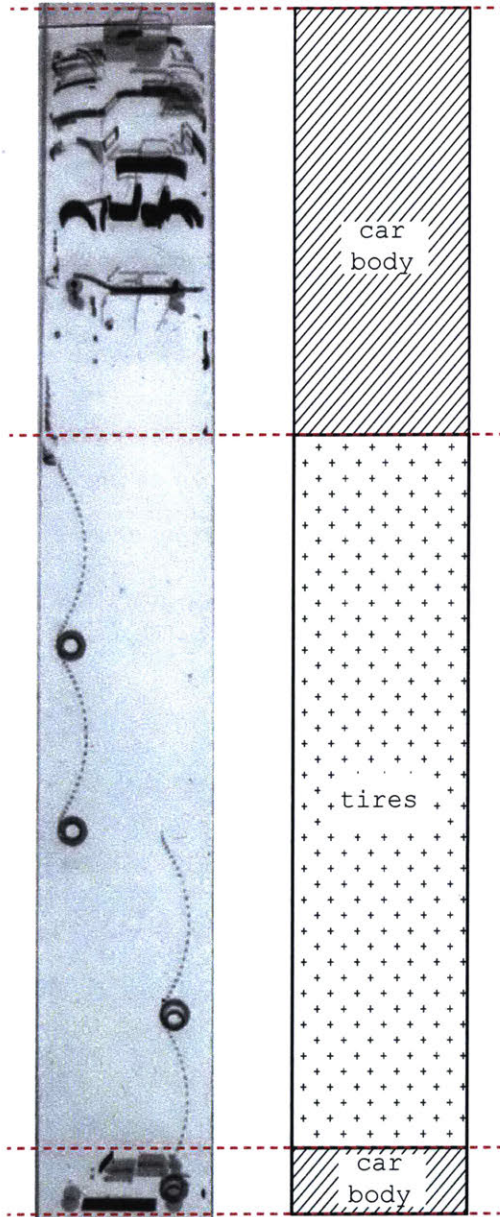
1D: Assorted 3D Files (Elevation View)

- 01 Airplane
- 02 Dish tray
- 03 Lifeguard Hut
- 04 Wrist watch
- 05 Bleachers I
- 06 Bleachers II



MIT Arts on the Radar, September 2017.

III. MESHING: CONCLUSIONS



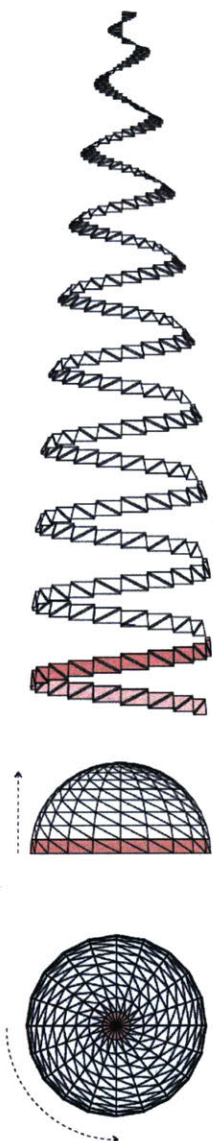
The time-lapse index representation acts like a kind of centrifuge for data. In the car model to the left, approximately 60% of the file's triangles describe the tire treads. The cabin cockpit and the body panels of the car take up relatively little of the overall data in comparison, despite the fact they are the most noticeable elements of the car from the software user's perspective.

Other exploded models reveal similar unexpected re-readings of everyday objects. The anvil, a massive object in our experience, takes on a lightness in being stretched apart. A humble lemon squeezer becomes a heroic jet engine, a banal set of bleachers becomes a musical score, a mysterious black box reveals itself to be a dishwasher and an compressed propane tank takes itself apart tidily and without incident.

tires ≈ 60%
car body ≈ 40%

Time-lapse Citroen 2CV

III. MESHING: CONCLUSIONS



A Time-lapse Hemisphere in Autodesk Maya



"Usually I go in rings from top-to-bottom, 'around' the sphere inside each ring, and then handle the top and bottom polygon separately. But there is also no reason you can't go the other way, in 'slices' of the orange, (metaphorically)."

-Sr. Principal Research Scientist at Autodesk and Creator of Meshmixer
Ryan Schmidt



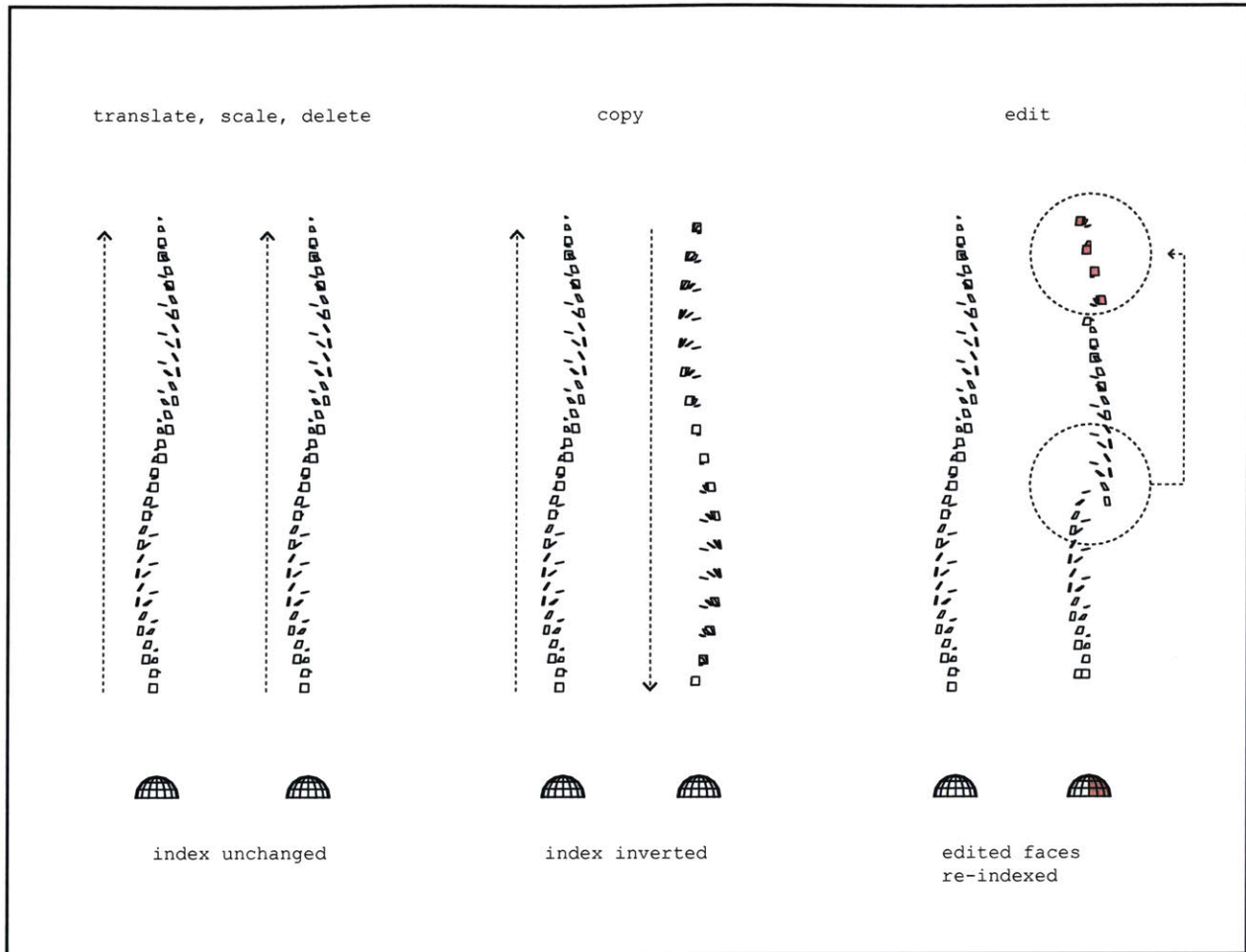
"The peeling an orange form for 3DS, Maya, and Fusion are just about what I'd do, too - it's the 'obvious' program to write."

-Eric Haines, Sr. Principal Engineer, Suites OGS, PDG-CP-Cloud Platforms

◀ Perhaps the most valuable insight from this Meshing foray appears to be the representing of different software programs achieving the same task in slightly different ways. Even though meshing a hemisphere is an extremely fundamental operation, it is possible to find a kind of Cambrian explosion of meshing strategies here.

In the comparison of hemispherical meshing, the time-lapse representation shows itself to have potential as a tool for analyzing simple algorithms through visual comparison. This form of visual description also invites the possibility of architects one day inverting this process by describing code spatially, illustrating a sequence which could then output a piece of code.

III. MESHING: CONCLUSIONS



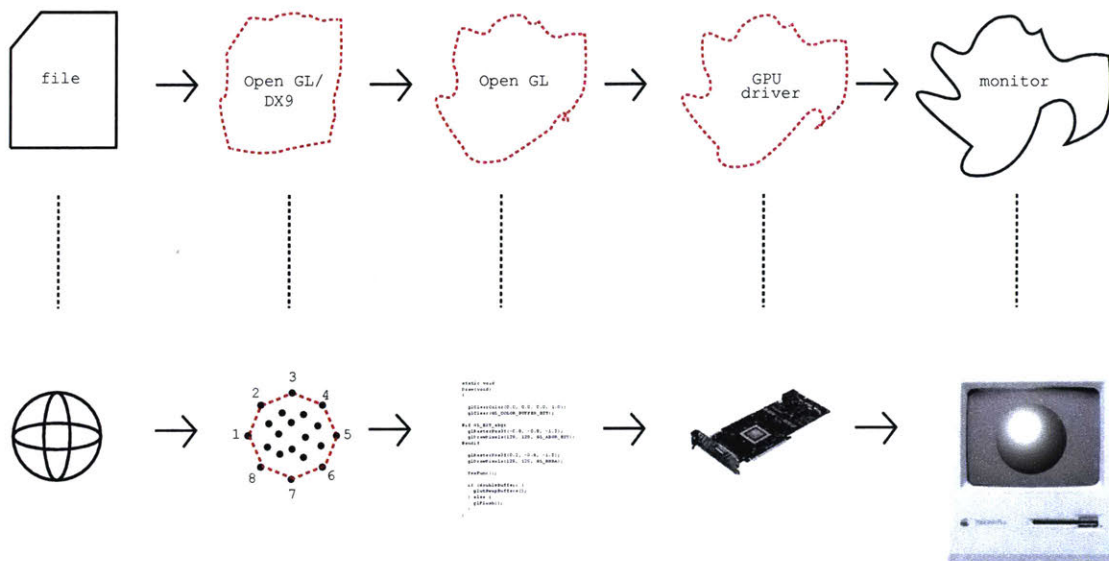
Re-indexing Time-lapse Study in Rhino

Taking the idea of representing triangles in a mesh a step further, it is possible to show how various modeling operations (like copy, scale, delete, edit, etc.) reshuffle the index of the triangles in the sphere. As seen above, in Rhino, scaling, translating and deleting faces on a sphere have no effect on the overall sequence in the index. However, copying and pasting a sphere results in an inverted sequence of the original triangle faces. Editing faces in part of the sphere throws those triangles to the end of the index stack.

The following section, Rendering, will first examine the sequences in which modeling elements exist in software databases before going on to explore CRT display processes.

IV. RENDERING

IV. RENDERING: BACKGROUND PT.I



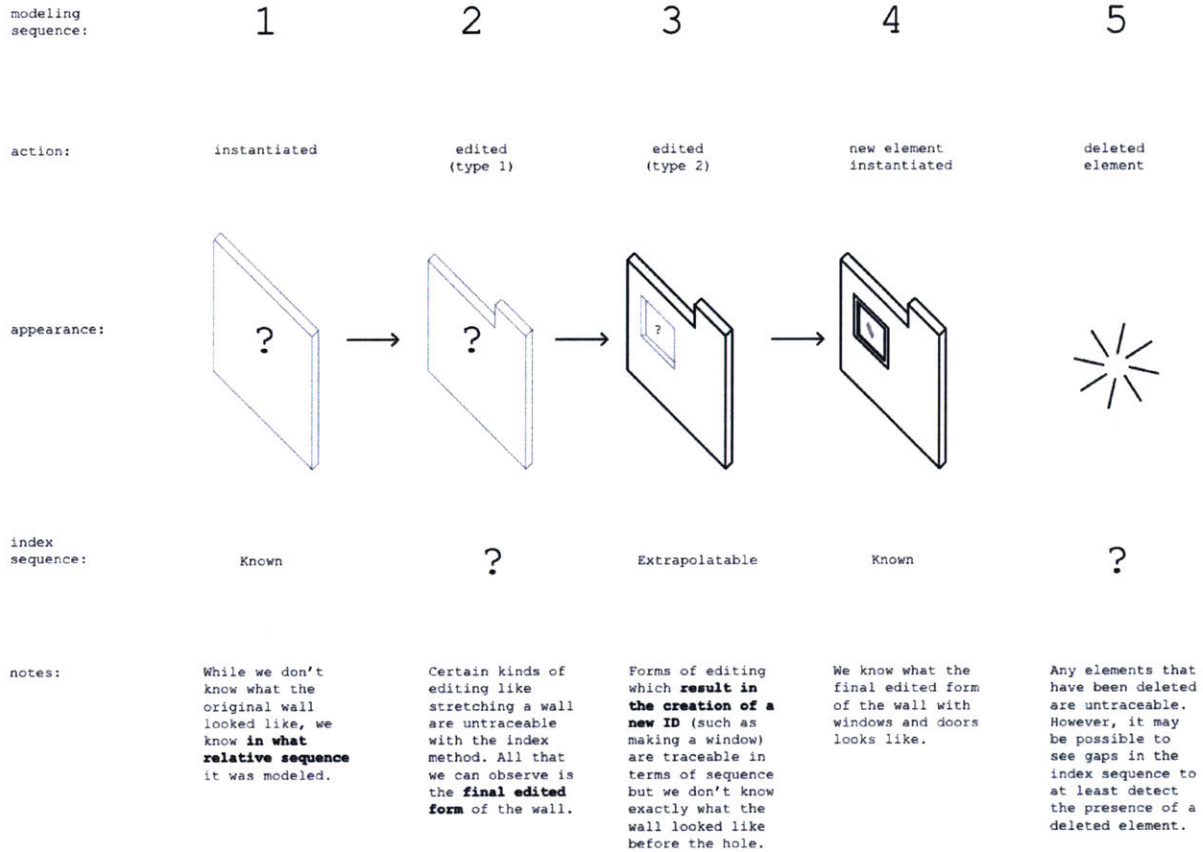
Rendering Toolchain

What happens inside the computer to display our drawings and models on screen?

Inside a modeling program are typically two representations of the contents of our file: i.) a database which stores the constituent drawing / modeling elements and may define their relationships, and ii.) a display list (or screen buffer) destined for our screen which is produced by the modeling program using the database.

The display list (ii) order depends largely on the database (i) order. Therefore the first part of this Rendering section is an attempt to determine the sequence in which programs like Revit and Rhino store objects in their databases. The database sequence is the raw material from which the display list is generated and has the greatest influence over the sequence in which image and model elements appear on our screen.

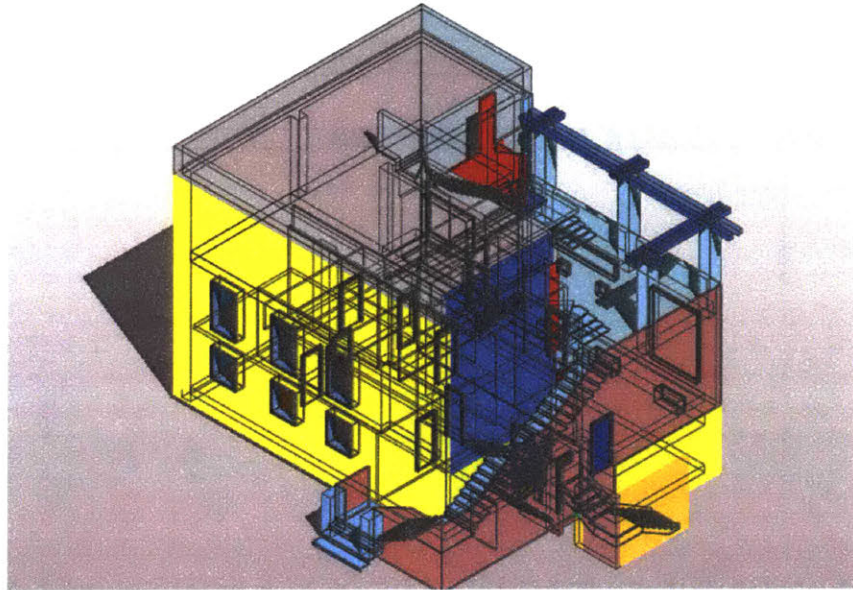
IV. RENDERING: BACKGROUND PT. I



Revit Object ID tracing

The sequence in which a program displays model elements typically depends on the sequence in which elements appear in the internal program index. Different programs index elements in different ways but most indices are mutable. Users can change the sequence in which elements appear in the internal database by editing parts, throwing those newly edited parts to the end of the database. Revit is an exception; unless a model is being worked on in collaboration mode, where two or more versions of the same file exist in parallel and eventually have to be merged into a single sequence, the Revit index is immutable. Therefore a deleted element produces a gap in the Revit Object ID index and an edited object keeps its original place in the index. As a result, it is possible to look at the relative "ages" of different Revit elements in a file and recreate a micro-history of sorts based on this information. This comes with various caveats: we can only see what a Revit element looks like in its most recent instantiation, and we can't see how it has been edited over time.

IV. RENDERING: BACKGROUND PT.I



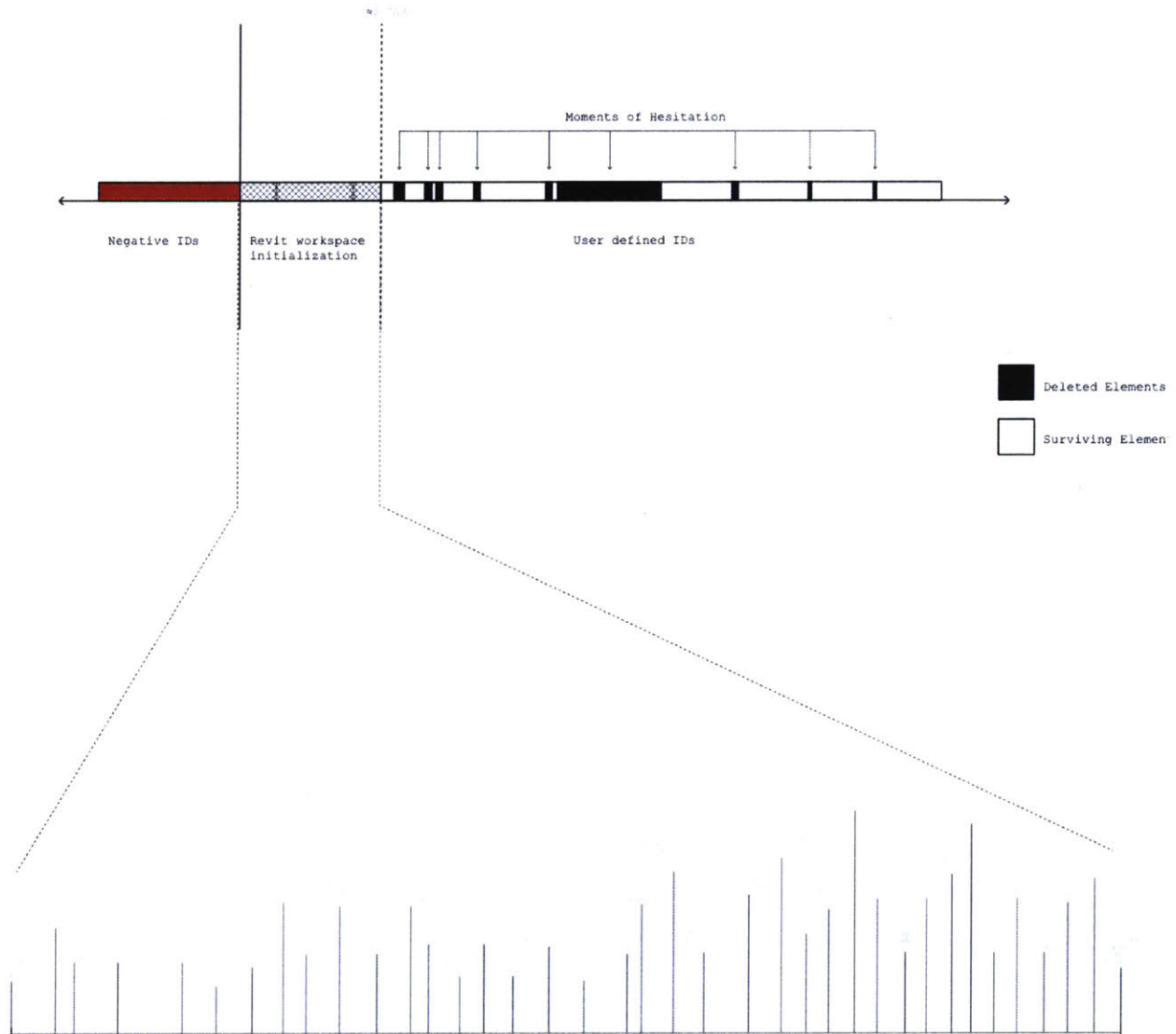
FIRST EXTANT ID#

LAST EXTANT ID#

Revit Extant ID "age" representation. 3D Villa Moissi model provided by Prof. Takehiko Nagakura.

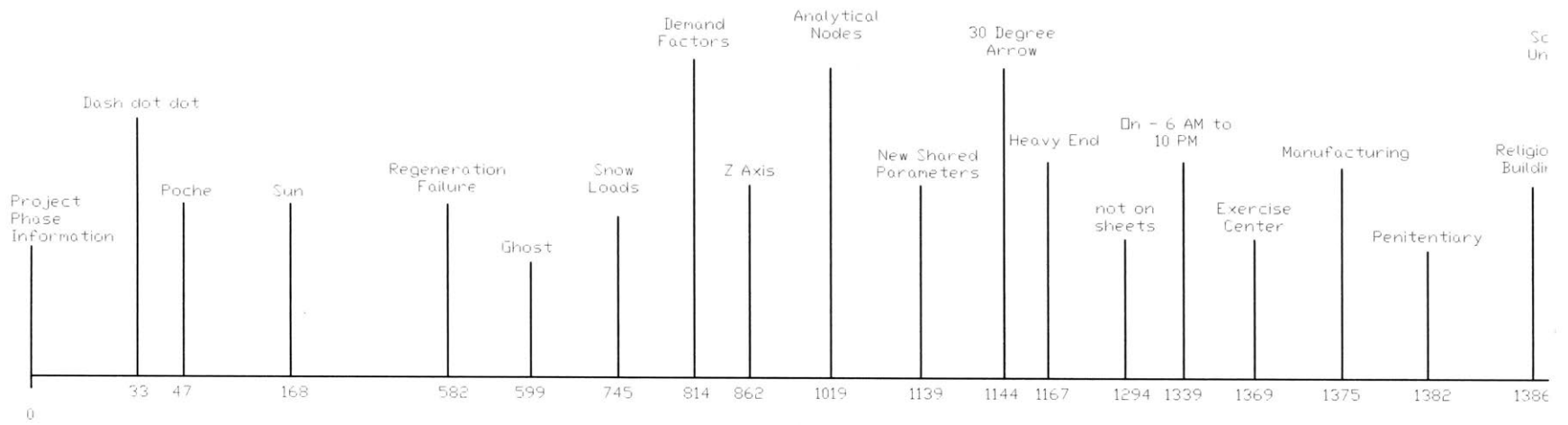
Using a simple Dynamo script, the sequence of user-instantiated Revit elements (like walls, windows and stairs) can be extracted from a given Revit file and assigned a color based on its "age" relative to the other extant elements in the file. The diagram above shows a Villa Moissi 3D model with the earliest extant elements in yellow and the most recent ones in blue.

This representation of a Revit file's partial "history" could allow us to gather information about how people use software programs, aggregating data to build an understanding of how our software influences the kinds of design decisions we make as designers. By comparing different users' attempts to model the same building, we can glean information about the different design priorities of users when faced with similar tasks.

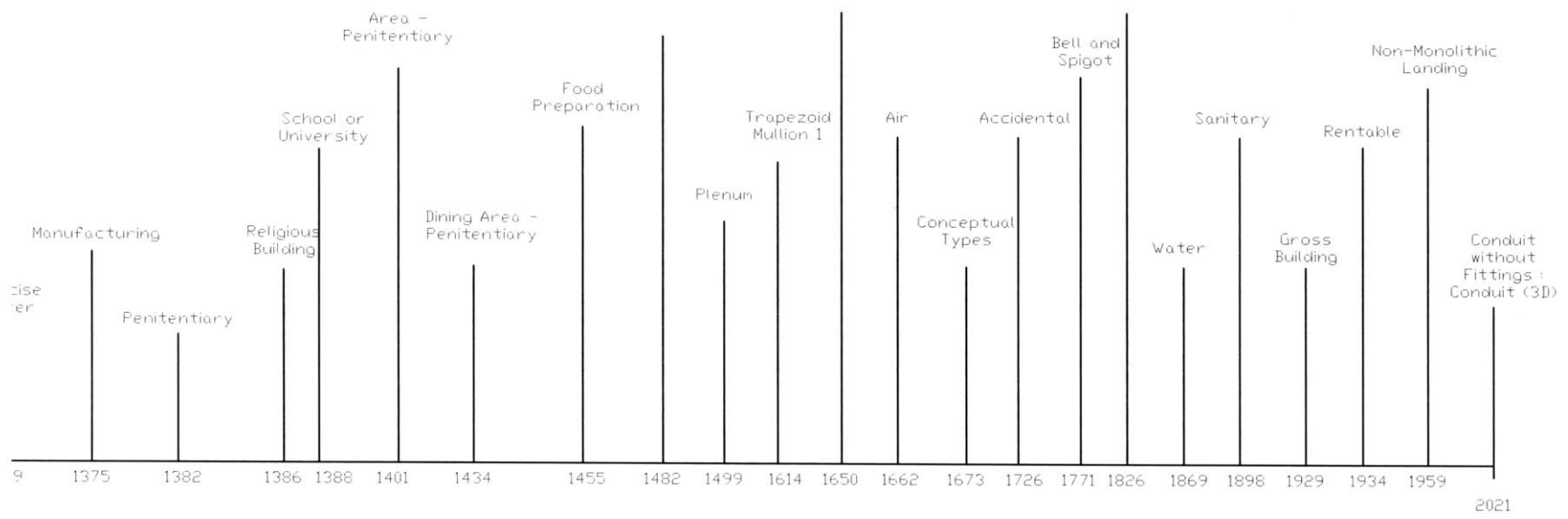


Revit-instantiated Objects in the Revit ID list

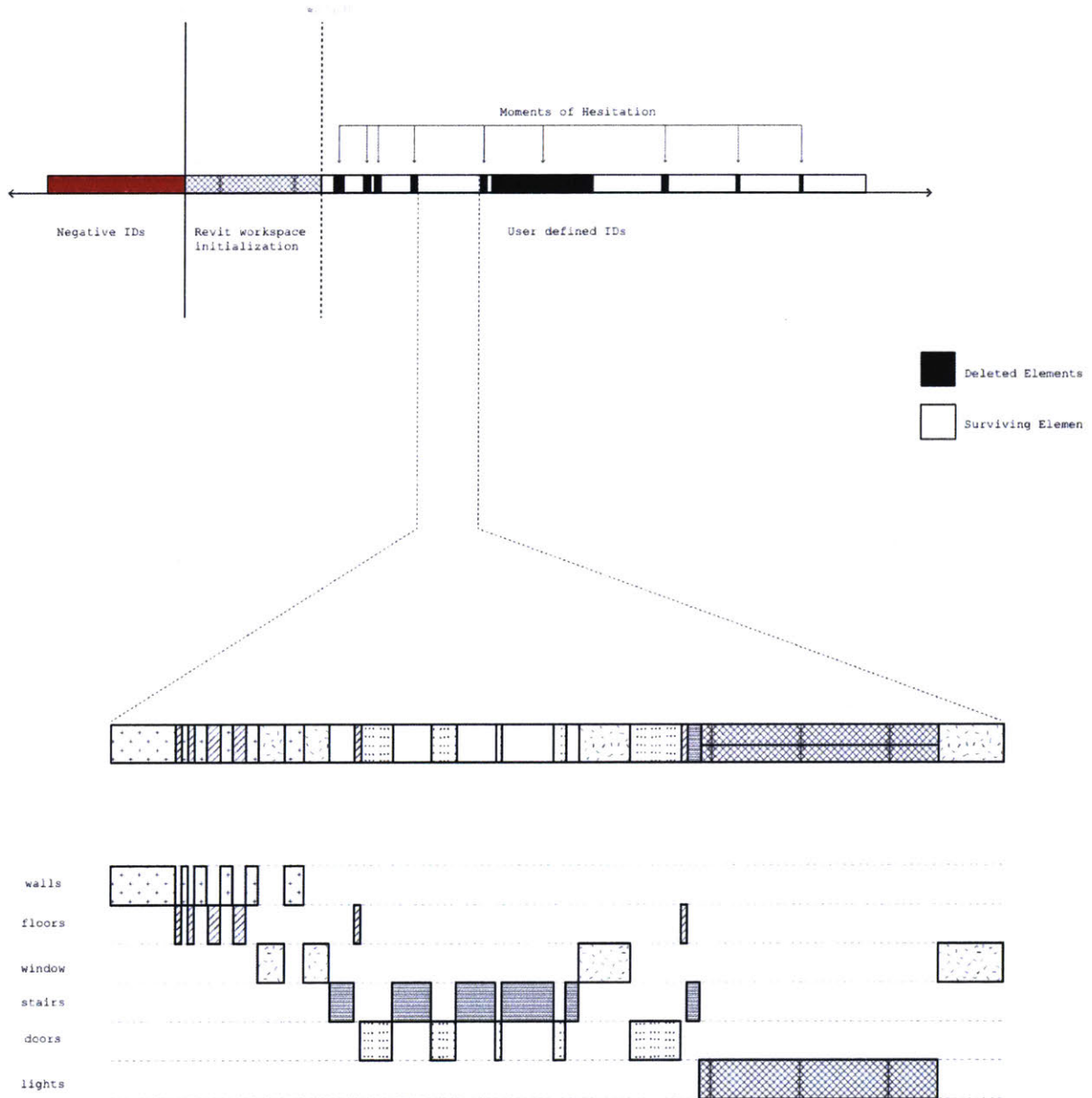
"Elements" in Revit include building components but also drawings, views, hover-activated notations, and other internal runtime software events. Revit assigns each element in a file a Revid ID number. Using the Revit API, it is possible to produce a chronological list of the objects which exist in a given file. What we observe by examining the Revit ID list is that the beginning of the database records Revit itself instantiating software environment objects. The first part of the list (shown on the subsequent pages) could be said to represent the setting-up of the Revit modeling space for the user. Examining the first 2,000 elements closely reveals Revit's world-view through its world-building.



Revit-instantiated Objects in a Revit ID list

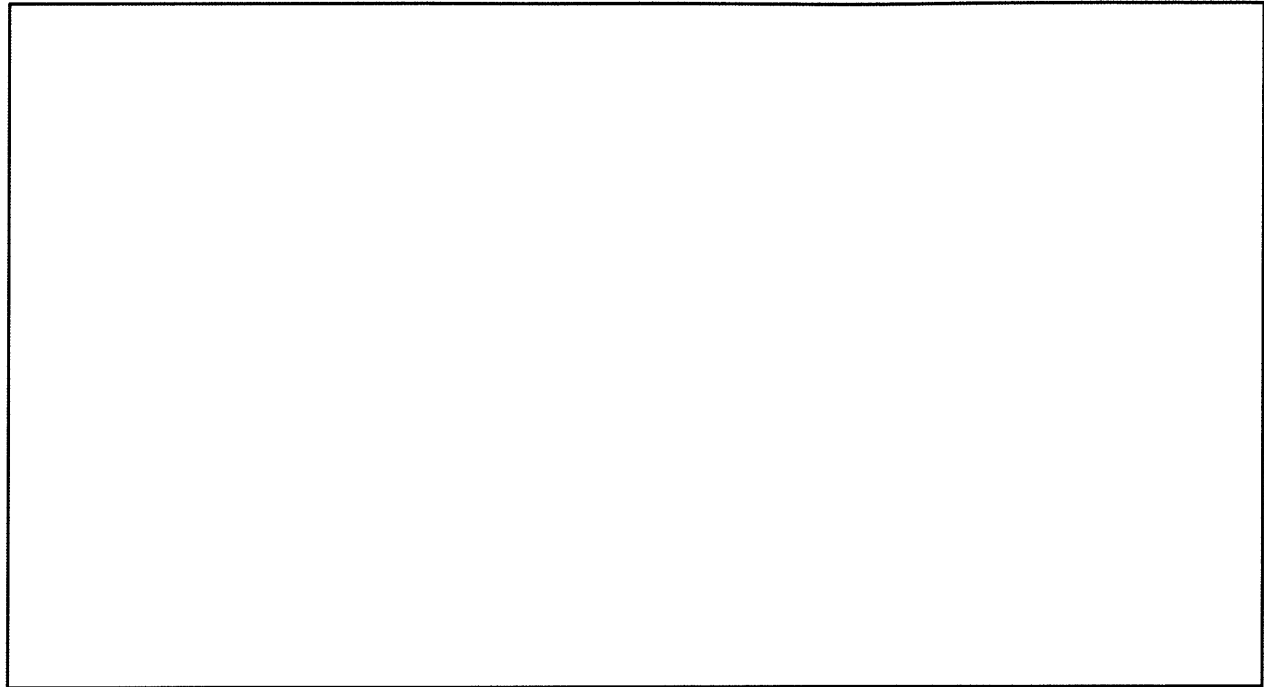


Revit-instantiated Objects in a Revit ID list



User-instantiated objects in Revit ID list

After the modeling environment is created by Revit, user-instantiated objects begin to appear in the database. With the Revit API it is possible to see the *types* of objects that were instantiated by the user over time in a given file. The diagram above shows how a user sequence can be broken down by type of object to provide a picture of modeling activity over time. In the above sequence, the user may have started by modeling walls, then walls and floors, followed by windows, stairs, doors and lights. Comparing two different user's models of the same building could illustrate two different ways of conceiving of architecture.



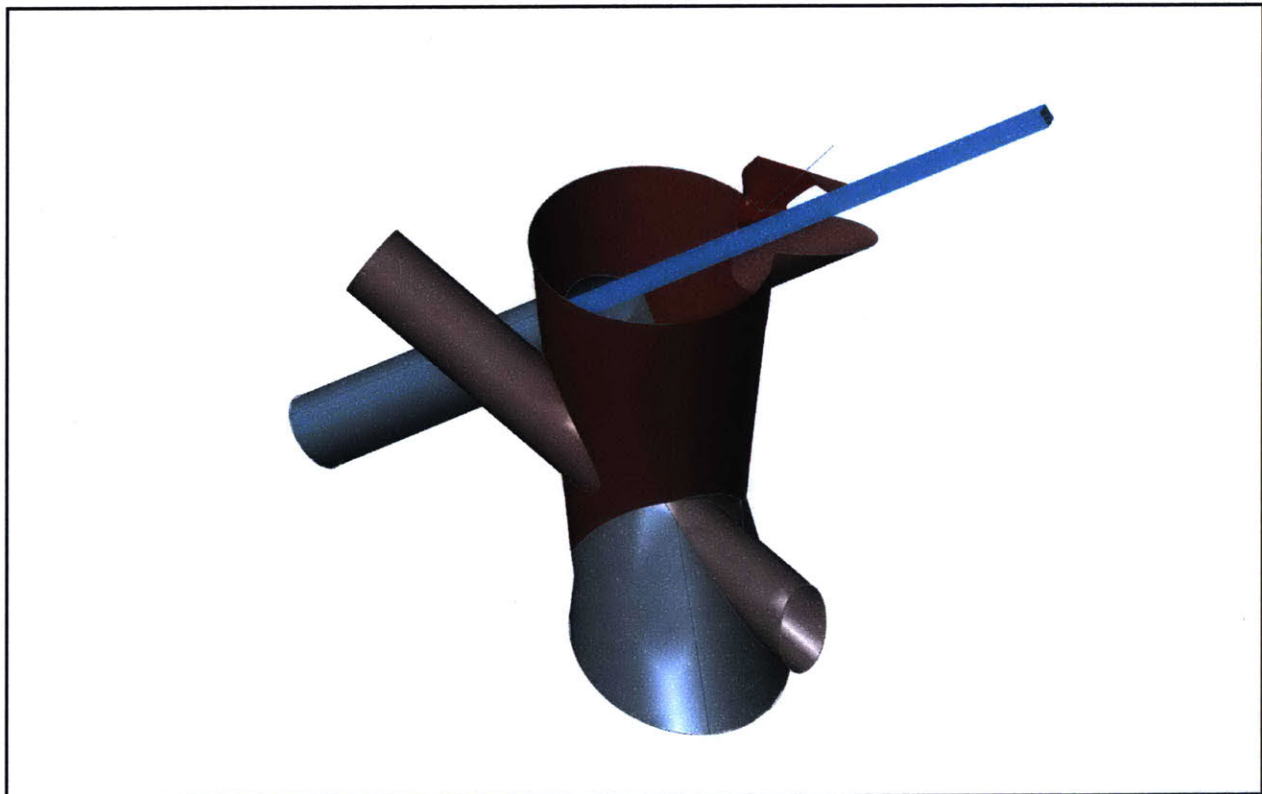
Villa Moissi Sequence of Extant Revit IDs. 3D Model provided by Prof. Takehiko Nagakura.

The diagram above describes a Revit modeling sequence by showing only newly instantiated objects moving from left to right, top to bottom.

The following page shows an attempt to use Grasshopper to represent the Rhino database of a given file. The Grasshopper script assigns colors to the sequence in which objects appear in Rhino's internal index. However, because Rhino's index is mutable, the color coding in this representation reflects the order in which objects have been *edited*, rather than instantiated, by the user.

<p>GU Object ID (Rhino ID) →</p> <p>GU User ID →</p> <p>Index Inverted if copied →</p>	<pre> surface ID: 89c40c70-f08d-11d4-8f86-000000000000 (93) Layer name: building Render Material: source = from layer index = -1 Geometry: Valid surface. Surface NURBS Surface *U*: degree =1 CV count = 2 (0.000 <= U <= 92.781) *V*: degree =1 CV count = 2 (0.000 <= V <= 91.671) Edge Tally: 4 boundary edges Edge Tolerances: all 0.000 Vertex Tolerances: all 0.000 Render mesh: none present Analysis mesh: none present Geometry User Data: UserData ID: 3B7FB506-437C-431e-B1D7-93C4CBFF417F Plug-in: Rhino description: Gumball grip frame saved in file: yes copy count: 1 </pre>
--	---

Rhino's Object Properties Info Panel displays information about the sequence in which objects were instantiated and by which user.



Detail from a Preston Scott Cohen Eyebeam Atelier Museum Rhino file from the Canadian Center for Architecture's Digital Archive. Color coded to show the most recently edited (red) to least recently edited (blue) surfaces.

IV. RENDERING: BACKGROUND PT.II

Revit and Rhino databases index objects differently: Revit has a (mostly) immutable index while Rhino has a mutable one. Their database orders appear to be determined by the order of user instantiation (Revit) or user editing (Rhino). To continue to investigate the initial guiding question regarding the sequence in which objects are rendered to a display, I now move from investigating the database to the algorithms which execute display lists generated from these databases.

For this next set of experiments I explore the sequence in which early Cathode Ray Tube Controllers (CRTCs) send information to be rendered on screen, using Ivan Sutherland's Sketchpad developed at MIT in the early 1960s as a case study.

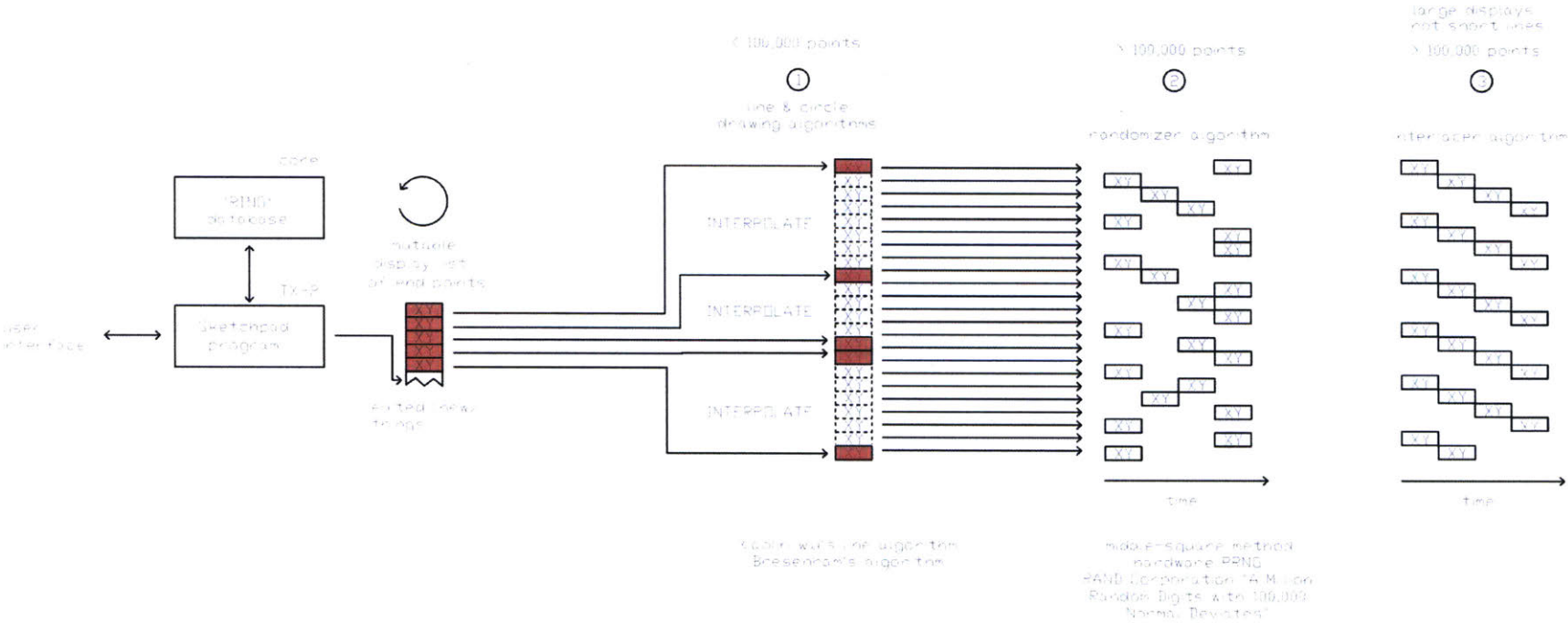
IV. RENDERING: BACKGROUND PT.II

The diagram on the following page describes Sketchpad's display process. Sketchpad takes user-inputs via the light pen and a series of switches, along with its database (called the "Ring" Structure) of user-instantiated drawing elements, to decide what should appear on the screen. Sketchpad takes user commands to zoom, add lines, show constraints, merge lines, etc., and produces a display list of endpoint coordinates in (X,Y) format. In his dissertation, *Sketchpad: A man-machine graphical communication system*, Ivan Sutherland describes how parts of a drawing which are being actively edited by the user are sent to the end (bottom) of the display list so they can be rendered without disrupting the rest of the image. The Sketchpad display list changes with each user input and can be reshuffled fundamentally at any time based on new data relationships being created. In this way, Sketchpad works with a mutable index similar to Rhino.

Sketchpad's display subprograms have the job of taking the display list provided by the Sketchpad program and turning the point coordinates into an image on a CRT screen. In Progressive Scanning, A Digital Differential Analyzer (DDA) first interpolates between the given point coordinates in the display list to plot points along vectors to create the illusion of a line. Depending on the complexity of the image being displayed, Sketchpad also had tricks to be able to represent images with more than the maximum number of points (100,000) for the TX-2 computer at MIT. These tricks are the Interlace and Randomizing algorithms which render different portions of the display file at different times quickly enough to give the illusion of displaying everything at once.¹

¹ These paragraphs cite: Ivan Sutherland, *Sketchpad: A man-machine graphical communication system*, MIT: 1963.

display subprograms

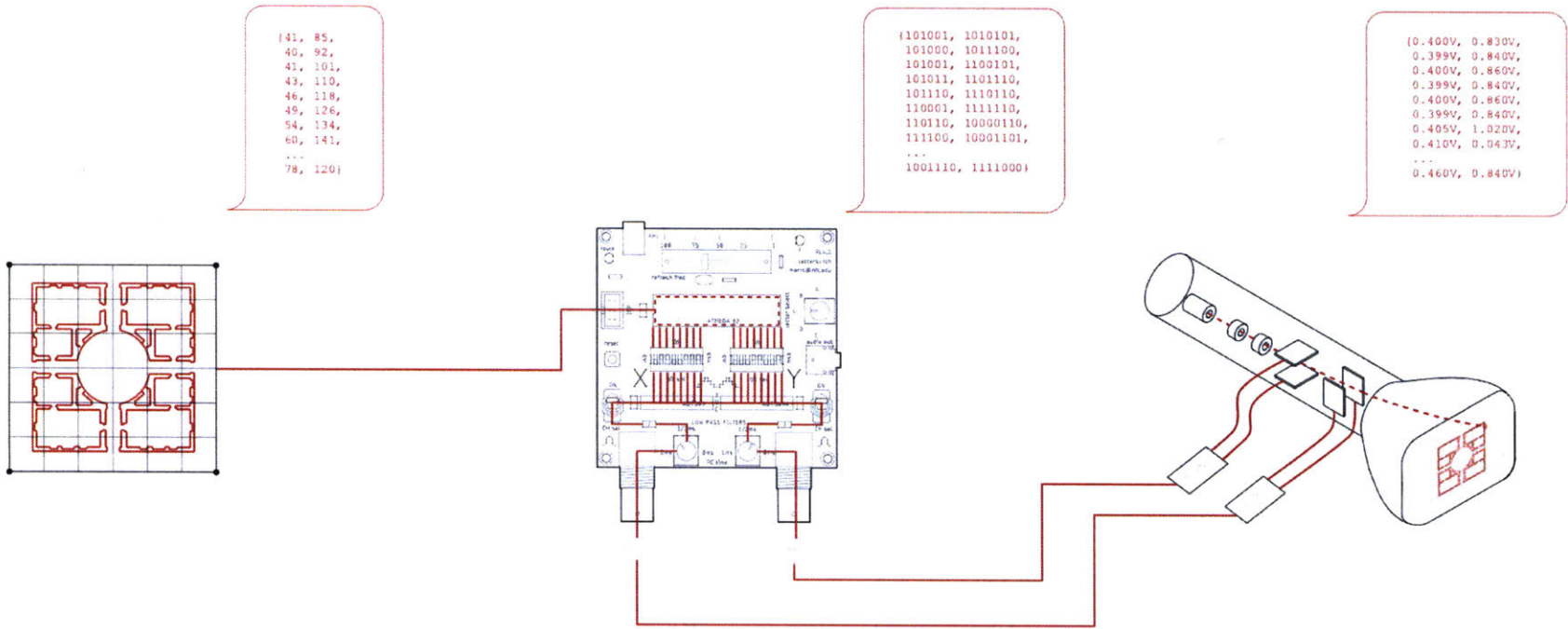


IV. RENDERING: PREPARED CRTC

In order to explore in greater depth the ways in which display lists are converted into images on a CRT screen, a prepared CRT Controller (CRTC) was developed to allow for manipulation of basic display parameters in real-time.

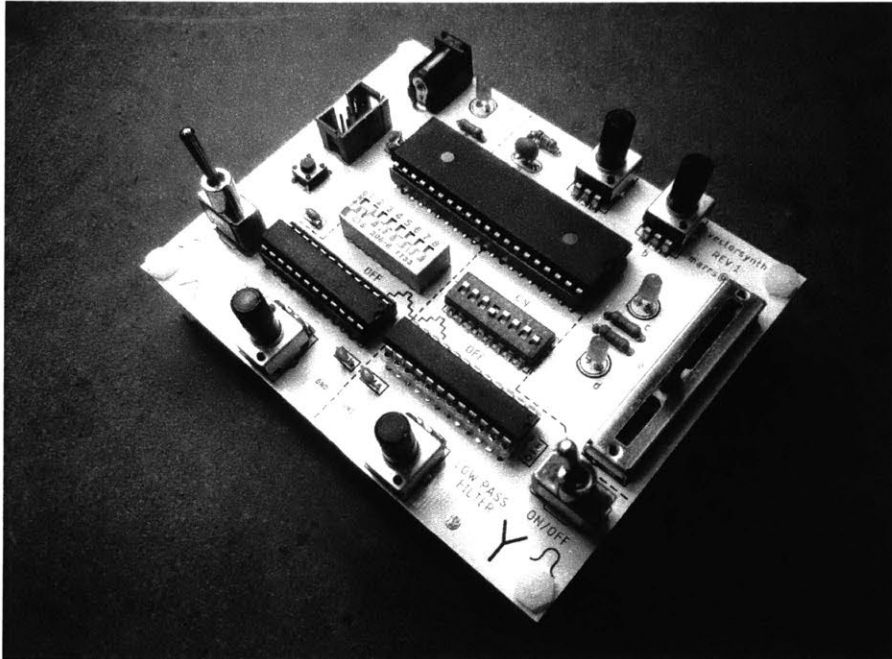
The diagram on the following page illustrates the description of the prepared CRTC toolchain. Using a simple Grasshopper script, a given input vector drawing is translated into a list of (X,Y) coordinate pairs describing the lines in the drawing. The list is then compiled into an array and flashed onto the microchip on board a custom CRTC. Once the CRTC is turned on, it reads sequentially from the point array stored in memory and sends the values in the array as bytes to an Analog-to-Digital converter. Here, 1's and 0's are translated into voltage levels between 0-2.5V for the Oscilloscope CRT.

The voltages generated by the CRTC travel through a pair of BNC cables to a CRT of an oscilloscope set to X-Y plotting mode. The X channel deflects the photon beam inside the oscilloscope in the left/right direction while the Y deflects the beam in the up/down direction. Together, the X and Y channel voltage levels function like extremely fast Etch-a-Sketch commands, pulling the beam around the screen where it strikes a phosphor coating on the inside of the vacuum tube and illuminates it for a period called the decay time. By constantly redrawing (refreshing) over the same image at least 30 times a second, the illusion of a still image made on the CRT can be generated.



Prepared CRTC Toolchain

IV. RENDERING: PREPARED CRTC



CRTC Rev.1

The CRTC1 allows for variation of several display parameters:

-By turning on and off switches in the two Dual In-Line (DIP) switch sets, bits can be removed from the bytes being sent from the microchip to the ADCs where they are converted into voltage levels.

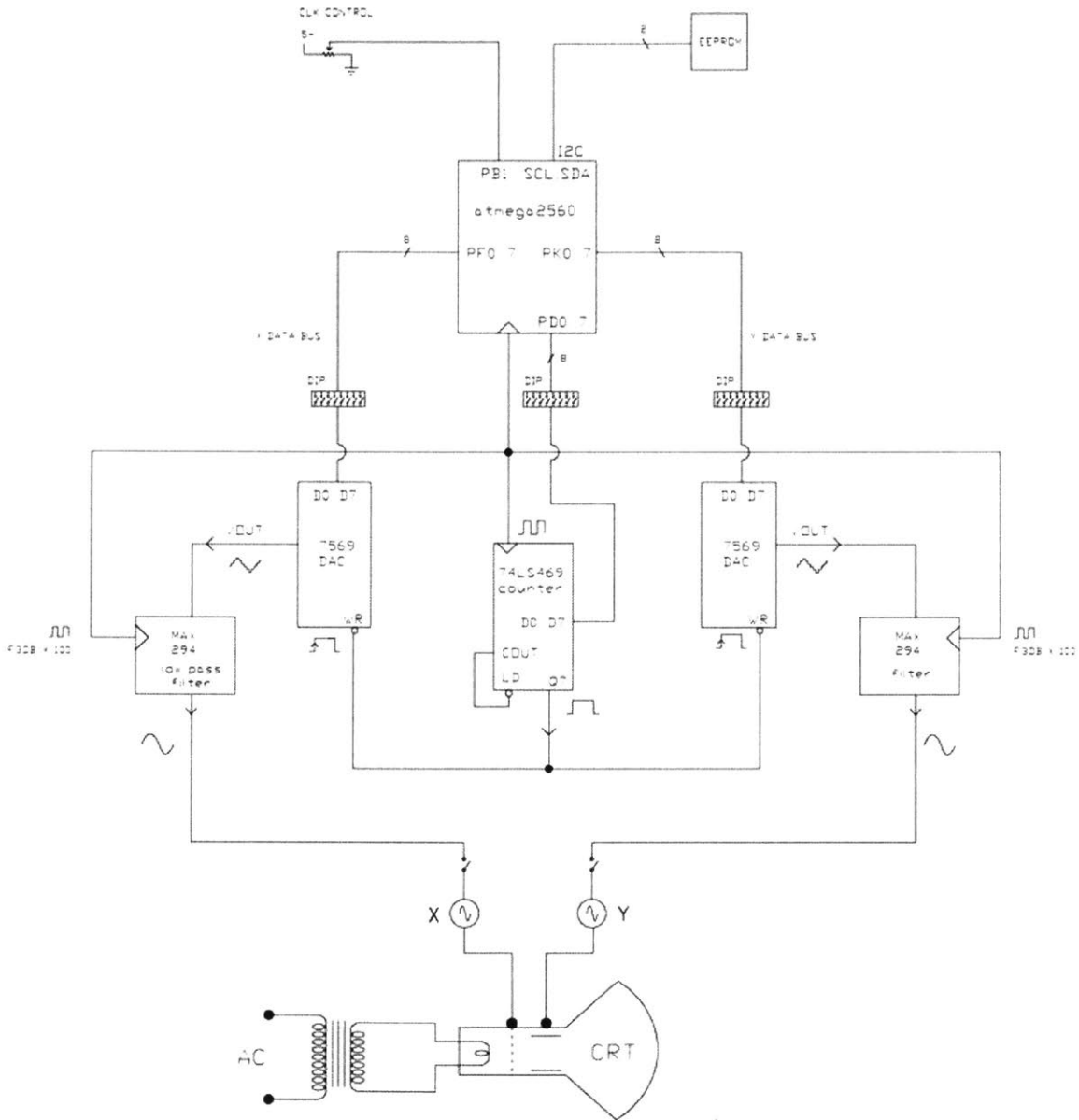
-Two flip switches allow for the X or Y channels to be individually turned 'on' or 'off.'

-Two potentiometers allow for varying of the low-pass filter's time-constant (RC), the factor which slows the movement of the beam between points to produce lines on our CRT screen.

-Finally, a potentiometer which is polled by the microchip is used to vary the delay time between the sending of bytes to the ADC to be converted into voltages. This knob functions as the refresh rate control, changing the overall speed at which the CRT screen is redrawing over the same image.

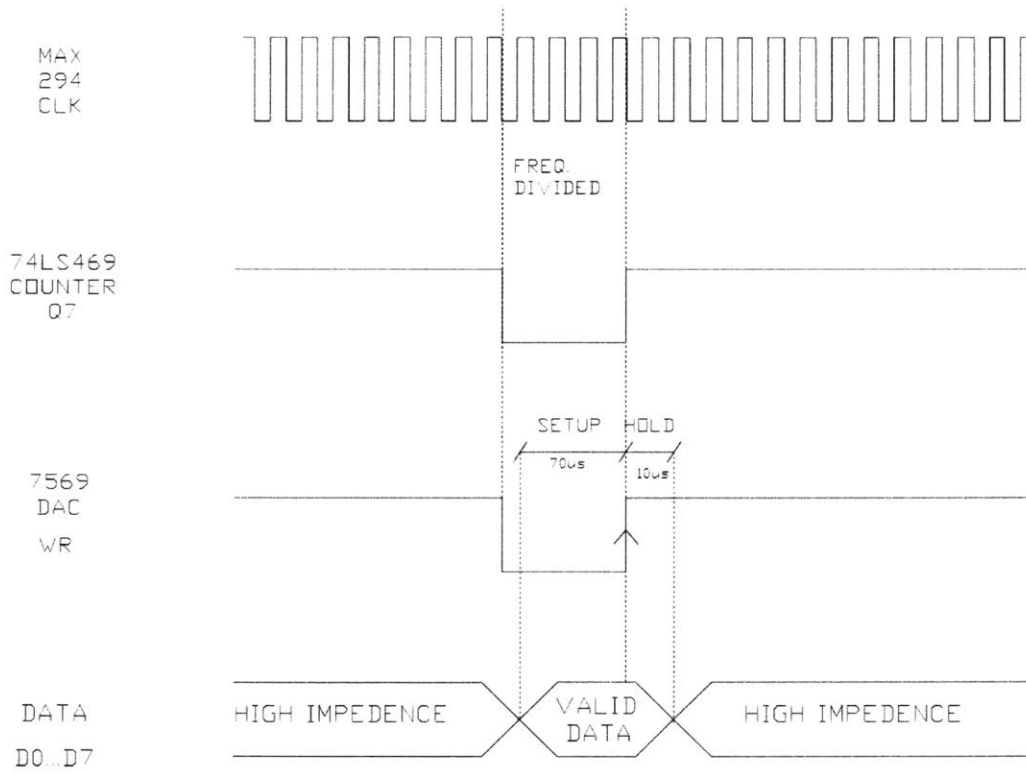
The screen shots on pages 98 and 99 visually explore the range of these parameters and how they transform an input image.

IV. RENDERING: PREPARED CRTC



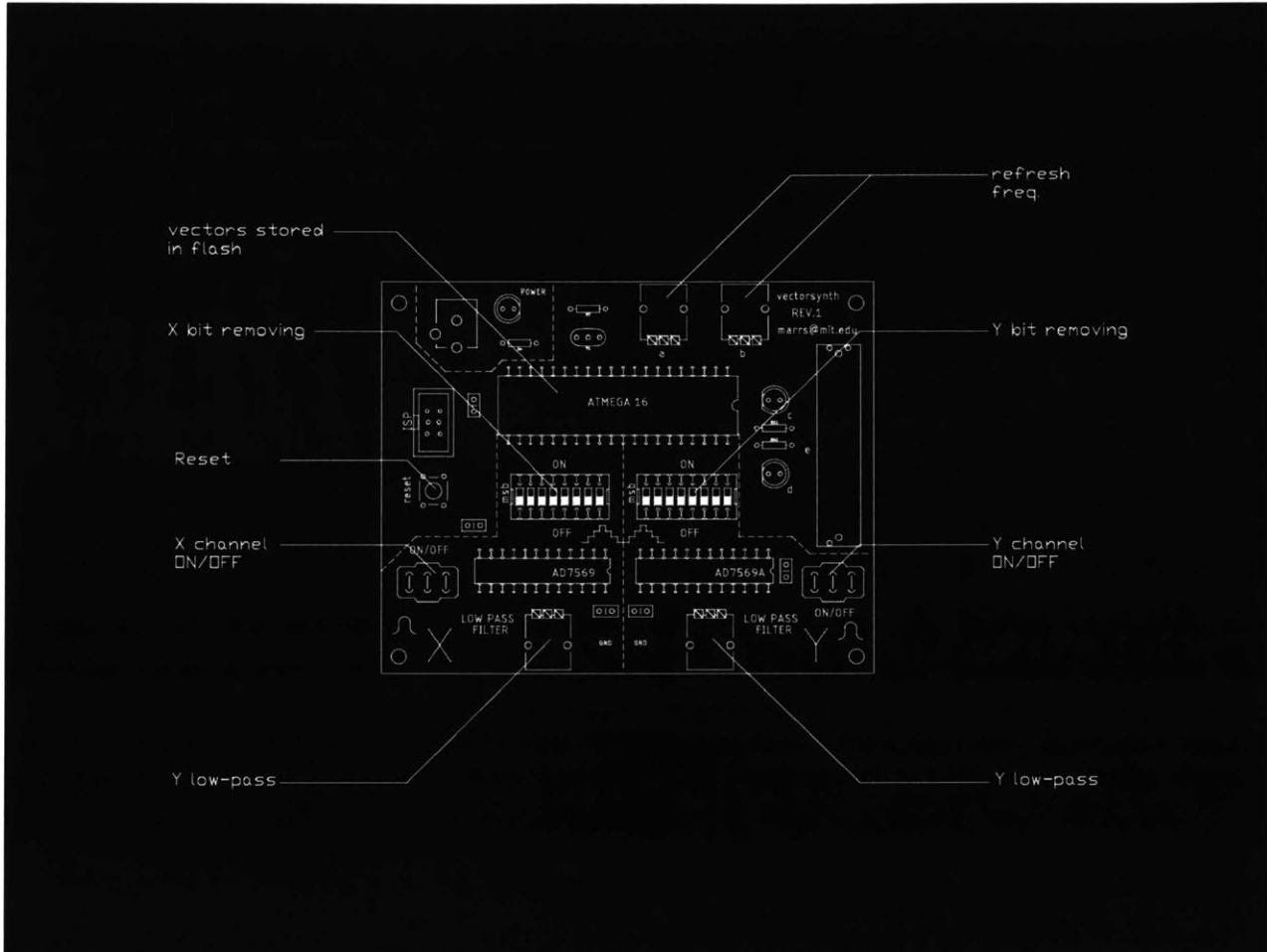
CRTc1 Circuit Diagram. Based on Thomas Haye's *Learning the Art of Electronics* circuit for CRT XY plotting control.

IV. RENDERING: PREPARED CRTIC



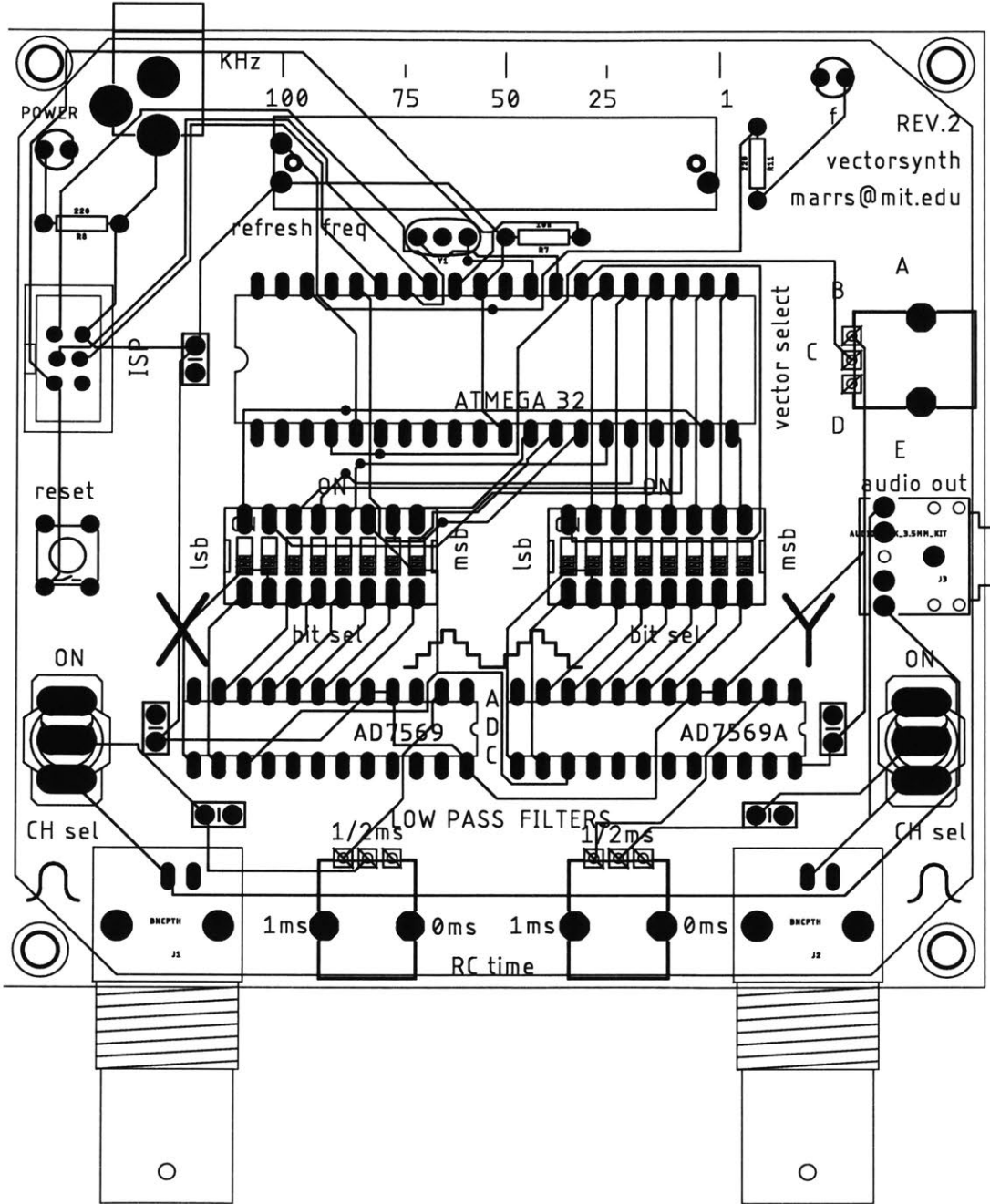
CRTIC1 Circuit Timing Diagram

IV. RENDERING: PREPARED CRTC



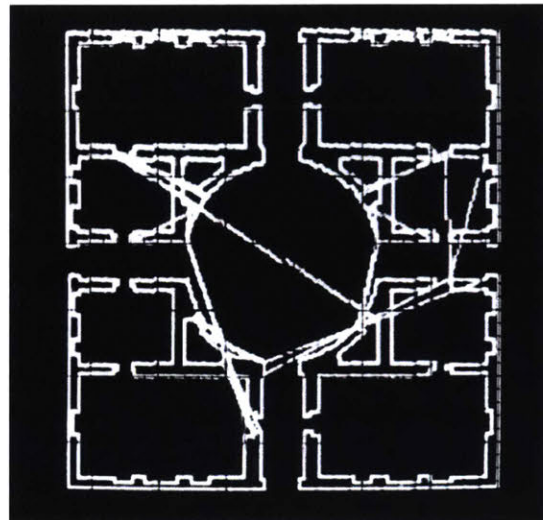
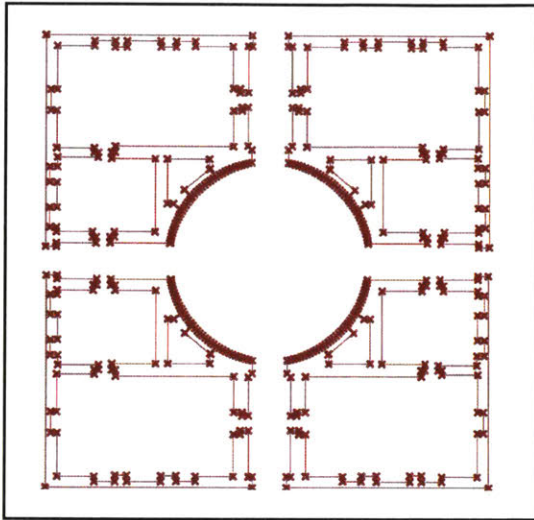
CRTC1 Circuit Function Diagram

IV. RENDERING: PREPARED CRTC



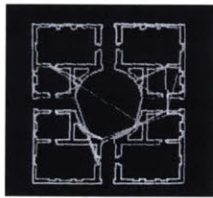
CRTC2 PCB design

IV. RENDERING: PREPARED CRTC

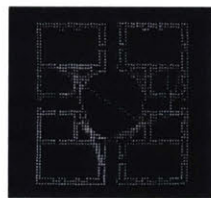


Selecting Points in Grasshopper

Output on Oscilloscope



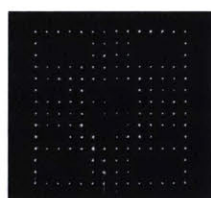
8 bits



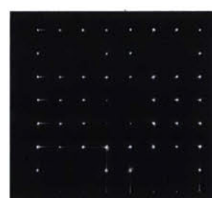
7 bits



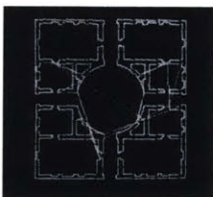
6 bits



5 bits



4 bits



0 ms



.25 ms



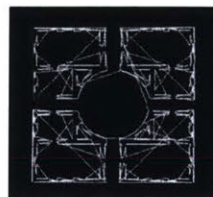
.5 ms



0.75 ms



1.0 ms



n=1



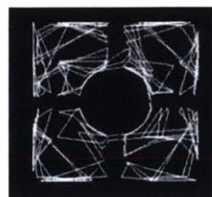
n=2



n=4

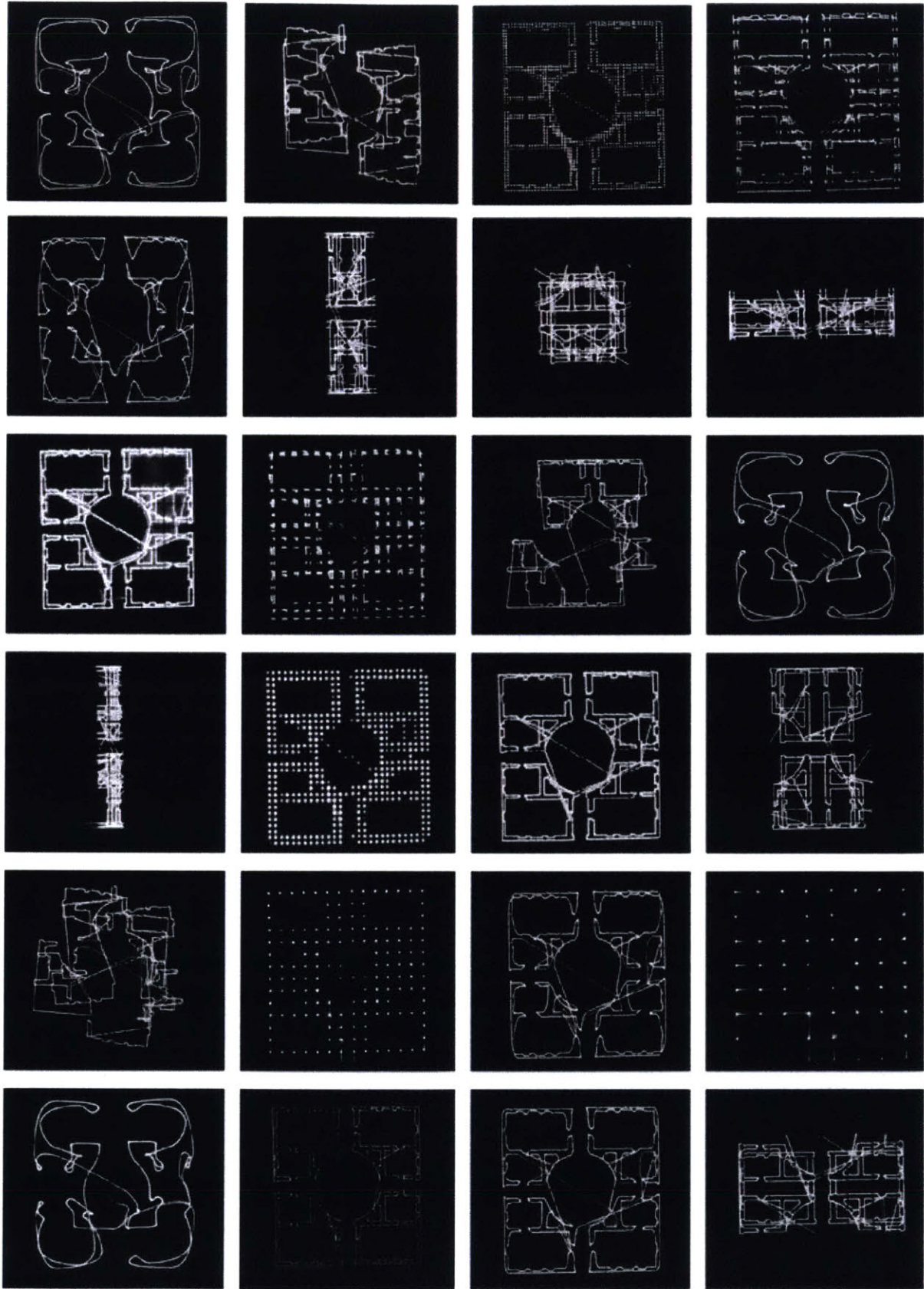


n=6



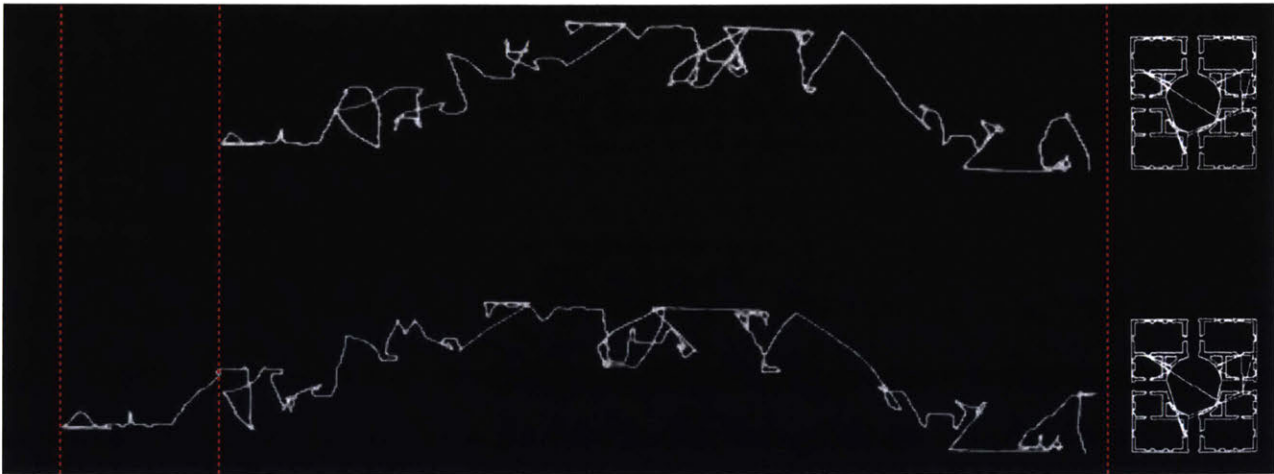
n=8

Removing Bits from ADC (top), Changing Low-Pass Filter RC (middle), changing degree of interlace (bottom)



"Entoptic Phenomena" of the CRTC itself: varying display parameters

IV. RENDERING: PREPARED CRT



Changing the number of points in the displaying of an image (Palladio's Villa Rotunda Plan) on the CRT. Fewer points (top) and more points (bottom) being used to describe the same image.

While difficult to discern looking at the CRT screen with the naked eye, when time-lapses of the same image (Palladio's Villa Rotunda Ground Floor Plan) described with different numbers of points (fewer at the top, more for the bottom) are placed side by side, one can see that display of the bottom image takes longer (because of the increased number of points which need to be output to the screen).

IV. RENDERING: PREPARED CRTIC

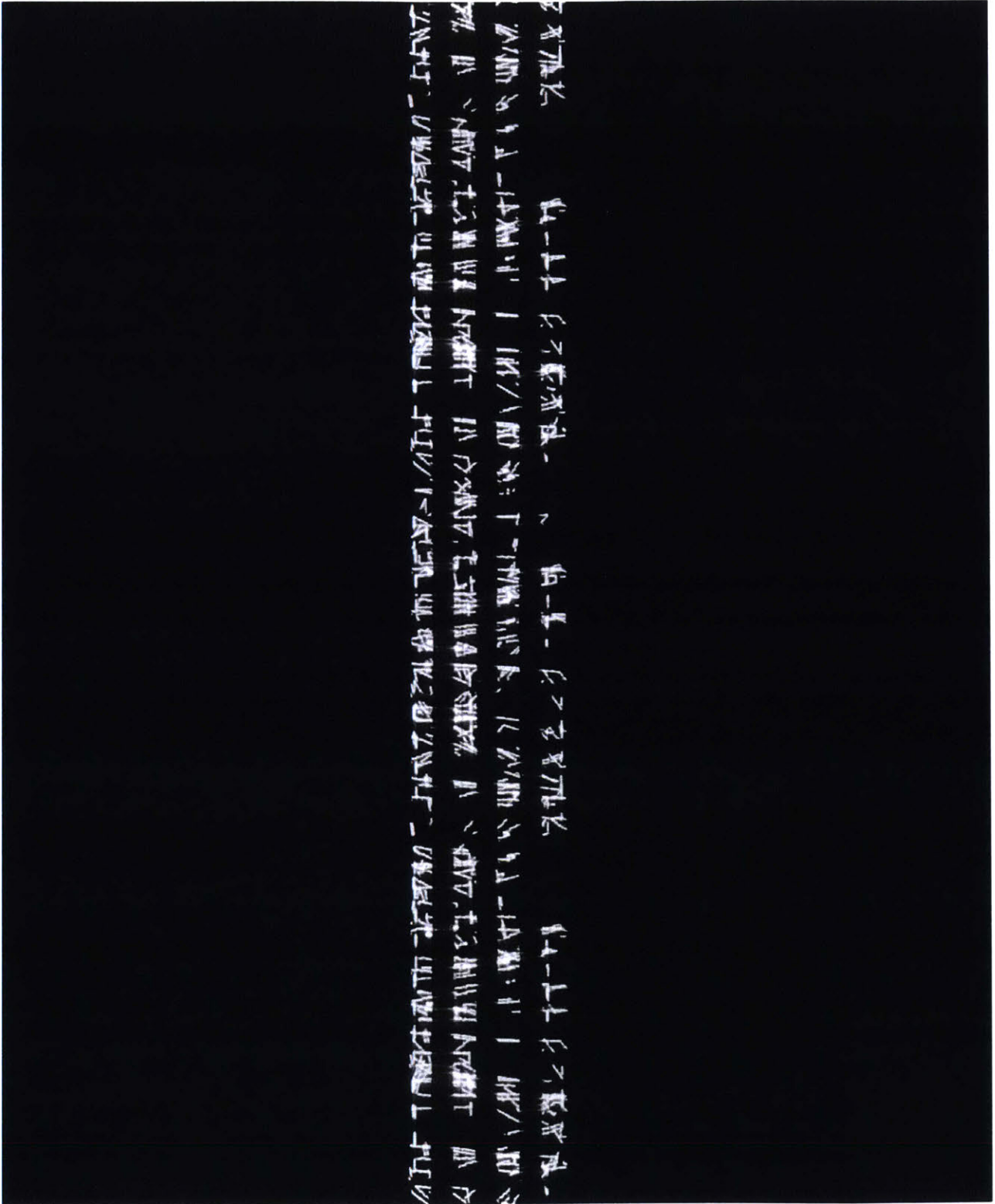
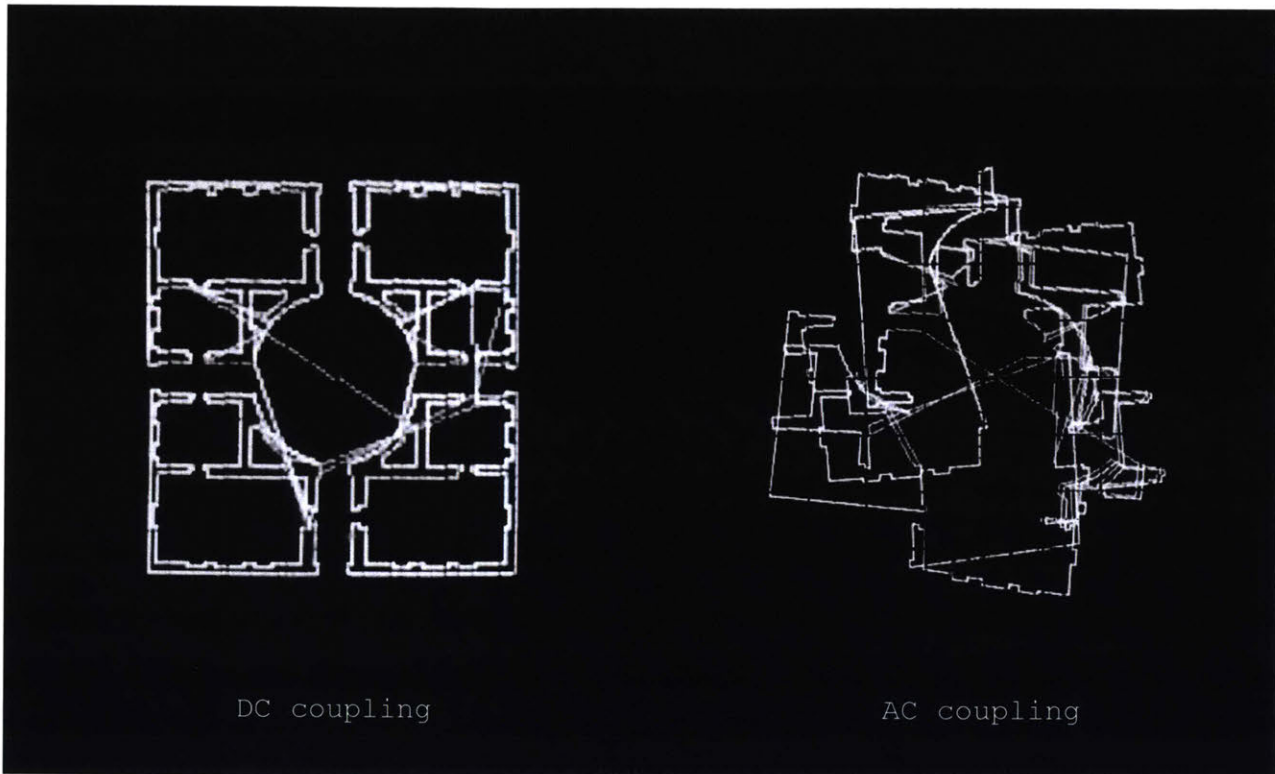


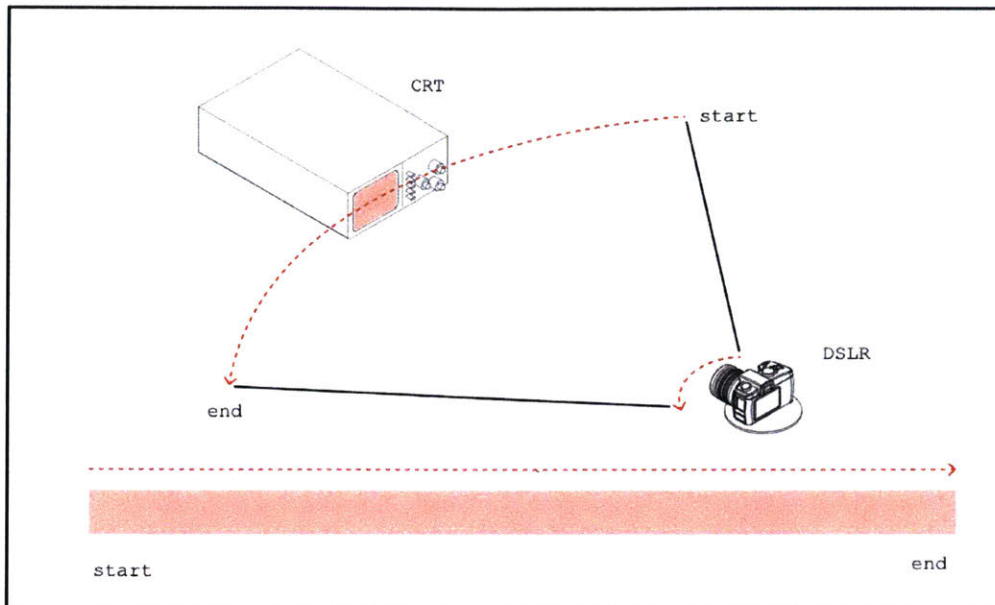
Image deconstructed into language (of bits): Palladio Villa Rotunda Floor Plan time-lapse with middle bits on CRTIC1 removed.

IV. RENDERING: PREPARED CRTC



Villa Rotunda with AC coupling on the X and Y channels (on the right) and without AC coupling (on the left). The AC coupling settings effectively discard any voltage levels that are not quickly changing (i.e. no horizontal or vertical lines are drawn).

IV. RENDERING: CRT SERIES No.1



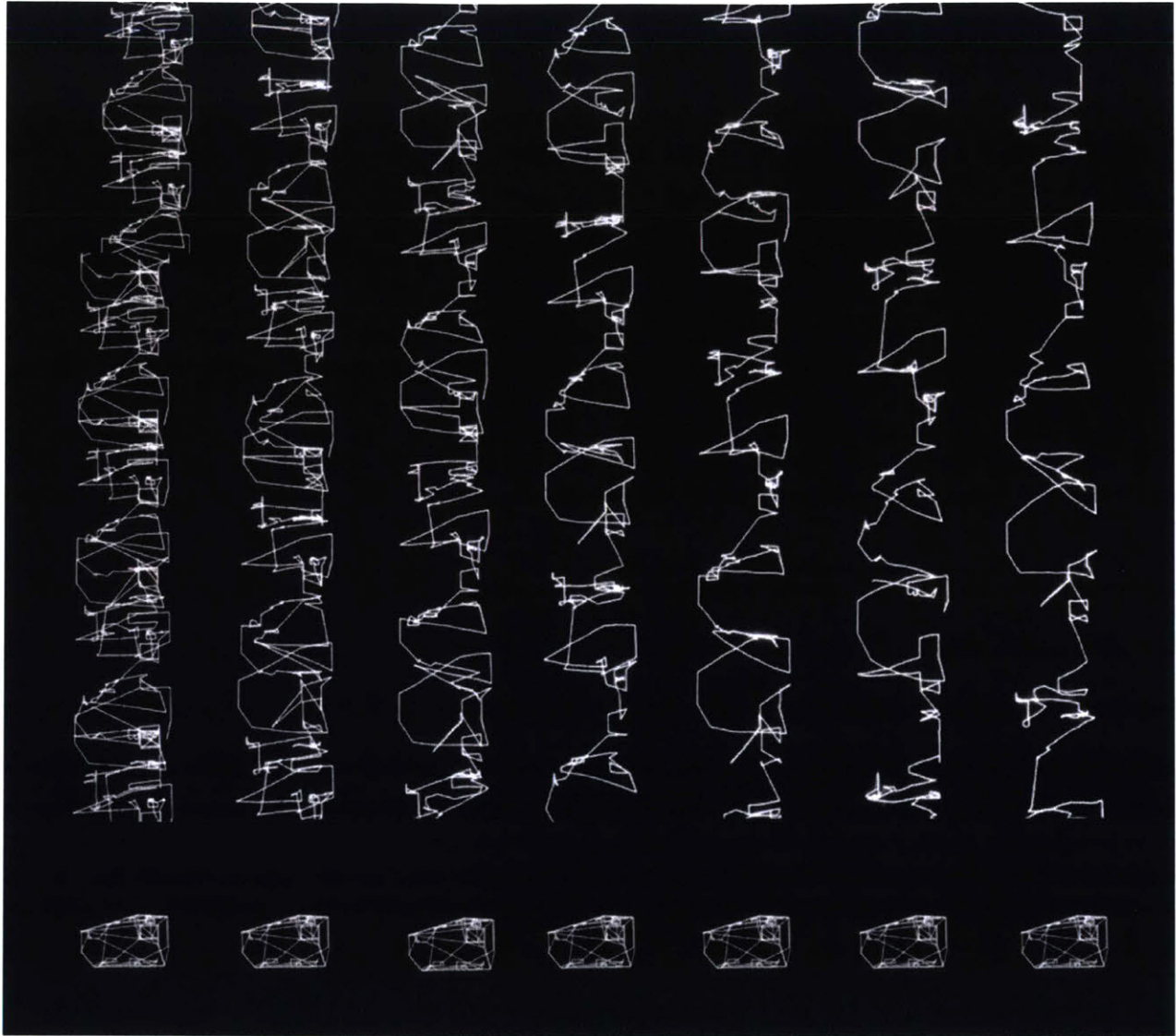
Time-lapse camera setup

By placing a camera atop a computer-controlled spinning platform and using long-exposure settings, it is possible to create time-lapse images of the CRT display sequences.

Unlike the plotting and meshing processes explored in the previous two sections, rendering is a process which never ends; unless the image is constantly redrawn it decays from the screen and is no longer visible to the observer. As a result, scanning sequences which are captured are part of an endlessly looping cycle, and their cyclical nature becomes the main artifact captured by the time-lapse image. In contrast to the previous two representations of computer time, this one is continuous: the camera moves in a single sweep to record the CRT and the CRT traces a continuous line without breaking it to relocate to the next point location.

By feeding the same image into the CRT and varying the refresh frequency, while maintaining the same long-exposure settings on the camera, we can represent different speeds of CRT refreshing and compare them to one another in a still image.

In the following drawings, the time in microseconds (μs) refers to the delay between each Analog to Digital conversion (one for each pair of coordinates) that is initiated by the CRT.



01

02

03

04

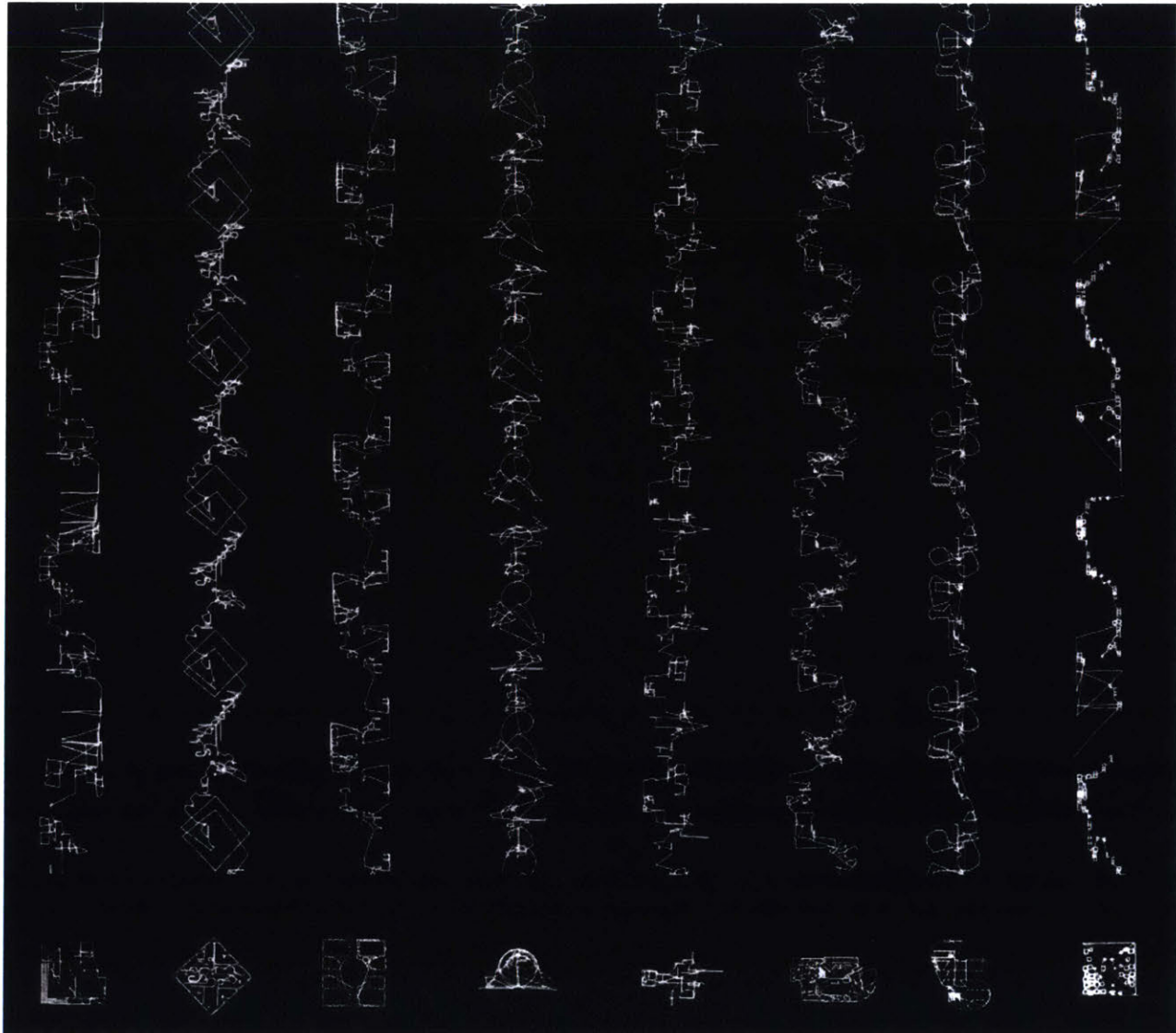
05

06

07

01 50 μ s
02 75 μ s
03 100 μ s
04 125 μ s
05 150 μ s
06 175 μ s
07 200 μ s

1A: Casa de Musica CRT Time Lapse (Elevation View)



01 02 03 04 05 06 07 08

- 01 Frank House
- 02 Diamond House A
- 03 Villa Rotunda
- 04 Cenotaph for Newton
- 05 Brick House
- 06 Wolfsburg CC
- 07 Carpenter Center
- 08 Zollverein School

1B: Assorted CRT Time Lapse at 100 μ s

IV. RENDERING: CRT SERIES No.2

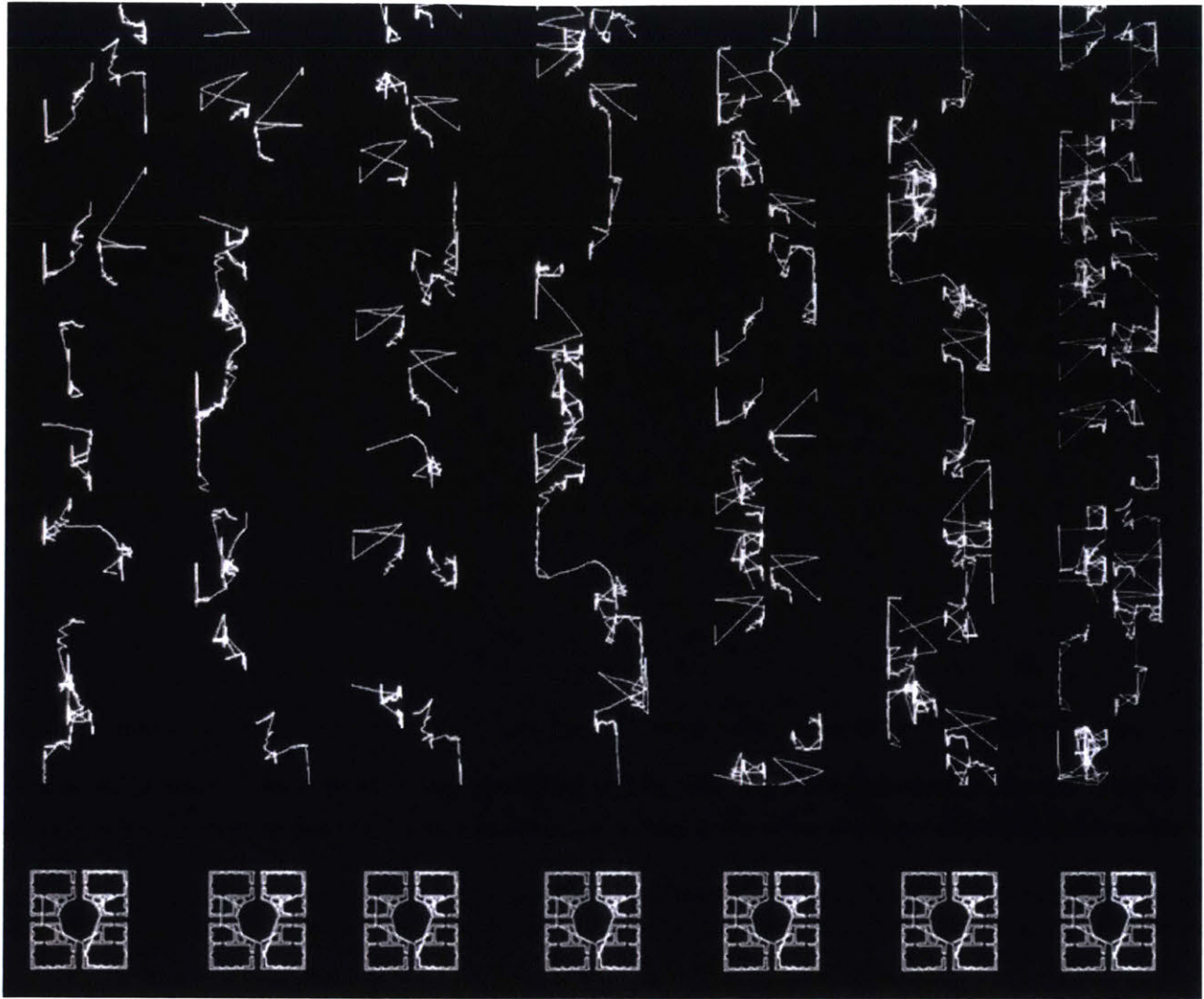


Figure 5.1 from Ivan Sutherland's Sketchpad Dissertation

In this second Rendering series, I attempt to reproduce Ivan Sutherland's "twinkling" display. Twinkling involves randomizing the sequence in which the CRTC sequences the drawing commands. Sutherland explains his method:

"Early display work with the display file led to the discovery by the author and others that if the spots were displayed at random, a twinkling picture resulted which is pleasing to the eye and avoids flicker entirely (see Figure 5.1).[...] Twinkling is accomplished by scrambling the order of the display spot locations in the display file. To do this, each successive entry is exchanged with an entry taken at random until every entry has been exchanged at least once."¹

¹ Image and quote from: Ivan Sutherland, *Sketchpad: A man-machine graphical communication system*, MIT: 1963.



01

02

03

04

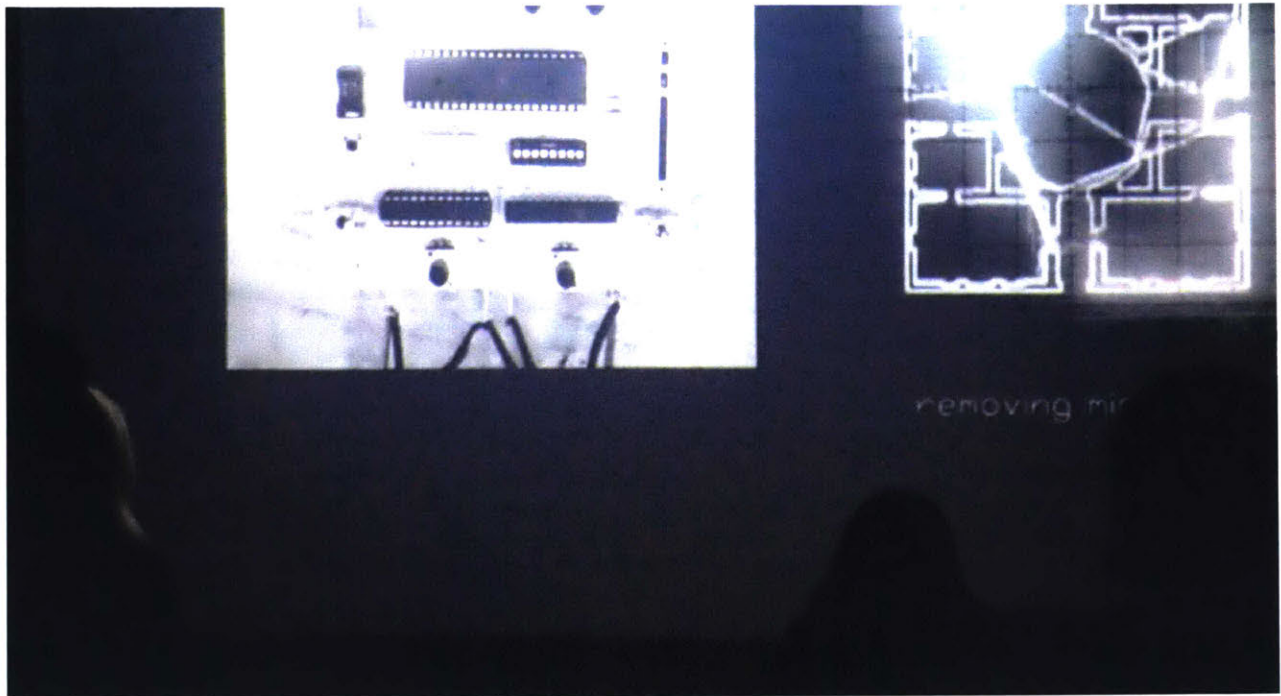
05

06

07

01 200 μ s
02 175 μ s
03 150 μ s
04 125 μ s
05 100 μ s
06 75 μ s
07 50 μ s

2A: Palladio Plan Randomized Sequence



Film demonstrating the prepared CRTCL device manipulating images in real-time. The film presents the CRTCL as a kind of visual synthesizer and conveys the idea that one can "play" or perform a display process as if it were an instrument.

Screening at the Cambridge Science Festival 2018

IV. RENDERING: CONCLUSIONS

In contrast to the time-lapse plotted drawings which were divided into drawing chunks, the time-lapse renderings are recorded continuously in a single sweep. There are no pen "lift-off" moments in the time-lapse renderings, the beam traces not only the drawing but also the path to move between different drawing elements. This gives the CRT time-lapse renderings a fluid aesthetic, illustrating why CRT vector displays are also known as "calligraphic" displays.

CRT displays are not rendered just once, like plots and meshes, but constantly refreshed at least 30 times a second, creating cyclical patterns when recorded over time. Sean Cubitt writes that in video display on modern TVs "time becomes material".¹ The same is true with earlier display technologies like the CRT. As a space to explore the temporality of computer representation in real time with physical circuits, the CRT display is ideal.

This section also explored a version of Ivan Sutherland's randomizing algorithm for the Sketchpad display. The random sequence created by such algorithms presents a challenge to the fixation with sequence employed in this thesis. Instead of revealing an algorithmic priority, the random sequence reveals the absence of any discernible priority. In the world of computer graphics, parallel computing is increasingly prevailing over sequential processing and, therefore, the focus on sequence here may also become less fruitful. New representation techniques will be needed to materialize parallel and random processes.

¹ Sean Cubitt. *The Practice of Light: A Genealogy of Visual Technologies from Prints to Pixels*.

**THESIS
CONCLUSIONS**

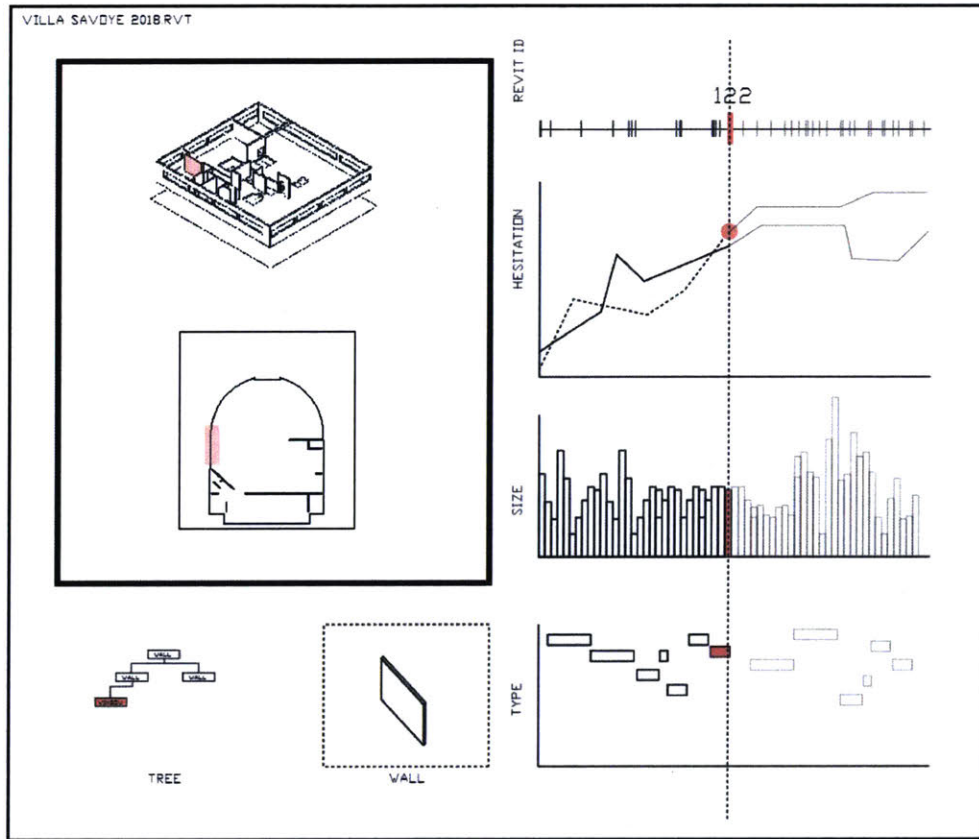
THESIS CONCLUSIONS: RECAP

This thesis has taken the premise of Zeynep Celik Alexander and John May's *Instruments Project* that representation technologies bring a positivistic world view to our field and undermine our representation traditions and the ideas built atop them. The thesis has claimed that architects can reclaim stewardship of the translation seams in the digital environment by exploiting the strategies of graphic forms of knowledge production (devising images which reflect a systems and operations) and design hacking (exploring the range of formal output in our visual technologies).

This thesis is an attempt at an interdisciplinary project and is inspired by applied media research from Media Archaeology, the History of Science and Technology, Engineering and Architecture History. Instead of a written thesis, the research engaged in here was primarily visual. The experiments in this thesis showcase a few tactics drawn from avant-garde artists, in particular: the use of 'found' objects, files and algorithms, à la Marcel Duchamp; the 'preparation' of instruments, à la John Cage; and the slowing down of time through "time-lapse" representation, à la Eadweard Muybridge, et al.

This thesis has presented three sets of experiments which materialize a plotting optimization algorithm, various meshing algorithms, and a vector display process. This thesis has suggested that the algorithms operating at these sites are critical tools, translating our files at crucial moments in the digital work flow. Along the course of these experiments, attempts were made to unpack the cultural prescriptions found in the representation technology we engage with and rethink our relationship with drawings, models and our tools by looking "through the eyes" of computer algorithms.

THESIS CONCLUSIONS: DIGITAL FORENSIC TECHNIQUES

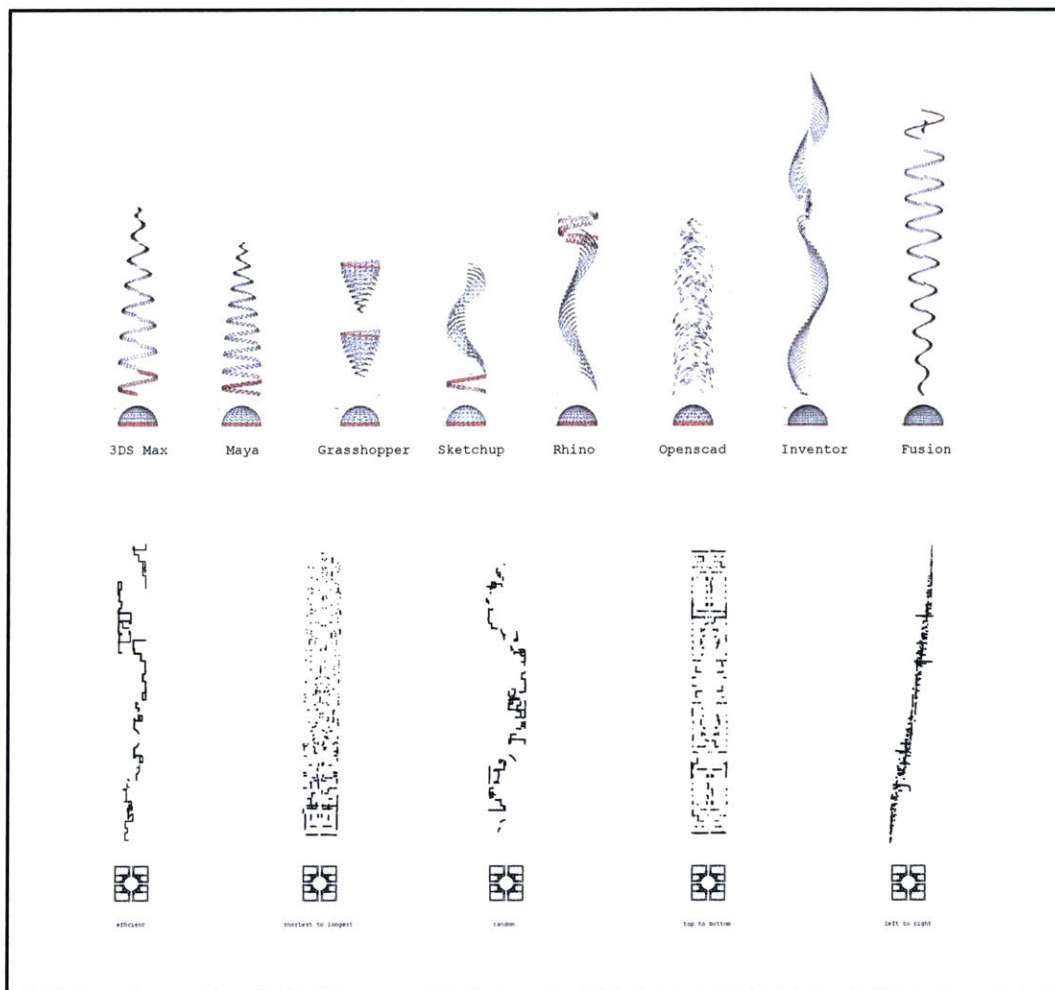


Prototype for a Digital Forensics GUI

This thesis has led to the developing of a small number of digital forensic techniques to represent Revit ID sequences, describe the data content of a file, and to identify the fingerprints of certain algorithms. Such techniques could be deployed by Architecture Historians working with digital archives to uncover the sequences in which files were drawn or modeled, in order to provide insight into the modeler's process. Such a process could reveal a new layer of information in a file, similar to how X-radiography has shown Art Historians hidden paintings.

However, there are also serious obstacles to gleaming valuable information through the index: untangling human sequences from algorithmically reshuffled sequences is challenging; modelers might often be interns instead of principles; and, gleaming intent from a sequence is not a straight-forward process itself.

THESIS CONCLUSIONS: DIGITAL FORENSIC TECHNIQUES

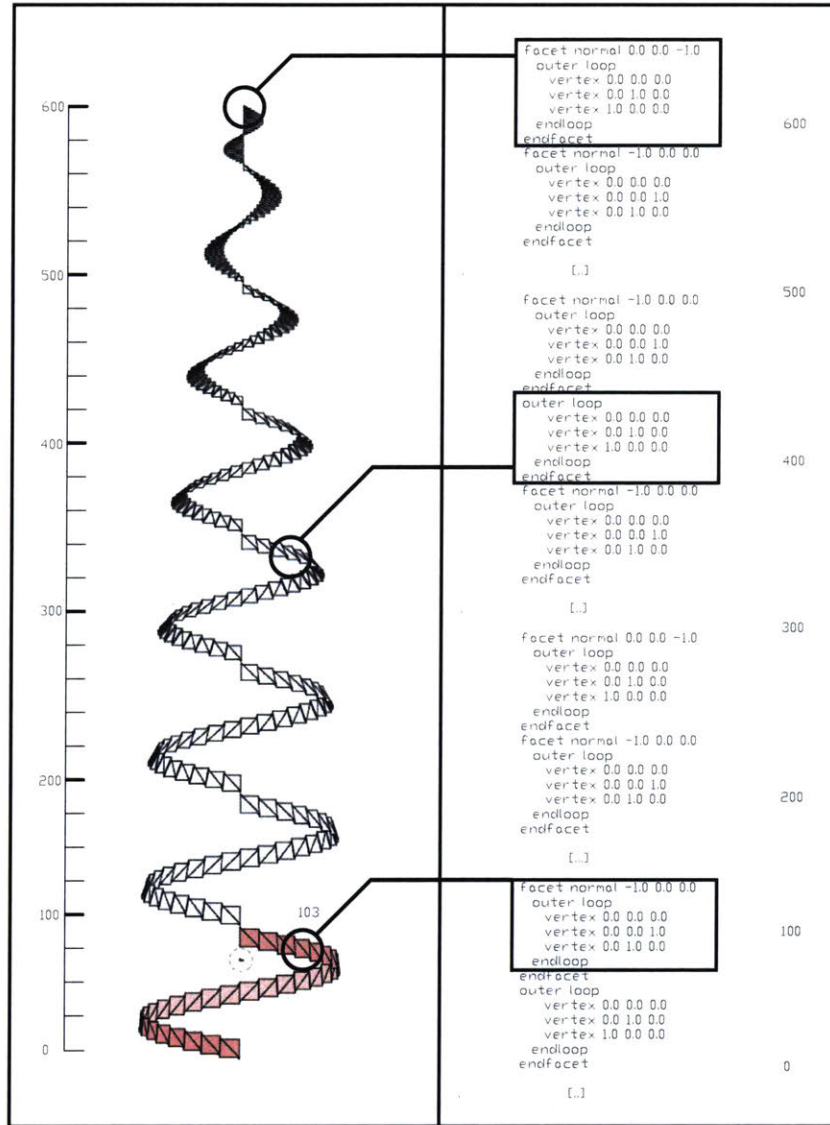


Representing algorithmic "fingerprints"

Turning our attention to the software itself and away from the users, the forensic techniques developed in this thesis could be used to study the evolution of our design software since the early CAD programs of the 1980's. How have Evans' "gaps" multiplied and spread over time in our digital environment? How have they changed in character?

Looking beyond our current coding paradigms, what new digital forensic techniques will be required of software currently being developed? As sequential computer processing, the focus of this thesis, increasingly gives way to parallel processing and AI systems which offer little to no decision making accountability, how will we be able to trace and represent computer behavior in the future?

THESIS CONCLUSIONS: INDEX CODING



Prototype for an Index Coding GUI

Just as it is possible to represent algorithmic sequences in space by mapping time to the horizontal axis, could it be possible to describe movement in space and somehow compile it into code? If such a thing were possible, it would take Grasshopper's visual programming language to the next level: into a *spatial* programming language. Such a program could make programming accessible to Architects who work and think primarily in three-dimensions.

THESIS CONCLUSIONS: CURATION AND CONTAMINATION

At several times during the course of my thesis work I became aware of the fact that I had inadvertently altered the results of my experiments. This sometimes took the form of curation: in deciding that some algorithms were more deserving of materialization than others. For instance, I drew the conclusion that CRT raster scanning and 3D printing slicer algorithms represent data in an unsurprising way: slicing is done in a particular order (bottom to top) because of the laws of gravity, while raster imaging already appears to represent the display buffer behind the screen. A fuller picture of Evans' "gap" space would include representations of these algorithms too.

At other points in my research, I inadvertently created visual artifacts through my own preparation of images. For instance, when I cut spheres into hemispheres for my hemisphere comparison image, I realized (with the help of several software engineers, of course) that I had altered the triangle index of the hemispheres by editing their equatorial mesh, taking the triangles that were sliced and throwing them to the top of the "stack". I had tainted a "visualization" of the meshing process and instead added evidence of my own mediation.

I also curated the types of images and 3D models I used as 'found' objects for the time-lapse prints, models and photographs. Many time-lapse models and images did not sufficiently deviate from their input images and were therefore not included in any shows or presentations.

Clearly, any explorer in Evans' gap space sees what they want to see; the time-lapse representations reflect my own subconscious just as much as they reflect any purported "computational imagination" I can claim to have discovered here. This problem is compounded with visual media; like graphesis, Drucker's notion of materially-embodied knowledge expressed graphically, the results of these experiments are ambiguous and open to interpretation.

However, I am not alone in finding and highlighting evidence of the chaotic aspects of Evans' gaps. Clement Valla, David O'Reilly and Erin Beseler, already mentioned in the Precedents section of this thesis, have come to similar conclusions. Whether our aesthetic reflects the current zeitgeist, what could be called a "post-digital" return to materiality, or the true nature of a part of the computation environment itself is up for debate.

THESIS CONCLUSIONS: COMPUTER VISION

Instead of a space of banal, machinic operations, that one might assume exists in the back end of a computer, the time-lapse images and objects produced in these experiments cluster in asymmetrical compositions, conveying a spatial and formal language of instability and entropy akin to Evans' description of the "gap" which exists between a drawing and its object.

One main explanation for the apparent absurdity of the time-lapses comes from the fact that computers (when operating in a sequential fashion, at least) read our files like we read a book, one line at a time,- they "experience" a model as a narrative rather than an object existing at one and the same time. When we are invited to see our models in a similar way, by stretching them in time and seeing them as a sequence of events, the oddity of such a perspective becomes immediately apparent to us.

There is a gap between the signified and the signifier, and a gap between building and drawing elements, which is accentuated by the computer. The computer can work with formal syntax, but does not engage the *semantics* of the image or the model components. To Revit, a penitentiary and a dashed line are equally weighted in the program's object ID list. As a result, the political dimension of construction and typology is covered over by the logic of the developer's spread sheet.

Computers work at a scale of time which is orders of magnitude faster than that of humans. Human time is geological in the context of digital electronics. When we push a button which is being polled by a microchip, the microchip may see the button turn on and off several times as if the switch is "bouncing" before the mechanical contacts finally settle into an "on" or "off" state. The computer's state of "awareness" is, from our perspective, extremely hyper-active. Asking the computer to discard information that appears not to be changing, essentially tuning an image to the "attention span" of a computer, also leads to absurd reimaginations of form. This is seen in Palladio's Villa Rotunda Ground Floor plan shown with A/C coupling mode on page 102, where the oscilloscope discards voltage levels which do not appear to be changing (straight horizontal or vertical lines beyond a certain minimum length) and thereby distorts the image drastically.

The software programs and systems I have been investigating are Frankenstein's monsters: they were built by a group of people working in a particular organization of labor. This organization of labor is evident in the hemisphere meshing comparison experiment, which shows various software programs doing the exact same task in different ways, despite the fact that most of these software programs are now owned by one software company (Autodesk). In my reading, the software environment depicted in the hemisphere comparison diagram appears unstable because it is a reflection of the capitalist system that created it.

The view that software is more wilderness than hyper-rational city grid is supported by some software engineers and software scholars.

Media Archaeologist Casey Alt shows that the structure of coding languages has undergone various paradigm shifts since the early days of software development. Alt describes how Machine Language, a sequential and linear code structure, gave way to Assembly and then FORTRAN, a code environment which allowed for non-linear jumping and focused more on organization of code portions than a defined sequence. Object-Oriented software then brought a decentralized coding environment where separate code entities communicate in a kind of multi-dimensional code space. Alt's analysis shows that the structure of the software environment has changed with time and led to a growing quantity and complexity of micro-level machine-machine interfaces within software processes.¹

Princeton CITP fellow Zeynep Tufekci shows that some software today is old enough to be "multi-layered," involving different generations of code patched together from as early as the 1960s. Tufekci argues that the exigencies of finance and the culture of start-ups are largely responsible for this situation. In order to cut costs, most proprietary software is not maintained (debugged) on a regular basis. The current culture of Silicon Valley, which promotes quick development of software to exploit initial customer interest, leads to hastily assembled code which is often patched over imperfectly at a later date. What can result from these trends is a series of stop-gap software solutions (digital duct-tape) and a situation where no single individual can understand a piece of code in its entirety. Tufekci's research

¹ This paragraph cites: Casey Alt in "Objects of Our Affection: How Orientation Made Computers a Medium." in *Media Archaeology: Approaches, Applications, and Implications* (2011), 292.

reveals that the ad-hoc coding typical in today's software culture combines software paradigms from different eras with results that can be difficult to predict.²

It is also possible, however, to examine the same evidence of time-lapse plots, meshes and renderings produced in this thesis and come to the opposite conclusion: The degree of order in Evans' "gap" spaces is the most notable characteristic. The spiral form in particular reappears in different indices, like a kind of DNA helix lurking in the background. One's conclusions about the apparent chaos or order of Evans' gap spaces may say more about one's worldview than the actual nature of the computational environment in question.

² This paragraph cites: Zeynep Tufekci. "Why the Great Glitch of July 8th Should Scare You." Medium.com.

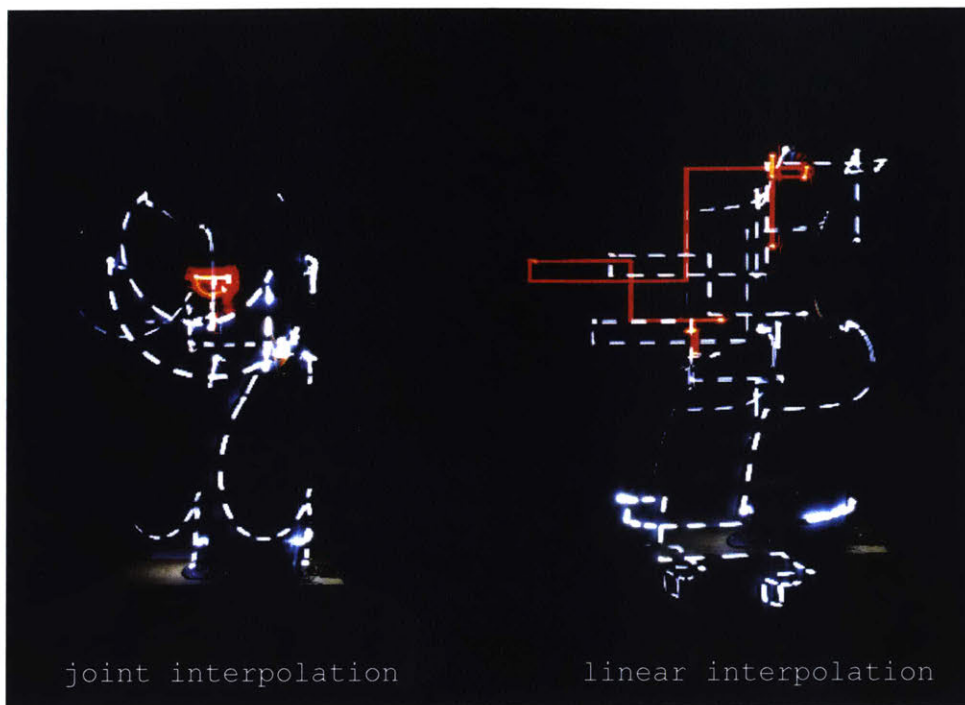
THESIS CONCLUSIONS: ANTHROPOTECHNICS

In investigating computer indices, it becomes clear that human and machine actions are increasingly intertwined today. As a result, tracing and isolating the actions of an individual by looking at a computer's database is a problematic operation when investigating Revit Object ID lists, or found digital drawings and objects. The activity of tracing digital footprints, known as Digital Forensics, is increasingly pertinent today. Police Investigators, Financial Auditors and Digital Archivists are required to establish digital "chains of custody" in order to do their jobs. We appear to have entered a new sphere of anthropotechnics where the influences of humans and machines are increasingly interdependent and where the work of representing their traces is increasingly necessary.

In his study of MPEG compression codecs, Media Theorist Sean Cubitt reveals the biopolitics in this seemingly apolitical video format. In contrast, the investigations in this thesis were perhaps at such a low-level that they may have appeared to be merely technical in nature. However, the activity of de-black boxing, looking inside proprietary systems produced by industrial processes and protected by intellectual copy write, is itself a political activity. Further, the algorithms studied in this thesis are clearly political in their applications: shortest path solvers are descendants of Fordist assembly line studies, which optimized human motion for industrial productivity; Meshing algorithms have roots in the surveying techniques of British imperialists who mapped in order to exert control over territory; and, CRT screens were used early on to display the locations of planes detected by radar.

Navigating the computational environment, one also realizes the limits of technological intimacy,- sometimes it is physically dangerous to get one's hands "inside the machine." For instance, the Tektronix 222 Oscilloscope CRT used in part of this thesis for the Rendering section has circuits which run at 1.6KV along with the usual deadly 120VAC from the wall outlet. The vacuum tube contains noxious gas and is made from glass.

THESIS CONCLUSIONS: FUTURE PROJECTS



Two time lapse photographs of a Universal Robot Arm, white FLEDs are located on each joint (shoulder, elbow, wrist) and a red LED is located on the end effector.

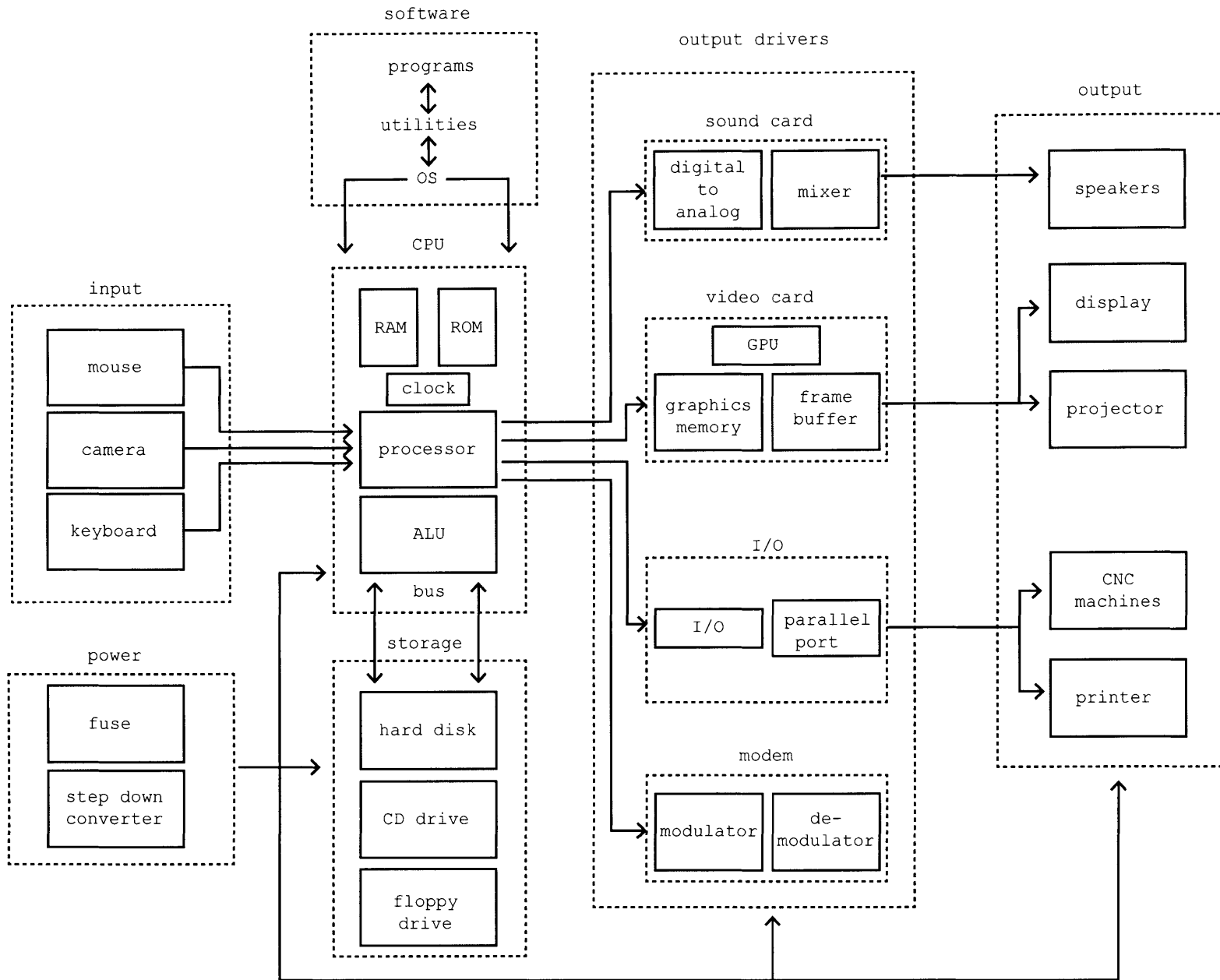
The same approaches employed in this thesis could be brought to bare on the technology of architectural *construction*. The prescriptions and politics embedded in the nail, the screw, the dry wall sheet itself, could become (and indeed are already, for certain architects) a site for experimental exploration.

Tools associated with construction could also be represented in similar ways as the algorithmic processes represented in this thesis. Architects typically ask the robot arm to move tools along paths, and the arm uses its inverse kinematic solver to achieve the necessary angles over time. However, if we were to move joints themselves, by so-called joint interpolation, we would discover resultant end-effector paths which relate to the way the robot arm is physically embodied. What kinds of forms could result from exploring the way a robot arm "wants" to move?

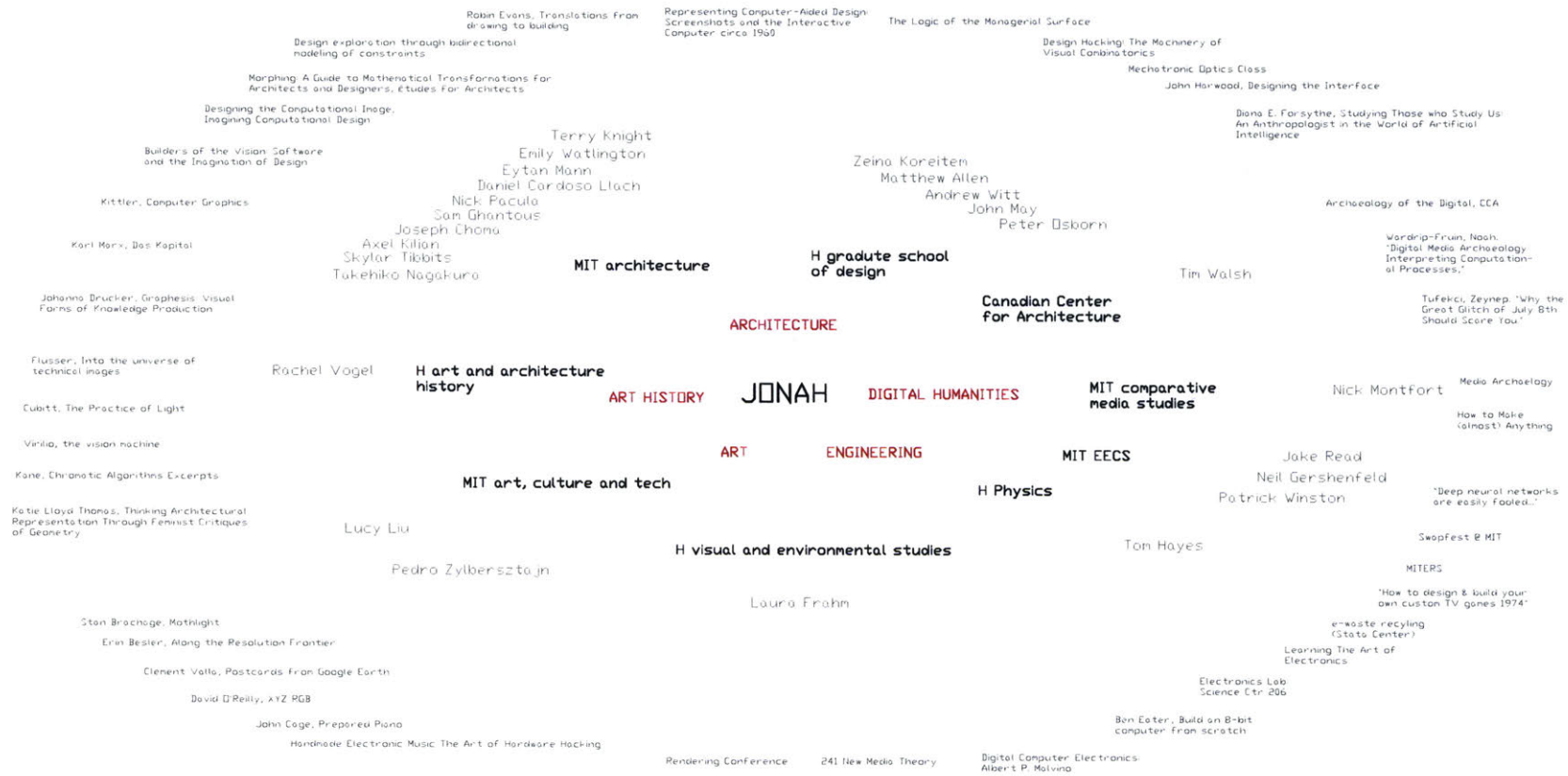
APPENDIX: COMPARISON TABLE OF EXPERIMENTS

	plotting	meshing	rendering
system medium	mechanical	software	electronic
atomic element	line	triangle	point
index	gcode	STL	display list
sequencing algorithm	Traveling Salesman	Delaunay, Marching Cubes, Poisson, Software primitives, etc.	CRTC Randomizer
materialization	time-lapse paper&pen prints	time-lapse mesh 3D resin prints	time-lapse photographs
artistic methods	found objects, instrument preparation, working in series.	found (digital) objects, corporate financing	found objects, instrument preparation, working in series
continuity vs discontinuity	discrete drawing elements, interruption discontinuous	discrete drawing elements, interruption continuous	continuous drawing process, interruption continuous

APPENDIX: MAP OF COMPUTER ARCHITECTURE "GAPS"



APPENDIX: CONTEXT MAP



BIBLIOGRAPHY

3D Systems, Inc. "Stereolithography Interface Specification." October 1989.

ABB Robotics Products AB. "RAPID Reference Manual Overview Online For BaseWare OS 3.1 Rev.1." Article No.: 3HAC 0966-50.

Alt, Casey. "Objects of Our Affection: How Orientation Made Computers a Medium." *Media Archaeology: Approaches, Applications, and Implications*. U C Press: 2011. 278-301.

Allen, Matthew. "Representing Computer-Aided Design: Screenshots and the Interactive Computer circa 1960" in *Perspectives on Science*, Volume 24, Issue 6, MIT: 2016.

Brakhage, Stan. *The Moving Picture Giving Taking Book*. Frontier Press: 1971.

Cardoso Llach, Daniel. *Builders of the vision : technology and the imagination of design*. (Cambridge: MIT Press, 2012).

Collins, Nicolas. *Handmade Electronic Music: The Art of Hardware Hacking*, (Routledge: 2006).

Cubbit, Sean. *The Practice of Light: A Genealogy of Visual Technologies from Prints to Pixels*. (Cambridge, MIT: 2014).

Evans, Robin. "Translations from Drawing to Building" in *Translations From Drawing to Building and Other Essays*. MIT: 1997. 153-193.

Fell, Mark. "Collateral Damage." Accessed from: <http://www.thewire.co.uk/in-writing/essays/collateral-damage-mark-fell>: January 2013.

Flusser, Vilém. *Into the Universe of Technical Images*. (University of Minnesota: 2011).

Harwood, John. *Interface: IBM and the Transformation of Corporate Design 1945-1976*. Univ Of Minnesota Press: Nov. 15, 2011.

Hayes, Thomas C. *Learning the Art of Electronics: A Hands-on Approach*. (Cambridge: 2016).

Heiserman, David L. *How to Design & Build your own custom TV Games*, TAB Books: 1978.

Hess, David J. "Introduction", in Diana E. Forsythe, *Studying Those who Study Us: An Anthropologist in the World of Artificial Intelligence*. (Stanford: 2001).

Kane, Carolyn L. *Chromatic Algorithms: Synthetic Color, Computer Art, and Aesthetics After Code*. (U Chicago: 2014).

Kilian, Axel. *Design Exploration through Bidirectional Modeling of Constraints*, MIT: 2006.

Kittler, Friedrich A. and Sara Ogger. "Computer Graphics: A Semi-Technical Introduction" in *Grey Room No. 2* (Winter, 2001), pp. 30-45 (Cambridge: MIT).

Knight, T.W. *Transformations in Design: A Formal Approach to Stylistic Change and Innovation in the Visual Arts*. Cambridge University Press: 1994.

Lloyd Thomas, Katie. *Lines in Practice: Thinking Architectural Representation Through Feminist Critiques of Geometry*.

May, John. "Field Notes from the Instruments Project" *JAE* 69:1 Spring 2015. 58-61.

Sutherland, Ivan. *Sketchpad A Man-Machine Graphical Communication System*. PhD Thesis: MIT, 1963.

Tufekci, Zeynep. "Why the Great Glitch of July 8th Should Scare You." <https://medium.com/message/why-the-great-glitch-of-july-8th-shouldscare-you-b791002fff03#.9z7vj43gi>. Accessed on September 1st, 2015.

Virilio, Paul. *The Vision Machine* (Cambridge: MIT Press, 1995).

Wardrip-Fruin, Noah. "Digital Media Archaeology: Interpreting Computational Processes," in *Media Archaeology: Approaches, Applications, and Implications*. U C Press: 2011. 302-322.

Witt, Andrew. "Design Hacking: The Machinery of Visual Combinatorics," in *Log 23: Fall 2011*. 17-25.

----, "A Machine Epistemology in Architecture: Encapsulated Knowledge and the Instrumentation of Design." *Candide: Journal for Architectural Knowledge* No. 03: 12/2010. 37-88.