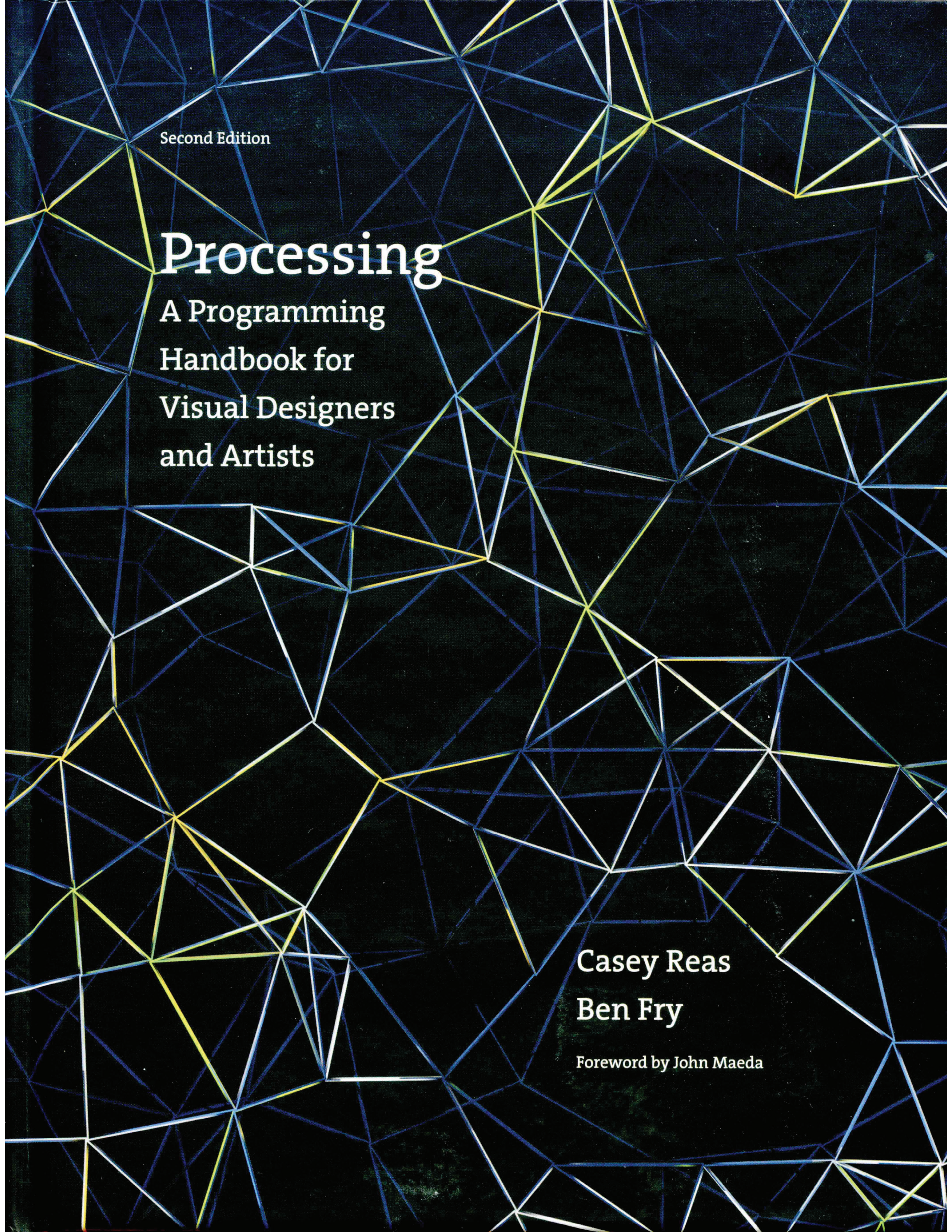


Second Edition

Processing

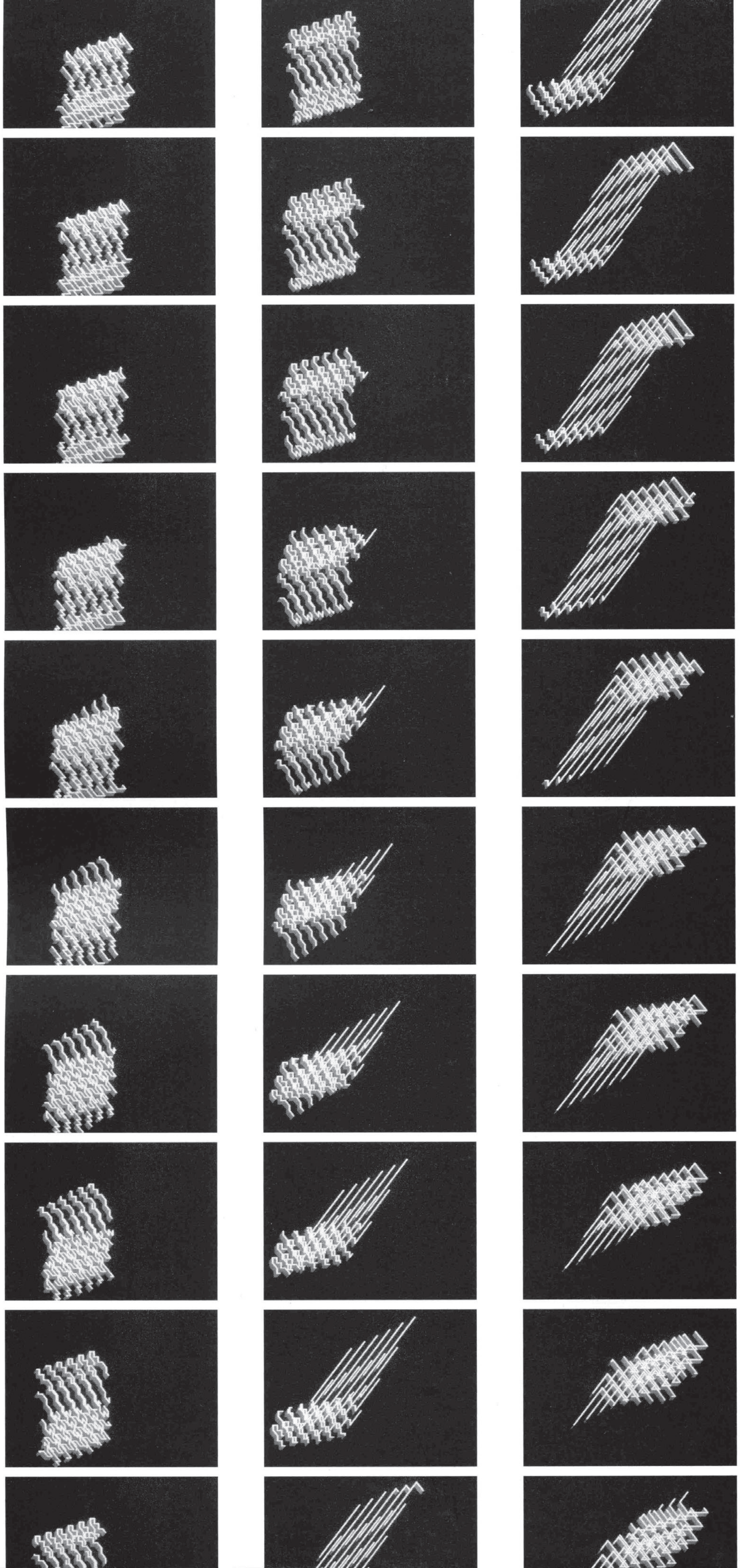
A Programming Handbook for Visual Designers and Artists

Casey Reas
Ben Fry

Foreword by John Maeda

27 Interviews: Motion, Performance

Larry Cuba (Calculated Movements)
Bob Sabiston, (Waking Life)
Golan Levin and Zachary Lieberman (Messa di Voce)
SUE.C. (Mini Movies)



Frames from the 16 mm film *Calculated Movements*. Images courtesy Larry Cuba.

Calculated Movements *(Interview with Larry Cuba)*

Creator	Larry Cuba
Year	1985
Medium	16 mm film
Software	Custom written in Z-Grass Programming Language
URL	www.well.com/~cuba/

What is *Calculated Movements*?

Calculated Movements *is a six-minute computer-animated film/video I made in 1985. It is the third work in a series of three generative animation pieces that began in 1978 with* 3/78 (Objects and Transformations) *and was followed the next year by* Two Space. *Each of these works is the result of extensive experimentation with algorithmic structure and the abstract, geometric forms that these structures generate. A simple, primitive figure (like a point or a line) traverses a set trajectory through space. As this figure is copied over itself multiple times, it is transformed in space and time producing a more complex figure that itself is then copied multiple times and transformed in space and time producing an even more complex multi-level graphic event. The trajectory and all aspects of the figures' repetitions and transformations at every level are determined by a hierarchical structure of mathematical functions and parameters. After evolving a portfolio of animation trials and software refinements, a selection of these experiments is combined into a composition.*

In the case of Calculated Movements, *the structure of the composition consists of five* movements. *The odd-numbered sections (#1, #3, #5) each consists of one long graphic event that spans the length of the scene from beginning to end. The even-numbered sections (#2, #4) are compilations of forty short events that appear throughout the scene, coming and going at different times, and sometimes overlapping. The appearance of these events is determined by a fixed on/off pattern that repeats throughout the scene. Because they are also divided into four voices with the cycle of each voice given a different on/off ratio, they beat out of phase. I worked with the composer, Rand Weatherwax, who programmed the same cyclical patterns into his Emu Emulator II for each of the four voices. The Emulator II was one of the first studio synthesizers to incorporate sampled sounds. We selected sounds from Rand's library of percussion instruments and distorted them to make them less recognizable and more abstract. Each graphic voice was then coded with its own sound. The soundtracks of the other three scenes are more melodic and were composed by Larry Simon (#1, #3) and Craig Harris (#5).*

By adding the soundtrack after the animation was complete, I avoided having an existing piece of music dictate the formal structure of the visuals. All of the experimentation was for the purpose of discovering the rules that subconsciously guide the esthetics of visual/temporal composition. Beginning with the soundtrack would have subverted that goal. The soundtrack was composed to complement the visual composition rather than the other way around. Although it would have been preferable, technical limitations prevented me from simultaneously generating both sound and image with a single algorithm.

Why did you create *Calculated Movements*?

I'd been exploring the relationship between mathematics, moving form, and perception. The possibility of generating temporal, visual compositions from mathematical relationships is what attracted me to computer animation in 1972. In the '70s, the personal computer was not yet powerful enough for the calculation-intensive work needed for my animation. Consequently, as prerequisite to creative work, I had to beg access time on the room-sized, institutional computers, a process that involved long negotiations, security clearances, paperwork, and so forth. When access was granted, it was limited to off-hours, requiring that I work the night shift.

In 1982, an affordable microcomputer, optimized for graphic operations, was developed, the Datamax UV-1. Purchased with a grant from the American Film Institute, my UV-1 finally enabled me to work at home, in my studio, on my own computer (during daylight hours!), and therefore the work on Calculated Movements *could begin. However, because the computer was slow and the need for experimentation great—over two hours worth of material was generated—the film took three years to complete.*

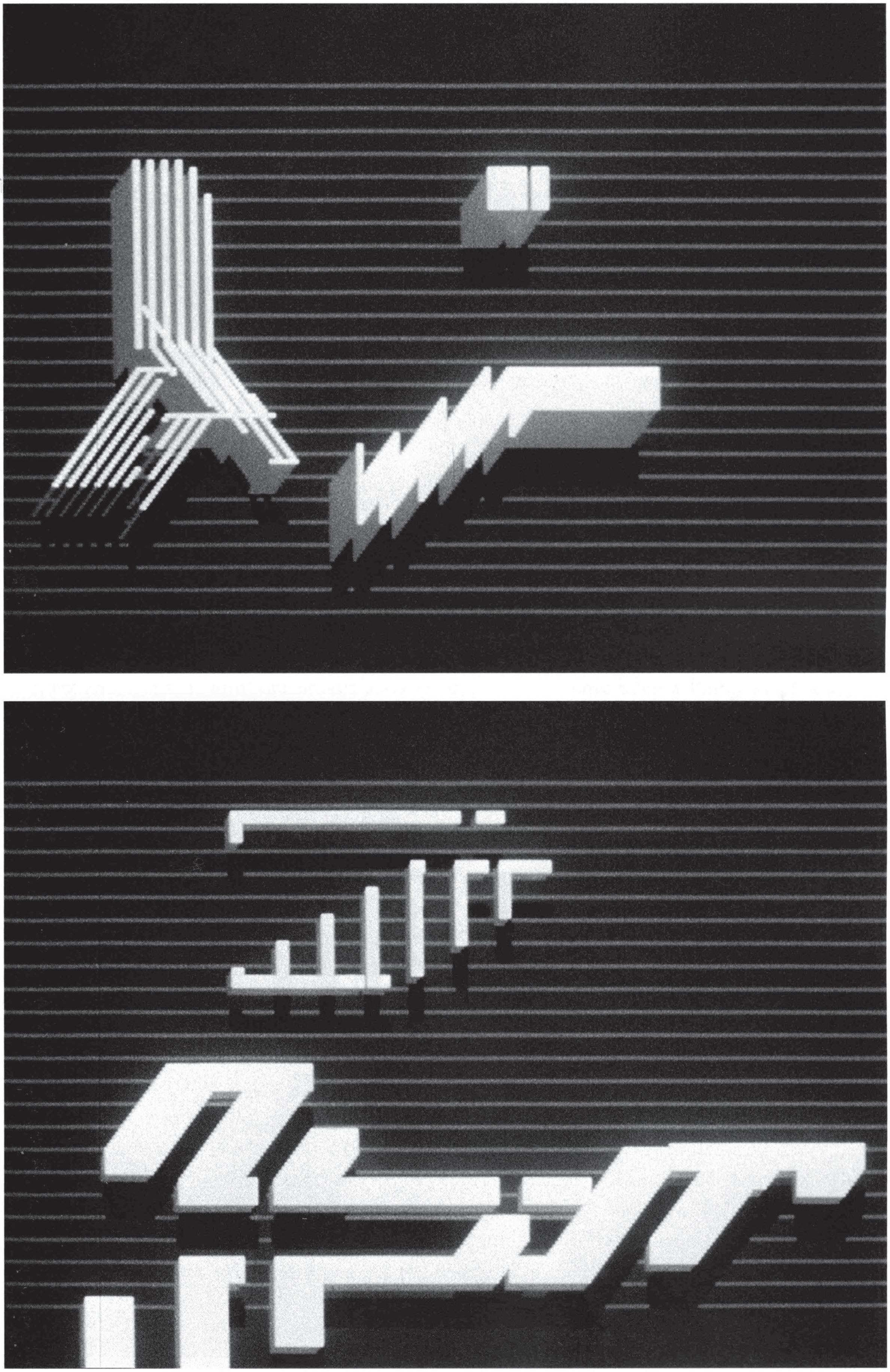
What software tools were used? Why did you use these tools?

At the time Calculated Movements was made, software tools were not the commodity they have since become. People would write programs in a language that was available on the particular computer they were using. The UV-1 computer was designed to run a single graphics programming language: Z-Grass, so named because it was Tom Defanti's GRAphics Symbiosis System ported to the Z-80 chip. Born on a much larger machine in the 70s, the GRASS language was developed specifically to make programming computer animation accessible to artists, without the need for serious technical training. The Z-Grass language continued that tradition on a smaller, affordable machine. Previously I had moved to Chicago for a year to produce my film 3/78 on the original GRASS system at the University of Illinois.

In all my work, it's been important for me to use a high-level programming language that is concise and responsive. Since the process consists of continuous experimentation, my software is in a constant state of change. For a programming language to support this approach, it must facilitate rapid development and give the artist immediate feedback. Although interpretive languages such as BASIC and Forth were appearing on the first PCs, the larger machines with the graphics hardware relied on the more technically oriented, compiled languages of FORTRAN and C, which made extensive experimentation impractical. The GRASS language was the first to deliver the power of expensive high-end graphics hardware via an interpretive programming language as simple and responsive as BASIC.

Why do you choose to work with software?

Programming for me has always been a vehicle for exploring an unknown and uncharted visual/auditory realm. Writing computer code is creating a (linguistic) pattern that generates a (visual) pattern. What is the connection between the abstraction of alphanumeric symbols and abstract form in motion? Can you develop a visual language with a symbolic language? At heart I'm driven to experiment. Unpredictable visual compositions emerge from mathematical structure. If I could picture the results of a particular process in advance, then the end would be known and running the program would serve no purpose. I collaborate with an animation-producing robot whose behavior I designed. Over time, the algorithms, the imagery, and the artist coevolve.



Two stills from the 16 mm film *Calculated Movements*. Images courtesy Larry Cuba.

computer science / programming / design

Processing: A Programming Handbook for Visual Designers and Artists
Second Edition
Casey Reas and Ben Fry, foreword by John Maeda

The visual arts are rapidly changing as media moves into the web, mobile devices, and architecture. When designers and artists learn the basics of writing software, they develop a new form of literacy that enables them to create new media for the present, and to imagine future media that are beyond the capacities of current software tools. This book introduces this new literacy by teaching computer programming within the context of the visual arts. It offers a comprehensive reference and text for Processing (www.processing.org), an open-source programming language that can be used by students, artists, designers, architects, researchers, and anyone who wants to program images, animation, and interactivity. Written by Processing's cofounders, the book offers a definitive reference for students and professionals. Tutorial chapters make up the bulk of the book; advanced professional projects from such domains as animation, performance, and installation are discussed in interviews with their creators.

This second edition has been thoroughly updated. It is the first book to offer in-depth coverage of Processing 2.0 and 3.0, and all examples have been updated for the new syntax. Every chapter has been revised, and new chapters introduce new ways to work with data and geometry. New "synthesis" chapters offer discussion and worked examples of such topics as sketching with code, modularity, and algorithms. New interviews have been added that cover a wider range of projects. "Extension" chapters are now offered online so they can be updated to keep pace with technological developments in such fields as computer vision and electronics.

Interviews with SUE.C, Larry Cuba, Mark Hansen, Lynn Hershman Leeson, Jürg Lehni, LettError, Golan Levin and Zachary Lieberman, Benjamin Maus, Manfred Mohr, Ash Nehru, Josh On, Bob Sabiston, Jennifer Steinkamp, Jared Tarbell, Steph Thirion, and Robert Winter.

Casey Reas is Professor of Design Media Arts at UCLA. Ben Fry is Principal of Fathom, a design and software consultancy in Boston. Together, Reas and Fry cofounded Processing in 2001.

"This essential book is a tool for unlocking the power of Processing. With this completely revised edition, Casey Reas and Ben Fry show readers how to use Processing for thinking, making, and doing. This remarkable software environment has opened the world of code to designers, architects, musicians, and animators. Providing a powerful alternative to proprietary software, Processing *speaks to self-education and networked engagement."*
Ellen Lupton, Director of the graphic design MFA program at Maryland Institute College of Art, Baltimore, and author of *Thinking with Type* and *Type on Screen*

"Processing has unlocked the potential of software as a creative medium by integrating a programming language and development environment and linking computation and the visual arts. This revised handbook provides expertly designed and invaluable tutorials that introduce the syntax and concepts of software and position it in the field of arts. Interviews with renowned artists give insight into the creation of their landmark software projects, illustrating how programming is applied in art."
Christiane Paul, Adjunct Curator of New Media Arts, Whitney Museum of American Art

"In addition to what you can expect—a great compendium explaining the software's features and applications—the second edition of Processing *comes with a well-curated series of interviews with artists and designers for whom software is key to their work. These exceptional insights into artistic practice contribute to the writing of history of software-based art and design and contextualize Processing in an adequate way."*
Joachim Sauter, University of the Arts Berlin, Founder and Creative Director of ART+COM

The MIT Press
Massachusetts Institute of Technology
Cambridge, Massachusetts 02142
http://mitpress.mit.edu

978-0-262-02828-8

90000

9 780262 028288