Graphics Remembrances

COLLECTED BY JULES BLOOMENTHAL

This article is a collection of reminiscences that Jules Bloomenthal put together in an attempt to capture the spirit of the past days of computer graphics.

Introduction

There are many, many fascinating stories concerning the development of computer graphics. A systematic collection would be most welcome, but for this article, I allowed myself a certain haphazardness, occasionally asking colleagues at Siggraph 1997, at work, during a telephone call, or by e-mail (my contact with Pierre Bézier was initiated by Brian Barsky during one of his visits to Paris). In all, I made inquiries to more than 60 individuals—about half of whom indicated an interest in contributing their thoughts, and a third of whom indicated a willingness to do so. Eventually, 11 contributed these essays, for which I am very grateful.

I asked contributors the following:

- to describe the context within which computer graphics developed;
- to explain how technology was transferred between disciplines, institutions, or individuals;
- to tell why they approached a problem in a certain way; or
- to describe the tools they used.

Duane Palyka and Alvy Ray Smith are two of the early connections between art and technology; they tell two very different stories of the relationship between visual imagination and the computer.

Nelson Max and Holly Rushmeier tell of their struggles with primitive tools—the former producing a computer animation in the early 1970s, the latter comparing reality with visual simulation in the early 1980s.

Rodney Stock describes his pursuit of improved antialiasing methods.

Nadia Magnenat Thalmann and Daniel Thalmann describe their pursuit of virtual characters.

The celebrated Bézier recalls the advent of CAD/CAM in a manufacturing world that operated very differently from the manufacturing world of today. Dave Rogers recounts some of the earliest work in CAD/CAM and how it came to the rescue of the U.S. Coast Guard. Bob Barnhill, Barsky, and Robin Forrest reflect on the origins of computer-aided geometric design and its interdisciplinary nature.

Although most contributors require no introduction to the computer graphics community, I offer a brief biography at the end of each essay for the benefit of other readers. Here, then, are 11 essays to shed some light on the brilliance that is computer graphics.

Jules Bloomenthal studied computer graphics at the University of Utah during the 1970s; later (much), he earned his PhD from the University of Calgary. He has conducted research at the New York Institute of Technology and at Xerox PARC and has taught computer graphics at George Mason University and the University of California at Santa Cruz. He edited *Introduction to Implicit Surfaces* (1997). He is a cofounder of Unchained Geometry, which develops tools for geometry and animation.

Jules Bloomenthal can be contacted at
432 24th Ave. East
Seattle, WA 98112
e-mail: jules@oz.net

History of Computer Graphics: Personal Recollections by Robert E. Barnhill

After receiving my PhD in numerical analysis at the University of Wisconsin in 1964, I went west to the University of Utah and joined its Mathematics Department. The next year, Dave Evans came to Utah from Berkeley and joined the University of Utah's Computer Science (then Electrical Engineering) Department. This confluence of journeys was to have significance a decade later.

I have always been interested in theoretical research with practical applications. Thus, my work and that of my students have always emphasized the more difficult higher dimensional problems: surfaces, volumes, and hypervolumes, rather than curves. To pursue these interests, in 1968 I spent a very fruitful summer with Bill Gordon at General Motors, learning about his explication of Steven Coons's surface patches. Garrett Birkhoff was a regular visitor to General Motors, and the three of us developed the first triangular Coons patch, the "BBG triangle."

During the next three years, my students Lois Mansfield, Greg Nielson, and Dick Franke and I developed solutions to multidimensional surface problems. Before my sabbatical (1971–1972), I

1058-6180/98/$10.00 © 1998 IEEE

spent six months working for Leila Bram at the Office of Naval Research, during which I organized a workshop on multidimensional approximation methods. At this workshop, Birkhoff and Phil Davis participated, and I had the pleasure of introducing Robin Forrest to the approximation theory community in the United States, since the fourth and final day of the workshop focused on computer graphics. At several subsequent meetings, I continued to introduce the subject of computer graphics to mathematical audiences.

I spent my sabbatical year at Brunel University in England, where I met John Gregory, and a productive partnership and warm relationship began. We developed two streams of thought that are still of importance: computable error bounds for polynomial interpolation and a constructive use of Boolean sums to develop new methods and explain properties of known methods, such as the *serendipity elements* of the engineers.

This meeting was a defining event in computer graphics, and many of today's experts in the field participated.

After returning to Utah from my sabbatical, I found that Gordon's student, Rich Riesenfeld at Syracuse, was interested in joining Evans at Utah. Riesenfeld and I formed a partnership that helped bring together the mathematics and computer science of the new subject. What should the subject be named? We decided that it involved the combining of mathematics with CAD, hence computer-aided geometric design (CAGD). CAGD is now known worldwide as a field of science and engineering that seeks to unify and explain theory and algorithms for curves, surfaces, and volumes that can be either free-form (designed) or representations/approximations of data from reality. Another way to view this duality is by Riesenfeld's "soft" data, which are designed, or "hard" data, which are experimentally or otherwise fixed. At an Office of Naval Research-sponsored meeting at the University of Utah in 1974, we unveiled the new name of the subject. This meeting was a defining event in computer graphics, and many of today's experts in the field participated.

In preparation for our own scientific participation for the meeting, Gregory, Robert McDermott, and I collaborated on graphics renditions of patches, which we tried on some real data. In testing our surface methods, a graphics rendering illustrated the existence of an error in the formulas, which we then corrected. This feedback approach illustrates how computer graphics can be a potent tool to the scientist/engineer. My view was that both the mathematics and graphics could be interwoven to solve practical problems, and, in particular, computer graphics enables one to make a prediction based on what is already known.

A new era in my interests began with Gerald Farin's coming to Utah in 1977. My second (and last) sabbatical, in 1979, was again to Brunel University, where Farin, Gregory, and I worked together. Spurred on by our Utah interest in developing methods for interpolating arbitrarily located data, Farin developed *Bézier triangles* for arbitrary triangles. In the early 1980s, I met Wolfgang Boehm, who was Farin's advisor in Germany, and we initiated the *Computer Aided Geometric Design* journal with the North-Holland Publishing Company. This was a very important step, because the fledgling subject, CAGD, needed both a publishing outlet and appropriate rigor in judging its publishable aspects. North-Holland gave us free rein to develop a quality journal, and the editorial board has been a who's who in the subject.

The intercontinental interest in CAGD was illustrated by the many meetings—especially in the United States, Germany, and England—as well as by the *Computer Aided Geometric Design* journal and other journals. Meetings at the Mathematics Institute at Oberwolfach, Germany, organized by Boehm, Josef Hoschek, and me, and, more recently, at the Computer Science Institute at Dagstuhl, Germany, organized by Farin and Hans Hagen, have been important over the years.

During the past 15 years, research in CAGD has diversified from the earlier searches for new methods for curve and surface design to studying the properties of known methods. One example is *geometry processing*, which includes many still-difficult problems, such as the accurate computation of the intersections of parametric surfaces and of offset surfaces. Rosemary Chang was instrumental in bringing some of these subjects, in real problems, to our attention. The application of scattered-data algorithms is just now reaching biomedical problems, such as the work by Tom Foley and his students at Arizona State University that is enabling the early prediction of Alzheimer's Disease.

There are wonderful computer graphics tools today, almost all developed by people who, one way or another, passed through Utah during its computer graphics heyday. Today's young people will experience the same thrill and insight I did when, with Nielson's help, I put up my first picture of a triangular patch on a Tektronix storage tube or later, with Frank Little, viewed a graphics rendition of a numerical integration subdivision algorithm that Lockheed used in the national interest.

As we say in soccer, "Keep it coming."

Robert E. Barnhill is one of the founding fathers of CAGD. He is now vice chancellor for research and public service at the University of Kansas and is president of the Center for Research, Inc.

Robert E. Barnhill can be contacted at
222 Strong Hall
University of Kansas
Lawrence, KS 66045-2300, U.S.A.
e-mail: rbarnhill@ukans.edu

Computer Graphics: A Personal History
by Brian A. Barsky

My interest in computer graphics began in 1973, about a year before the first Siggraph conference, although it was not until several years later that I even heard of Siggraph and attended my first Siggraph conference (in San Jose, California, in 1977). In 1973, I was an architecture student. It was apparent to me that the computer could help in the creation of renderings of buildings that were still on the drawing board. As surprising as it may seem today, such ideas were not only unusual but also unwelcome. I remember being told that computers were suitable for banking and accounting but not for endeavors that involved creativity, such as the design process. I argued that, on the contrary, the it-

erative aspect of the design process made computers particularly appropriate and, further, that interactive graphics was even more in line with the design process. I felt that rather than being a threat to the creative process (as computers were viewed at the time), computers instead could lend a helping hand (or should I say a helping "digit?"), multiplying the power and productivity of a designer and enabling him or her to develop and consider a wider range of potential designs than would otherwise be possible.

I was particularly intrigued intellectually at how computer graphics formed a synthesis of many disciplines and provided motivation and applications of many theoretical fields. I had been deeply involved in photography since childhood, and, thus, the study of and research in computer graphics provided me the opportunity to comprehend the mathematics and physics involved in modeling the photographic process. As a photographer, I was well aware of the component of the interplay of light and material, which, of course, forms much of modern computer graphics rendering algorithms research. Another component of photography is shape (form), which translates to geometry, from the scientific point of view.

I recall my delight in writing a computer program that could display a 2D perspective image of a 3D object viewed from an arbitrary vantage point.

I recall being intrigued with the geometric optics involved in the photographic process, with such concepts as the role of f-stops in depth of field and depth of focus. The mathematical elegance of the perspective projection captured my imagination. I recall my delight in writing a computer program that could display a 2D perspective image of a 3D object viewed from an arbitrary vantage point. When coupled with the notion of forming a sequence of such images to produce the illusion of smooth motion in animation and simulation, the full power of this new medium could be unleashed.

One of the aspects of computer graphics that has been particularly interesting to me is geometric modeling. Objects must be modeled in computer graphics, and computer graphics underlies computer-aided design and manufacture. Furthermore, the fundamentals draw from areas of mathematics, such as approximation theory and numerical analysis. It has been rewarding over the past 20 years to see the interplay among these various related disciplines.

My own work in introducing the idea of geometric continuity and developing the Beta-spline based on this measure was intended to provide a curve and surface model explicitly developed for computer graphics, rather than adapting methods that had been created in other contexts. My use of shape parameters had the goal of improving the user interface compared to requiring the user to specify shape through less intuitive means, such as derivatives and points that do not lie on the curve or surface.

In recent years, due to my own eyesight problem, I have become interested in the biomedical role of computer graphics. Having a cornea with a somewhat irregular shape has meant that eyeglasses do not fully correct my vision; contact lenses can provide a significant improvement, but ironically the same irregularity of the corneal shape that impedes sharp vision exacerbates the fitting process. It then occurred to me that the problem of measuring and modeling the shape of patients' corneas and the design of contact lenses to properly fit them could benefit from the ideas of geometric modeling. Furthermore, with the recent interest in surgery intended to correct vision through modification of the shape of the cornea, this topic is very current and of great interest in the optometry and ophthalmology communities.

The evolution of computer graphics over the past 25 years has been nothing short of mind-boggling. From the primitive wireframe images of that period have come images and animations with incredible fidelity. The old commercial tag line of "Is it real or is it Memorex?" is now transformed to "Is it real or is it computer graphics?" I must confess that, sometimes, I cannot be sure myself.

Brian A. Barsky has been a professor of computer science at the University of California at Berkeley since 1981. He introduced the Beta-spline, has contributed numerous articles on computer graphics, and was Siggraph papers chair in 1985. He is coauthor of *Making Them Move* and *An Introduction to Splines* and author of *Computer Graphics and Geometric Modeling Using Beta-Splines*; he also serves as editor for Morgan Kaufmann's book series on computer graphics.

Brian A. Barsky can be contacted at
Computer Science
University of California
Berkeley, CA 94720, U.S.A.
e-mail: barsky@cs.berkeley.edu
http://http.cs.berkeley.edu/~barsky/

A View of the CAD/CAM Development Period
by Pierre Bézier

Introduction

The advent of CAD/CAM has been, without any doubt, one of the major events in mechanical industry during this half century.

Describing its present state and foreseeable future would require quite an encyclopedia; hence, this text aims to show only the origins of CAD/CAM, its cause and means, especially in the car industry, and precisely in the definition, tooling, and production of car bodies.

When a system is used worldwide by millions of people, as CAD/CAM is now, one is prone to forget, or even not to have known, the basic idea behind it.

Maybe remembering a long past period will help those who are, or will be, in charge of shaping the future of industry.

Origin and Development

Initiating a radical change, and CAD/CAM undoubtedly was one, generally needs two conditions:

1) the perception of a more or less urgent requirement and

2) the advent of new means able to help solve the relevant problem or problems—the laser and the transistor have been two such means, for example.

Needs

In the mechanical industry, circa 1950, most of the very accurate surfaces were defined by lines and circles (i.e., planes, cylinders, cones, spheres, and tori); some exceptions (such as cams or gear teeth flanges) were manufactured by machine tools specially designed for that purpose: copy-grinding or gear-cutting machines. The dimensions were expressed with numbers complete with limits and included straightness, out of round, etc.; the limits sometimes amounted to 10 or 20 μm (0.4 or 0.8 thousandth of an inch) and sometimes less. Other faces were left rough from the foundry or forge; their dimensions were only approximately defined, and much was left to pattern or die makers. Lines, circles, and some templates were quite convenient in such cases, but when the field of fluid dynamics became very important, the shape of aircraft or boat hulls needed greater accuracy. In the automotive industry, the definition of a car body was entirely dependent on the taste, experience, and skill of people such as stylists, designers, draftsmen, methods workers, and tool makers. As it was, things were running smoothly, according to a tradition that, in Western Europe, was four centuries old.

Some mechanical engineers did endeavor to inject some mathematics into the process.... But this would have required a large amount of computing at a time when the only equipment was crank- or motor-operated adding machines.

Of course, the transmission of data from style to drawing office, clay building, final drawing, production master, production engineering, etc. was performed with the help of drawings, templates, and copies of the 3D master. These were not very accurate or even consistent; each operator could introduce a change, so long as it was difficult to detect and small enough so as not to alter the stylist's basic conception. This was liable to create difficulties, but, lacking a more stable definition, nothing could be done to improve the system. It was considered unavoidable, as were measles and whooping cough, though it entailed delays, haggling, and lead time.

Some mechanical engineers did endeavor to inject some mathematics into the process, translating some tricks of the trade. But this would have required a large amount of computing at a time when the only equipment was crank- or motor-operated adding machines, derived from cash registers, not to speak of Bollée or Babbage's machines. Furthermore, had a system been devised, the reaction of operators-to-be would hardly have been enthusiastic.

Looking at the situation coolly, nothing useful could be done until computing speed could be increased by five or six orders of magnitude.

Means

The computing-speed increase was triggered by some problems posed to the armament industry. The result was ENIAC, which was a powerful machine, but subject to breakdowns; it was not totally reliable; and very few companies could have afforded it. Regarding some industrial applications, it had taken part in the definition of some vital but very difficult parts, such as 3D cams and turbine foils.

After 1950, some machine tools were equipped with numerical control, but the mechanical industry was not very interested. The federal government ordered 200 of these machine tools equipped with numerical control to be lent, for a symbolic rental fee, to a few companies, hoping to jump-start that technique, which had strategic importance during the cold war. In 1955, a score of such machine tools were displayed at the International Machine Tool Exhibition, held in Chicago. The number of those machine tools significantly increased after the Chicago exhibition.

At the beginning, those machines performed point-to-point drilling, reaming, boring, or tapping; then, partial milling appeared. In fact, this was not actual CAD/CAM, but a case of preparing a simple program for a rather elementary batch of operations.

One can state that, about 1955, numerical control could be transformed into CAM, since the components were available (i.e., sensors, computers, motors, CRT, servocontrollers, machine tools, and more).

It remained, with those components, to create a complete system and build up the relevant software.

Choice

In 1960, in the European automotive industry, the accepted practice was that the Styling Department had, first, to trace sketches of future models, then develop them into full-scale drawings and small-scale (1/5 to 1/8) clay mockups. On the selected model or models, offsets were measured for a large number of points located on cross-sections; then, on a large drawing board, 2×8 meters (7×27 feet), these points were traced full-scale and, with the help of sweeps, templates, lathes, and French curves, the lines were traced and then converted into plywood templates to make the "ribs," or skeleton, of the clay model. These models were then hand-finished by highly skilled plasterers. The models could be modified at the request of the Style, Sales, or Management Department, etc. Then it was considered "frozen," as the saying was, the final expression of the shape of the "skin" of the car body. Next, the drawing office issued the drawing of each part of the body, including frame, inner panels, locks, hinges, seals, and the brackets holding the mechanical parts: power station, gearbox, steering system, axles, etc. Finally, a master model was built, to become the only standard for as long as the car would be in production.

As one can easily imagine, it was a long, difficult, and painstaking task, requiring skill, experience, taste, and imagination.

Tool engineering and manufacturing were not any easier, mainly because the definitions were expressed by drawings and by copies of the master, which was liable to warp as time passed.

To put it in a nutshell, it was a rather fuzzy and hazy process, and it yielded discrepancies, discontinuities, delays, and added costs.

Two ways are available to those who endeavor to find a solution to improve a process:

A) Consider the different steps involved in the entirety of the system and improve one item or another, taking advantage of techniques or recently developed means. For instance, consider the case for measuring offsets, tracing some curves, or milling stamping tools.
B) Forget the past, clean the blackboard, and start from scratch to build a complete and consistent solution, taking full advantage of the power of recently developed techniques, of which the computer was the most striking example.

Of course, Solution A is reasonably safe and can be adapted according to difficulties and successes met as time evolves. Solution B is much more risky, but it can yield rich results; it reminds us of "first of Cav" [the First Cavalry Regiment—ed.] and the highly respected General G.S. Patton.

This would mean that stylists would use computers to obtain drawings and 3D models, be they small or large. Top management, including the Style Department, considered this point unrealistic and insane, to say the least.

Solution A: The basic idea was to let the stylists draw sketches and build small-scale mockups. Then, the drawing office would trace curves running through measured points, and computer experts would express these curves with numbers (i.e., vector coefficients). These curves would define a net covering the clay model. Then, they would express the points contained in each mesh, or patch, ensuring the continuity between adjacent patches. The solution was invented by Steven Coons and published in *Project MAC-TR-41* (1967), but it cannot be totally automatic. It is widely used and is famous worldwide.

One important task was to get an automatic solution to the problem of running a curve through given points. The solution was invented by mathematicians such as Bill Gordon, Rich Riesenfeld, E. Cohen, E. Mehlum, Robin Forrest, C. Lang, Malcom Sabin, Andrew Armit, Gerald Farin, Robert Barnhill, Les Piegl, and Brian Barsky and, of course, by Coons. The curves were splines, but different varieties appeared: uniform or nonuniform splines, Beta-splines, Nu-splines, and, somewhat later, nonuniform rational B-splines (NURBS). Since no algorithm is 100 percent robust, piecewise curves had to be smoothed by the arbitrary decision of an operator trying to make curves "sweet," "clean," and "fair." The use of so many words for expressing the same idea generally means that the idea is not very clear; in this case, it shows that a system cannot be totally automatic.

To give credit to whom credit is due, it should be noted that, as early as 1958, Paul de Casteljau, an outstanding and true mathematician, had invented a perfectly correct system based on Bernstein's functions; but Citroën, for which he worked at that time, was very secretive, and the results were not disclosed until 1972, depriving him of his legitimate reputation and fame.

Solution B: At Renault, a small group, very small indeed, of mechanical and electrical engineers thought that throughout the process of car-body production (from the Style Department down to inspection at the exit station of the assembly line), information and data should be carried exclusively by numbers; of course, drawings and 3D models would remain useful, but only as a complement and an explanation for the benefit of operators.

This would mean that stylists would use computers to obtain drawings and 3D models, be they small or large. Top management, including the Style Department, considered this point unrealistic and insane, to say the least.

The system needed to fulfill the following conditions:

- define space curves directly and not just by couples of projections;
- be easily understood and grasped by stylists, designers, draftsmen, and methods people having a fair knowledge of elementary geometry;
- trace a curve in a matter of seconds, a couple of minutes at the most (it should be remembered that laying a lath five meters (17 feet) long with "ducks" might have taken two to four hours);
- carve Styrofoam at the rate of one square meter (10 square feet) an hour on a specially devised milling machine, the feed of which would be about 300 mm/second (60 feet/minute); and
- work in an interactive mode.

This meant that a medium-scale computer would be available 24 hours a day. Circa 1960, such a viewpoint was considered totally heterodox, since the accepted practice in a company was to have a very large computer working for different services, first for administrative services (accounting, mail, invoicing, statistics, and personnel) and then later for technical development "when time permitted."

The cost of the minimal prototype equipment (i.e., a drawing machine, a special milling machine, a simple computer, and elementary software) was supposed to amount to $500,000. Renault limited its risk to $100,000, and the project carried forth with help from a company producing medium-size computers and from D.G.R.S.T. (France's General Agency for Scientific and Technical Research).

Prototype Equipment

To design a drawing machine and a milling machine was a trivial problem for well-trained mechanical engineers; the servo-controller conditions were more exacting, but the problem was finally solved. The mathematical problem, however, remained. The company staff included some professional mathematicians who could, no doubt, solve it—but they were not interested. So, the problem was dealt with by engineers who were not bad at math but were far from "chartered" mathematicians. The mathematical solution was developed and tested with the help of a small plotter.

In 1965, the required amount of money became available, and so the manufacturing of the equipment began; it was delivered in April 1968 and then took about a semester to adjust and debug. In 1970, it was concluded that it could be used in the drawing office and in methods. Five drawing machines and as many milling machines were ordered, built, and commissioned in 1972.

During three years, the system was operated in parallel with the traditional method, so as to make a sound comparison. In

1975, it was decided to make the big change, after having looked twice or thrice.

In fact, designers and draftsmen were rapidly convinced that the system was easy to use and more comfortable. It took much more time for the staff, especially the stylists, to admit that it was really useful.

But, as Rudyard Kipling said, "That is another story."

Conclusion

The system, at that time named UNISURF, was officially accepted and since then has been expanded more than 10-fold. One can safely admit that it totally replaced the previous method; it leaves the stylists free to express their first intentions with handmade sketches, drawings, and small-scale mockups, but from that state on, any precise information or datum is expressed exclusively by numbers.

It came from the ability to work, think, and react in the rigid Cartesian world of machine tools and, at the same time, in the more flexible, *n*-dimensional parametric world.

The original software, which was limited by the power of the second- (or third-?) hand computer, has been greatly expanded; NURBS have been incorporated with other solutions.

Many systems now exist, some of which have some similarity with that of Renault. They are used the world over and in various industries.

Instead of looking at them in detail, it seems more convenient to consider the reasons for which a minute makeshift project, initiated by a very small group of mechanical and electrical engineers, has been somewhat successful.

Looking back at those years (1960–1985), it is possible to perceive some salient points:

- Mechanical engineers define shapes with dimensions and limits that leave no place for discussion; they thought that it would be a good thing to use the same principle for car bodies.
- They figured that it was urgent to do away with a centuries-old system, scrap it, clean the blackboard, and start from scratch.
- They had been in most of the trades that would be involved in the system-to-be: pattern makers, foundry and forge hands, tool setters, machine tool operators, draftsmen, designers, electricians, servocontroller specialists, etc. Consequently, they could imagine the reactions of people who would take part in the new process and could take their reactions into account. The only people they had not met—because of the need to keep new designs secret—were stylists, but body designers had explained their behavior, judgment, and, sometimes, whim and superstition.
- Any solution would require a certain amount of math. The problem was to define space curves and, then, patches. The basic idea had been to choose a not-too-complicated curve and modify it to obtain a convenient shape. Modification of a curve was accomplished not by altering the definition of the curve, but by distorting the Cartesian cube in which it was originally defined. For the sake of simplicity, the distortion of the cube would be linear (or affine). Hence, the cube would become a parallelepiped (pppd), into which the coordinates of a point would remain the same as those of the corresponding point of the initial curve.
- A pppd is defined by three vectors having the same origin. Hence, distorting a pppd requires moving only four points. The new set of coordinates (i.e., the pppd) is related to the Cartesian set of the machine through a 4×4 matrix; the operator is free to choose in which set he wants to work.
- Instead of defining a pppd with three vectors issued from the same point, it seemed simple—how simple—to put those vectors end to end, thus building an open polygon, the shape of which mimics that of the corresponding curve.
- It became evident that the new vectors were not compelled to be linearly independent. It was possible to increase the number of polygon legs, thus increasing the variety of curves available.
- Defining surfaces, or patches, as foci of parametric curves was straightforward. The whole system was expanded to provide slope or curvature continuity between adjacent patches. Now, no day passes without texts or books describing new possibilities of CAD/CAM.

To sum up the basic ideas of the system, it can be said that it came from the ability to work, think, and react in the rigid Cartesian world of machine tools and, at the same time, in the more flexible, *n*-dimensional parametric world. It is even possible to imagine that an object defined in a Cartesian 3D space could be inserted into a triparametric space that can be twisted and warped, exactly as a patch is a twisted square inscribed in a diparametric space. This notion should be handled with care, because it is liable to raise the final Cartesian definition to an unmanageable order.

Migration between two different worlds is reminiscent of the story of *Alice in Wonderland*. The mirror is the border between her sound home world and the senseless world in which she meets the Snark, the Queen of Spades, the Mad Hatter, and the Cheshire Cat.

After all, Lewis Carroll was the nom de plume of Charles Dodgson, who was also a professor of mathematics.

Pierre Bézier is one of a very few people whose name resides in the lexicon of computer graphics. He has taught, authored books, directed the Renault car company, and, of course, provided the world with the first robust method for free-form surface design. In 1985, he received the Steven Coons award, Siggraph's highest honor.

Pierre Bézier can be contacted at
2, av. Gourgaud
75017 Paris, France

The Emergence of NURBS
by Robin Forrest

I have been fortunate to participate in some significant developments in computer graphics. Elsewhere,[5] I have described the origins of the Bernstein form of the Bézier curve, the form that is now

universal, and how that led to the use of B-splines for computer-aided geometric design (CAGD). A second thread is the emergence of nonuniform rational B-splines (NURBS) as the standard curve form for CAGD. NURBS followed logically Rich Riesenfeld's work at Syracuse University[8] with K.J. Versprille's thesis on rational splines[11] and L.C. Knapp's thesis on nonuniform splines.[6]

As a graduate student in the Mathematical Laboratory at Cambridge University, I was urged to seek help on curves and surfaces from in-house mathematicians: H.P.F. Swinnerton-Dyer admitted I had a problem, but said that such geometry had been in vogue at the turn of the century but was no longer of interest, and J.C.P. Miller, the noted compiler of mathematical tables, suggested that I would find all the answers in the manual for the 1936 Brunsviga calculator. Help of a totally different nature came when I spent a summer at the Massachusetts Institute of Technology working with Steven Coons.

Sutherland used to tease me about being at the technical school down the road, until I pointed out that all his degrees were from technical schools.

When I arrived in the other Cambridge, I was somewhat surprised to find that Coons was on sabbatical leave, but as this was just up the road at Harvard University, I was reassured. Coons was working with Ivan Sutherland, who had gathered Bob Sproull, Danny Cohen, and Ted Lee to work with him. Sutherland used to tease me about being at the technical school down the road, until I pointed out that all his degrees were from technical schools (Carnegie Tech, the California Institute of Technology, and MIT). Coons had arranged for me to have an office in Project MAC next to Doug Ross's office, who led the AED Group, but my status was enigmatic, since visiting research students were in no known category. Coons solved my problem by persuading the authorities to declare me a visiting research fellow, boosting my résumé.

In the summer of 1967, Coons was working on the famous "little red book,"[1] (*MAC-TR-41*), which despite its June date did not appear until the fall. L.G. Roberts, following his introduction of homogeneous coordinates to computer graphics,[9] had demonstrated that conic arcs could be represented as rational parametric quadratic functions. At Boeing, M.S. Rowin had developed the T-conic,[10] which merged the parametric cubic curves of J.C. Ferguson[2] with the conic arcs then conventionally used in aircraft lofting. T-conics were rational parametric cubics with a quadratic denominator that enabled the homogeneous component to be controlled by a single shape factor related to the conic shape factor. This neatly avoided problems with asymptotes but reduced the generality of the curve. Coons and I spent the summer of 1967 developing the rational cubic form. My particular concern was to discover how best to specify the homogeneous coefficients that gave extra degrees of freedom in controlling curve shape. Thus, my 1968 PhD thesis:[3]

- contains recipes for defining straight lines;
- contains recipes for defining circular, elliptic, parabolic, and hyperbolic arcs in rational cubic terms;
- explores means for reparameterizing rational cubics while retaining the same shape;
- explores means for splitting curves for subsequent refinement; and
- explores the use of rational blending functions to control patch interior shape.

Lee's Harvard thesis[7] dealt with rational bicubic surface patches. Further work on controlling the shape of a rational cubic by specifying the intersection of the curve with a plane defined by the tangent points and the mid-chord appeared as a Cambridge CAD Group document in 1970.[4]

A characteristic of much of the early work in computer graphics and CAGD was the mode of publication. With few exceptions, work was reported as theses, technical reports, and company internal documents rather than in journals and at conferences. The relatively small community working in the area managed to communicate via word of mouth and by exchanging papers thought to be of mutual interest. The Report Library that I started at Cambridge University soon became widely known, and, in some cases, others used it as a repository of material for record. It is a source of regret that much of the material gathered in the 1960s and early 1970s has never been published in conventional form.

References

[1] S.A. Coons, "Surfaces for Computer-Aided Design of Space Forms," M.I.T. Project MAC, *MAC-TR-41*, June 1967.

[2] J.C. Ferguson, "Multi-Variable Curve Interpolation," *Boeing D2-22504*, July 1963. Later published in *J. ACM*, vol. 11, no. 2, pp. 221-228, Apr. 1964.

[3] A.R. Forrest, *Curves and Surfaces for Computer-Aided Design*, PhD thesis, Cambridge University CAD Group, July 1968.

[4] A.R. Forrest, *The Twisted Cubic Curve*, Cambridge University CAD Group Document 50, Nov. 1970. Later published as "The Twisted Cubic Curve: A Computer-Aided Geometric Design Approach," *Computer-Aided Design*, vol. 12, no. 4, pp. 165-174, July 1980.

[5] A.R. Forrest, "Interactive Interpolation and Approximation by Bézier Polynomials," *Computer-Aided Design*, vol. 22, no. 9, pp. 527-537, Nov. 1990.

[6] L.C. Knapp, *A Design Scheme Using Coons Surfaces With Nonuniform Basis B-Spline Curves*, PhD thesis, Syracuse University, 1979.

[7] T.M.P. Lee, *A Class of Surfaces for Computer Display*, PhD thesis, Harvard University, 1969.

[8] R.F. Riesenfeld, *Applications of B-Spline Approximation to Geometric Problems of CAD*, PhD thesis, Syracuse University, 1973.

[9] L.G. Roberts, *Homogeneous Matrix Representation and the Manipulation of n-Dimensional Constructs*, M.I.T. Lincoln Laboratories, MS-1405, May 1965.

[10] M.S. Rowin, *Conic, Cubic and T-Conic Segments*, Boeing D2-23252, Apr. 1964.

[11] K.J. Versprille, *Computer-Aided Design Applications of the Rational B-Spline Approximation Form*, PhD thesis, Syracuse University, Feb. 1975.

Robin Forrest is a founding father of computational geometry and noted authority on CAGD, publishing in the areas of curve and surface representation, antialiasing, and visualization. He is a professor at the University of East Anglia, United Kingdom.

Robin Forrest can be contacted at
University of East Anglia
School of Information Systems
University Plain Norwich
Norwich NR4 7TJ, England
e-mail: forrest@sys.uea.ac.uk

My Six Years to Evert a Sphere
by Nelson Max

One of the high points of the Siggraph conference is always the film show, because it illustrates the latest modeling, rendering, and animation techniques and also the raw computer algorithm power to generate 24, 25, or 30 frames per animated second. The films that jump out of my memory from past film shows are the following:

- a man juggling (from Information International Inc.);
- the first talking human, namely, an aging singer/piano player reminiscing (*Tony de Peltrie*, by Philippe Bergeron et al.);
- a flight through fractal mountains (*Vol Libre,* by Loren Carpenter, 1980);
- a huge robot construction worker ant, controlled by a smaller one inside (*The Ant,* by Dick Lunden et al. of the New York Institute of Technology);
- a few seconds of a woman swimming (by Rebecca Allen of the New York Institute of Technology);
- an infinite periodic vine (by Ned Greene and Paul Heckbert, in the Siggraph 1984 Omnimax film show); and
- the annual brilliantly colored swirling abstract metaball animations, inspired by undersea creatures (by Yoichiro Kawaguchi).

My own personal greatest challenge was modeling the geometry for *Turning a Sphere Inside Out*, a film that took me from 1970 to 1976 to complete. In 1970, the rendering engines capable of producing the film did not exist and neither did the modeling tools. My first test was stop-motion clay animation. I tried to imagine modeling everything out of pieces of spheres, planes, cones, and torii and made several clay models of intermediate stages. I even tried to continue this primitive design process in my head while trekking in Nepal in 1971.

In the summer of 1971, I spent a week at Cambridge University, hosted by Robin Forrest, as an early user of Andrew Armit's MultiObject, an interactive (by 1970 standards) design system based on bicubic patches. It completely filled the computer, and when I complained that one differentiability constraint was missing, it took some work for Armit to squeeze in the extra code and data. Models had to be saved on paper tape from one session to the next, because there was not enough disk space on the computer. After a week's work, I had modeled only 20 percent of one static stage in the deformation and gave up. It was then that I was informed that all previous models had been constructed by taking measurements from actual physical objects; no one had yet tried to create a geometric model from imagination alone.

When I started at Carnegie Mellon University in the fall of 1972, I tried to construct a similar patch-modeling system there. Luckily, Charles Pugh, a mathematician at the University of California at Berkeley, had seen the sketches of the sphere eversion in my storyboard and decided to construct anodized chicken wire models of several key stages to decorate the Mathematics Common Room. So I had models to measure, which I did at the Stanford AI Lab by setting the models on graph paper and getting the third coordinate from a plumb bob that I made from marked string and chewing gum. The people there suggested that I use their calibrated robot arm instead, but I thought my method was more reliable.

The patches were divided into polygons and finally rendered in the mid-1970s on the Case Shaded Graphics System, constructed by Dave Evans and Ivan Sutherland for Case Western Reserve University, as a hardware implementation of Gary Watkins's "span coherence" scan line hidden-surface algorithm. This was before frame buffers were practical. The machine was the descendent of E&S flight simulators and turned the rendered spans into video signals on the fly, without attempting to store more than one or two scan lines' worth. William Clifford programmed the PDP-10, which controlled the rendering, and his department chair later told me that his involvement was so distracting that it cost him his PhD.

For more recollections like the above, see Max.[12] It also contains sketches of, and the mathematics behind, the sphere eversion. For the computer graphics techniques involved, see Max and Clifford.[13]

References

12. N. Max, "Computer Animation in Mathematics, Science, and Art," *Computers and Mathematics*, D. Chudnovsky and D. Jenks, eds., Lecture Notes in Pure and Applied Mathematics, vol. 125, pp. 321-345. Marcel Decker Publishers, 1990.
13. N. Max and W. Clifford, "Computer Animation of the Sphere Eversion," *Computer Graphics*, vol. 10, no. 2, pp. 32-39, 1976.

Nelson Max is a professor of applied science at the University of California at Davis and a computer scientist at the Lawrence Livermore National Laboratory. He directed the National Science Foundation-supported Topology Films Project in the early 1970s. He has codirected two Japanese stereo Omnimax films for international expositions. His research interests include scientific visualization, volume and flow rendering, computer animation, molecular graphics, and realistic computer rendering.

Nelson Max can be contacted at
Mail Stop L-307
Lawrence Livermore National Laboratory
7000 East Ave.
Livermore, CA 94550, U.S.A.
e-mail: max2@llnl.gov

The Unfulfilled Spiritual Potential of Computer Art
by Duane M. Palyka

In the early 1950s, as a young boy in church, I became fascinated with stained-glass windows. Because the service itself was incomprehensible—the Mass was in Latin back then and the sermon was in Hungarian—my attention strayed to the glowing, sunlit windows illuminated from the outside. The images seemed to reach out to me and, each Sunday, endowed the church with a warm and mystical atmosphere.

A decade and a half later, as a graduate student at the University of Utah, my attention turned to computer graphics—the re-

cently invented frame buffer replaced the stained-glass windows, and a programming language replaced Latin.

In both of these situations, I found myself caught between the love and warmth of imagery and the rigid and obscure nature of language. As a computer artist, I wanted my images to create themselves and interact on the screen, but, instead, my artistic expression was impeded by the limitations of the programming language.

Fig. 1. Duane Palyka doing a self-portrait (see the front cover of this issue for a color version).

It was hard to be satisfied during those early days of computer art. No matter how sincere the programmer's intention may have been to provide a flexible tool for the artist, a sameness resulted in the produced works. All of the tools showed a preference for certain actions or certain shapes. This effect is not new. Ken Knowlton at Bell Labs, one of the first programmers to provide tools to artists, concurred with me on this point in the late 1960s. In reflecting on the results from his project, "Engineers and Artists in Technology," in which he distributed his programs to artists, he felt that the artwork he received from those artists looked much too similar. I have seen this in my own work, and I believe it occurs with today's tools as well.

Western art has traditionally illustrated and reflected religious and spiritual concepts. The paintings of Caravaggio and Da Vinci, for example, were new forms of expression containing older spiritual concepts. In this technical era, I have been able to design and implement self-generating and self-organizing computer art systems. They are meant to be meditative and devout, because that is where my spiritual and artistic interests have converged. But when an artist pursues authentic spiritual development, his tools become inadequate. I need, for example, to permeate my aesthetic into deeper levels than my tools allow. In fact, in many cases, I have needed a totally different approach (see Fig. 1).

For example, in 1979, I wrote a program that was a computer art analogy of gravitational attraction in outer space. In this program, I tried to make concrete what I had visualized as an artist: That is, I wanted to see how the images would be distorted by simulated gravitational spaces. I allowed uncontrolled distortions to happen, and the results contained my aesthetic. How could I have done this without my own programming? Although all the concepts I had were visual, the tools I used to create them were not. They were literal and mathematical, and I had difficulty working with them. Programming took the joy out of creation.

A visual programming language might encourage, rather than stymie artistic expression. Thus far, however, I have yet to see a visual programming language that is more than a flowcharting tool. But I think in terms of gestures and movements, not in terms of text and symbols. I think in terms of spaces, not polygons. I think about qualitative issues as well as quantitative ones. The computer is a flexible device, not limited by the hard definitions of traditional artist media. Simply stated, I would like more natural means to exploit the computer's rich potential.

Looking deeper, part of the problem may reside in our reliance on science. Although the scientific process is very good at solving technical problems, it falls short when applied to artistic thinking. Approaching holistic subjects by building on detail just does not work very well. Huston Smith has discussed this topic.[14] The question becomes: Is the scientific process capable of creating a logical expression for the spiritual?

Perhaps we need to eliminate the separation of *artist* and *scientist*. To encourage the natural philosopher, art schools should not teach that art is an escape from the oppression of technology, and engineering schools should not discourage artistic thinkers. In computer science, we should develop an artistically inclined computer language.

We must "undo" the development of the computer, which suffered from the unbalanced art/science attitudes of its times. We can, for example, develop new hardware and software that reflect a more balanced (and more timely) attitude toward artistic and scientific issues. If we can design a computer in which visual, holistic thinking predominates over linear, detailed thinking, then, I believe, the computer would be a more encouraging and less intimidating device. This may yield interesting results that might affect other aspects of our lives and society. And, of course, such progress would facilitate my work with spiritual computer art.

Reference

14. H. Smith, *Beyond the Post-Modern Mind.* Quest Books, 1982.

Duane M. Palyka began producing computer art in 1965 while studying at Carnegie Mellon University. He has since worked as both an artist and programmer at the University of Utah's Computer Science Department in the 1970s and at the New York Institute of Technology's Computer Graphics Lab in the 1980s. He supported the pioneering efforts of Jim Blinn, Ed Catmull, Jim Clark, Martin Newell, and others. He was a professor at the University of Michigan School of Art in the early 1990s.

Duane M. Palyka can be contacted at
1725 Oxford St., Apt. 101
Berkeley, CA 94709, U.S.A.
e-mail: ArtMath1@aol.com
http://members.aol.com/ArtMath1

Computer-Aided Geometric Design by David F. Rogers

Much of my early work in the fields of computer graphics and computer-aided geometric design involved the development of

techniques and programs for the dynamic manipulation of curves and surfaces for design and the use of those designs in computer-aided manufacturing.

The early work on dynamic manipulation of curves and surfaces for design was particularly inspired by that of Pierre Bézier at Renault and Robin Forrest at Cambridge University. The underlying thrust of the work is that the user is the designer and, thus, should have direct and immediate control of the design process. It is chronicled in a series of papers[15,16,17,18,19,20,21,22,23,24,25,26] that span nearly three decades. The culmination of this work is the 1990 paper[26] in the issue of *Computer Aided Design* honoring Bézier on his 80th birthday. That paper gives a very fast algorithm for dynamic manipulation of rational B-spline surfaces, including dynamic manipulation of the homogeneous weighting factor.

Dynamic manipulation differs from interactive manipulation in that modification of the curve or surface occurs *instantly*. Today, we call this rubber banding. Forrest, Les Piegl, and I believe that the 1977 Society of Naval Architects and Marine Engineers paper[15] is arguably the first published paper on dynamic manipulation of B-spline curves. This paper is almost unknown in the computer graphics community. Furthermore, the 1980 Siggraph paper[17] may be the first published paper on dynamic manipulation of nonrational B-spline surfaces.

As is evident from the references, I was ably assisted in this work by Steve Satterfield, who did much of the interface programming to my designs, and Francisco "Paco" Rodriguez, who did much of the numerical control interface programming, while I concentrated on the underlying mathematics and programming the fundamental algorithms. When I refer to *my designs*, understand that at that time we had only a vague idea of what a dynamic manipulation user interface should look like. Consequently, the design process consisted of my mumbling that I would like to be able to do this, and maybe this was the way to do it. Then, Satterfield programmed it, and I said, "Well, that is not what I meant" (which really meant I did not have a clue). After some discussion, I suggested something else. The process continued until we both agreed that this was the best we could do, or maybe one of us just got tired.

All of the early work was supported by the U.S. Coast Guard and, consequently, was done in the context of ship hull design. We used an Evans & Sutherland Picture System 1 (PS-1, Serial Number 9) driven by a DEC PDP-11/45 (see Fig. 2). The PS-1 was a calligraphic (line-drawing) refresh monochrome display. An interesting sidelight is that Dave Evans once remarked that I was the only person he knew who used a PS-1 as it was intended. On the other hand, in 1974, when the system was ordered, who else but the government could afford $200,000 for a single-user engineering workstation.

Just designing a ship hull was not enough; one had to produce the ship. Well, the U.S. Naval Academy was not in the business of actually building ships, especially not by aeronautical engineering professors. However, we were in the business of building towing tank and wind tunnel models. The powers-that-be thought I might be able to handle that. As a result, a relatively large, numerically controlled milling machine was purchased. At that time, one used APT or some similar language to write a program to drive a numerically controlled machine. The transfer document was a paper drawing. Rather inaccurate that. Here, we had spent considerable time and effort carefully fairing the lines constituting the ship surface, and we were to use a drawing and the, at best, circular interpolation in APT to manufacture the model. Not likely.

Well, there is a computer in the NC mill, we will reprogram that. Nope, the manufacturer will not release the source code. Hmm. A paper tape reader is used as the input device for the NC mill. So, we will write a program for the PDP-11/45 that will take the data from the CAD program and generate a paper tape to drive the NC mill. A few quick calculations showed that the resulting tape would be so long that it could be wrapped around the rather large engineering building several times.

Fig. 2. Evans & Sutherland Picture System 1.

After walking around in circles scratching our heads for a while, Satterfield, Rodriguez, and I came up with the idea of fooling the computer in the NC mill into thinking that the output of a Tektronix 4051 workstation, connected to the computer in the NC mill via an IEEE 488 GPIB bus, was a paper tape reader. Rodriguez designed and built a simple electronic interface and wrote a program, running in the Tektronix 4051, that converted the raw data from the CAD program into NC commands, including the necessary cutter offsets, machine speeds, etc., and did so in real time. In effect, we ran the NC mill from the Tektronix (see Fig. 3). The program ran the NC mill in what is called point-to-point mode, thus we had complete control over the accuracy of the cut. The program also allowed proofing the data and provided a graphical picture of the cutter location that was several steps ahead of the actual cutting operation (see Fig. 4). This may have been the first incidence of a general-purpose graphical workstation being used to directly run an NC machine (see Fig. 5). A similar program, running on a PC, is still in use today at the U.S. Naval Academy. I consider the Tektronix 4051 to be the first true production stand-alone engineering workstation. It contained a CPU, mass storage (a cartridge tape), a graphics screen, a keyboard, and a built-in programming language (Basic) complete with an editor, as well as both ASCII and GPIB data communication capability.

For nearly two decades, the U.S. Coast Guard generously continued support for this work. Early in the program, as a demonstration of the concept, I faired the lines for a proposed Coast Guard icebreaker using the design system, which we called computer-aided milling. Of course, the Coast Guard gave me a difficult ship to fair. The ship was short and fat (for a ship) and

straight-sided. Thus, fairing both the stern and bow areas became a challenge. In fact, the bow was rather a departure from common practice. I would work on it for an hour, which was about all I could stand, go teach a class, work on it a bit, go to a meeting, etc. Consequently, both to check the fairing and to proof the NC system, we decided to cut a small model about two feet long before cutting the final large model.

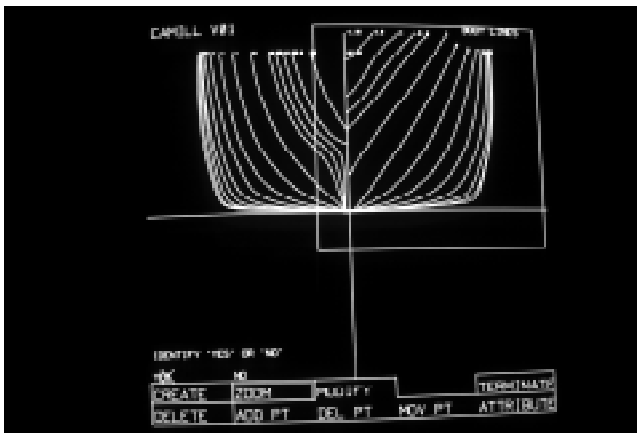

Fig. 3. The Tektronix display of the ship hull.

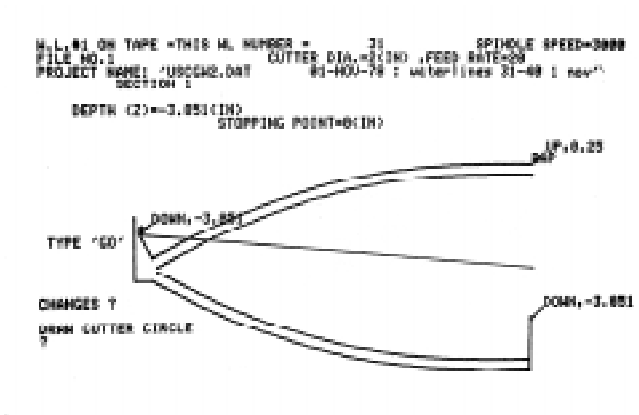

Fig. 4. A display of the milling machine cutter location.

The day before the model was to be cut, I called and asked Howard Chatterton, our sponsor at the Coast Guard, if he wanted to come and watch the model being cut. I also warned him it might take all day and that there were no guarantees. The system worked very well, so when Chatterton arrived late in the afternoon, the model was almost complete. Chatterton said he had a design meeting with the admiral and representatives from all the various departments the next day and asked if he could take the completed model with him. Sure, why not, he was paying for it. Off he went with the model tucked under his arm. Evidently, the structural engineering department had a problem with the bow design. They said there was no way it could be built strong enough to withstand the rigors of ramming the ship into a thick ice pack. So, they sat there during the meeting and redesigned the bow by drawing on the model with a ballpoint pen. The following day, Chatterton arrived with the "redesigned" model and asked if I could refair it using the ballpoint pen markings. What could I say except yes? After a week of muttering to myself, the redesign was complete. The subsequently built towing tank model used the redesigned bow. That model is still in use at the Naval Academy. Chatterton estimated that by catching the problem early, the Coast Guard saved several million dollars. I guess the research money was well-spent.

Fig. 5. The milling machine cutting out the model of the ship's hull.

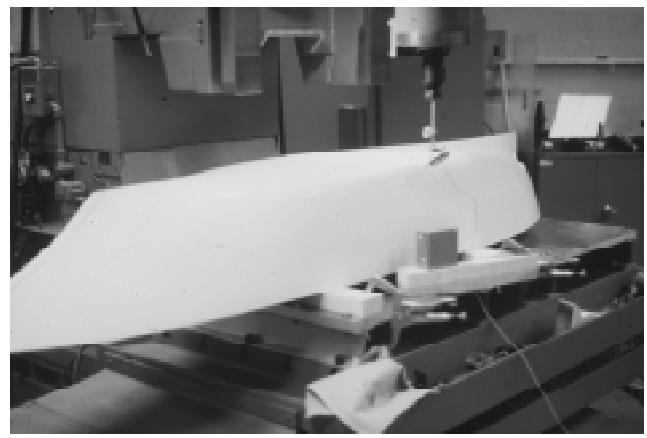

Fig. 6. The milling machine being used as a digitizer.

We also wanted to confirm that the resulting wooden towing tank model, which required handwork to finish, was built as designed. To do this, we built a digitizing probe using a standard machinist's dial gauge and another small electronic interface and wrote a program running in the Tektronix 4051 that turned the NC mill into a large 3D digitizer (see Fig. 6). The resulting data were stored on the Tektronix 4051 tape unit and transferred to the design system for comparison. It worked a treat. Furthermore, the results were surprisingly accurate.

Additional details of the historical aspects of our work in these two areas are chronicled in a book called *Fundamental Developments of Computer-Aided Geometric Modeling*,[27] edited by Les Piegl. This book is recommended for those who are interested in the early developments in computer-aided geometric design. The book contains articles by Andrew Armit, Robert Barnhill, Pierre Bézier, Carl de Boor, Ian Braid, Paul de Casteljau, Maurice Cox,

Charles Eastman, James Ferguson, Bill Gordon, Robert Mann, Mike Pratt, Aristides Requicha, Rich Riesenfeld, Doug Ross, Malcom Sabin, and Herb Voelcker. I felt honored to be included among these true giants of the early days of computer-aided geometric design development.

References

15. D.F. Rogers, "B-Spline Curves and Surfaces for Ship Hull Design," *Proc. Society of Naval Architects and Marine Engineers, First Int'l Symp. Computer Aided Hull Surface Definition*, Annapolis, Md., 26-27 Sept. 1977.
16. S.G. Satterfield, F. Rodriguez, and D.F. Rogers, "A Simple Approach to Computer Aided Milling With Interactive Graphics," (Siggraph 77, San Jose, Calif., 20-22 July 1977), *Computer Graphics*, vol. 11, pp. 107-111, 1977.
17. D.F. Rogers and S.G. Satterfield, "B-Spline Surfaces for Ship Hull Design," (Siggraph 80, Seattle, Wash., 14-18 July 1980), *Computer Graphics*, vol. 14, no. 3.
18. D.F. Rogers and S.G. Satterfield, "B-Spline Surfaces for Ship Hull Design," *Computer Graphics 80*, Brighton, England, 13-15 Aug. 1980.
19. D.F. Rogers and S.G. Satterfield, "Dynamic B-Spline Surfaces," *Proc. Int'l Conf. Computer Applications in the Automation of Shipyard Operation and Ship Design IV (ICCAS 82)*, pp. 19x-196, Annapolis, Md., 7-10 June 1982. New York: North-Holland, 1982.
20. J.C. Dill and D.F. Rogers, "Color Graphics and Ship Hull Surface Curvature," *Proc. Int'l Conf. Computer Applications in the Automation of Shipyard Operation and Ship Design IV (ICCAS 82)*, pp. 197-205, Annapolis, Md., 7-10 June 1982. New York: North-Holland, 1982.
21. D.F. Rogers, S.G. Satterfield, and F.A. Rodriquez, "Ship Hulls, B-Spline Surfaces, and CADCAM," *Proc. Intergraphics 83*, Tokyo, ll-14 Apr. 1983.
22. D.F. Rogers, S.G. Satterfield, and F.A. Rodriquez, "Ship Hulls, B-Spline Surfaces, and CADCAM," *IEEE Computer Graphics and Applications*, vol. 3, pp. 37-45, 1983.
23. D.F. Rogers, S.G. Satterfield, and F.A. Rodriguez, "Ship Hulls, B-Spline Surfaces and CADCAM," *Proc. NCGA 83*.
24. S.G. Satterfield and D.F. Rogers, "A Procedure for Generating Contour Lines From a B-Spline Surface," *Proc. Computer Graphics Tokyo 84*, Tokyo, 24-27 Apr. 1984, *IEEE Computer Graphics and Applications*, vol. 5, no. 4, pp. 71-75, 1985.
25. D.F. Rogers and N. Fog, "Constrained B-Spline Curve and Surface Fitting," *Computer Aided Design*, vol. 21, pp. 641-648, 1989.
26. D.F. Rogers and L. Adlum, "Dynamic Rational and Nonrational B-Spline Surface for Display and Manipulation," (invited paper in the commemorative issue honoring Pierre Bézier on his 80th birthday), *Computer Aided Design*, vol. 22, pp. 609-616, 1990.
27. D.F. Rogers, "Interactive Graphics and Numerical Control for Computer-Aided Design," L. Piegl, ed., *Fundamental Developments of Computer-Aided Geometric Modeling.* New York: Academic Press, 1993.

David F. Rogers has authored several textbooks, including the popular *Mathematical Elements for Computer Graphics* and *Procedural Elements for Computer Graphics*. He helped establish the Aerospace Engineering Department at the U.S. Naval Academy, where he has been a professor since 1964. His research interests include computational fluid mechanics, flight dynamics, highly interactive computer graphics, computer-aided design and manufacturing, and computer-aided education.

David F. Rogers can be contacted at
Aerospace Engineering Department
U.S. Naval Academy
Annapolis, MD 21402, U.S.A.
e-mail: dfr@usna.navy.mil

Comparing the Real and the Synthetic: Building the Cornell Box by Holly Rushmeier

In the early 1980s, graphics researchers began to look at simulation techniques, such as the radiosity method, to generate realistic synthetic images. One group was at Cornell University, working under the direction of Don Greenberg and Ken Torrance.

Generally, new image synthesis techniques are of interest because either they generate pictures faster or they generate more interesting pictures. Initially, radiosity was an incredibly slow method for generating images of incredibly boring scenes. The important feature of radiosity was that it accurately simulated what the scene being rendered would really look like. A new technique for demonstrating this feature was needed in place of amazing timings or dazzling images to justify work on radiosity. To develop this demonstration, Greenberg initiated a project to do a side-by-side comparison of a real scene and a simulated scene.

Initially, to perform the comparison would seem to be very simple. Just build a scene, like a box, take a photograph of it, and put that picture next to a synthetic picture. The problem turned out to be far more complex. On one hand was the difficulty of getting suitable input for the simulation; on the other hand was the problem of getting a suitable photograph.

In the original work at Cornell, Cindy Goral had generated images of an empty cube, with one side being a white diffuse light emitter and the other sides being red, white, and blue diffuse reflectors. This geometry nicely illustrated the color-bleeding effects that could be produced by radiosity solutions. The problem was how to make a physical equivalent of this box. What paints would be diffuse enough? How were the color values for the paints to be determined? How do you build a diffuse light emitter? And most baffling, how do you take a picture of a closed empty cube?

The results of the first box built by Michael Cohen (a carpenter in his pregraphics life) are shown in the 1984 Siggraph paper by Goral et al. The white emitting source "wall" was formed by removing the wall and placing the cube in a large, diffusely lit white enclosure with a hole in it for the camera. Since light entered the missing wall from all directions, this simulated a diffuse source at the wall. The resulting photograph demonstrated the color-bleeding effects but was clearly not identical to the simulated image.

In continuing work on this problem, three things became clear:

- Better measurements of the box reflectances and light source were needed, guessing would not do.
- There were two phases in generating the image that needed to be examined separately: simulating the light transport and, then, mapping the results to the gamut of the monitor.
- Photographs were inappropriate for the comparisons. The transformations of tone and color in the photographic process were significantly different and more complex than the process we were trying to simulate.

Better measurements would seem to be just a matter of money and time. Contrary to common perception, just working in a well-known graphics program does not allow an unlimited

budget. In particular, we had essentially no budget for physical experiments. All we had was a photometer that had been purchased for monitor calibrations. We made the most of this, using it to make directional measurements from surfaces, restricting the field of view using the cardboard tube from the center of a roll of printer paper. The rest of our "equipment" consisted of relatively inexpensive plywood, latex paints, lightbulbs, and a metal trash can (see Fig. 7).

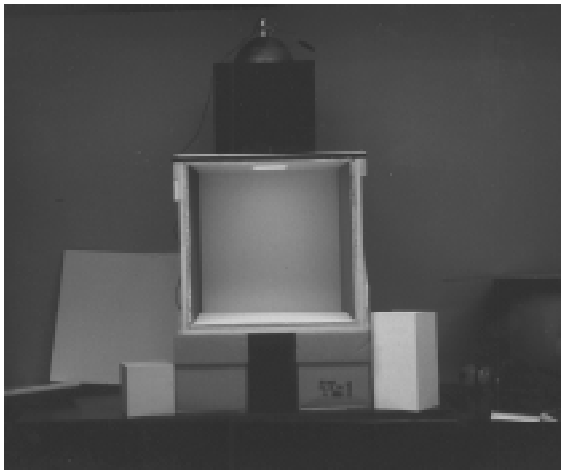

Fig. 7. An intermediate version of the Cornell Box.

Furthermore, just being at a university does not necessarily permit access to all of its resources. Space being typically the most precious resource on campus, we felt lucky to be able to take over a space not much bigger than a broom closet. Torrance gave us direction on what measurements were needed and where we could get help on campus. We were fortunate that several departments at Cornell were generous with their time and facilities:

- The College of Human Ecology let us use its spectrophotometer for an afternoon.
- The Physics Department helped measure our light source.
- A professor in the Chemistry Department lent us filters.
- The School of Architecture lent us extra space.
- Plus, the School of Chemical Engineering gave us some heavy black curtains from an old microscopy laboratory.

To examine separately the two phases in forming an image, we found that our photometer could be used for measuring the results of the light transport simulation if we could somehow put it in the closed cube. We came up with the idea of making a small diffuse light source in the ceiling of the box in place of the emitting wall. We could leave one side of the cube open to a black environment and represent this as a black wall and, then, place our photometer on this open wall. Our first source was a floodlight shining into a foam core box painted white on the inside. This worked fine until the box started smoking. We replaced the foam core with a "15-inch-high metal cone"—a metal wastebasket painted white on the inside with the bottom removed. Even with the wastebasket arrangement, it was noticeable that the light source was not perfectly diffuse. Rather than build more sources, we developed a variant of the radiosity method to account for light sources with directional variations.

The second part was to verify the mapping to the monitor. We wanted people to see the real and simulated versions side-by-side, in a sort of Turing test, to see if they could distinguish which was which. We did not want to use photographs, and anyone viewing the box and the display directly would be able to distinguish a box from a monitor. To overcome this problem, Gary Meyer came up with the idea of having people observe the scene and its simulation through view cameras (lent to the project by the School of Architecture). The view cameras create images on ground glass in their backs and made the real and synthetic versions appear as images of the same size, without the alterations of color and tone involved in photography.

We faced one more major problem (or feature). By the time we were ready to do our experiment, Cohen had developed the hemicube algorithm, so we could put objects into the cube that would cast shadows. With all the measurements done, we were set to make incredibly accurate images complete with interreflections, directional light, and shadows. The first images were a great disappointment—they were much redder in appearance than in the real box. The problem was that with the limited dynamic range of the monitor, the light source in the image could not be rendered with a color and brightness consistent with the rest of the scene. Rather than dealing with the dynamic range problem, we gave up and put a little black tab on the box that blocked the direct view of the light source.

The results of the project were reported in the paper "An Experimental Evaluation of Computer Graphics Imagery" in the January 1986 issue of *ACM Transactions on Graphics*. The bottom line of this experiment was that people could not tell which was which, and, in fact, trained computer graphicists got it wrong more often than did randomly selected subjects. In this article, I have tried to provide some idea of how the project evolved as we learned more about how images were formed and how to put together experiments from existing resources.

Holly Rushmeier is a pioneer in and leading authority on radiosity methods. She has conducted research at the National Institute of Standards and Technology and is presently with IBM's Watson Research Center. She was Siggraph papers chair in 1996.

Holly Rushmeier can be contacted at
IBM Thomas Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598, U.S.A.
e-mail: holly@watson.ibm.com

George Lucas Discovers Computer Graphics
by Alvy Ray Smith

My colleagues and I have been blessed to encounter a sequence of individuals of a unique variety I call *accidental visionaries*—a peculiarly American type, I suspect, and probably more fundamental to U.S. technological leadership than is commonly appreciated. Three of them, Alexander Schure (patron of the New York

Institute of Technology's computer graphics lab), George (*Star Wars*) Lucas, and Steven Jobs (now chief executive officer of Pixar), have directly influenced my life and helped to make the computer graphics industry robust and pervasive. They all deserve the title "visionary" in the sense that they jumped into the digital fray based almost solely on intuition and paid generously for our years of early experimentation and development. They made it happen. But, I believe they did not recognize the power of the technology they supported, which is why I call them "accidental visionaries." This story, for example, is about how filmmaker Lucas discovered computer graphics.

By this time, however, it had dawned on me that [George Lucas] did not understand raster graphics.

The time is 1981 in Marin County, California, across the Golden Gate Bridge from San Francisco. Ed Catmull is director of the computer division of Lucasfilm, and I am director of its computer graphics branch. Our goal of many years has been to make a completely computer-generated animated film. We believe Lucas hired us to use computer graphics in his films and, hence, help us along the path to our dream. True, he had not explicitly mentioned this as his motivation for engaging us. Instead, his specific request of us was to build three pieces of equipment to modernize the technology of filmmaking:

- a digital film printer (part of which would later be known as a "Pixar");
- a digital audio synthesizer; and
- a digitally controlled video editor.

But surely he must be interested. He had, after all, used computer graphics in his first smash hit *Star Wars,* in the form of Larry Cuba's black-and-white vector graphics—used in training X-wing pilots. And *Star Wars* was known for its use of computers (but no one was quite sure how).

By this time, however, it had dawned on me that he did not understand raster graphics. I had already commented to Catmull that "George doesn't know what he has." Actually, I had said he "doesn't know what he has *either*," since I had made almost the same remark about Schure years earlier at the New York Institute of Technology. Indeed, the use of computers in *Star Wars* turned out to be for control of mechanical devices—repeatable camera dollies for blue-screen shots—not for making pictures directly. Production of *The Empire Strikes Back* was under way, but there was no request that we help with it. In fact, it would have no computer graphics. So, although Lucas was clearly a visionary about digitizing Hollywood's technology, the best I can say about his computer graphics vision at that time was that he allowed me to assemble the best team of computer graphics wizards in the world, probably because he just did not know who they were.

Several of our colleagues from New York Tech, including Tom Duff, Bill Reeves, and David DiFrancesco, had joined us or would shortly. I hired Tom Porter from Ampex, Loren Carpenter from Boeing, and Rob Cook fresh out of Cornell University. Jim Blinn joined us from the Jet Propulsion Laboratory, where I had worked with him briefly between New York Tech and Lucasfilm—although he was to return to the Jet Propulsion Laboratory before the heart of this story transpired. The place was hopping. But there were no requests from Lucas to do what we were *really* good at.

The big break finally came from next door. Industrial Light & Magic (ILM), the Lucasfilm special effects division (at that time, it was working with physical models, not computer graphics), was in the building next to ours in San Rafael, California. ILM produced effects for other companies as well as for Lucasfilm. In this case, the outside company was Paramount Pictures, and the contract work was for *Star Trek II: The Wrath of Khan*. The director expressed his wish for a spectacular effect using this newfangled technology called computer graphics. Luckily for us, although ILM did not know computer graphics, one of its employees used an Apple computer and was, therefore, aware of us computer folks next door. I got the call to go next door for a meeting.

The story of the film hinged on the so-called genesis effect, a phenomenon that instantaneously brought dead matter to life. I quickly concluded that the director and designers did not know what could or could not be done with computer graphics, so I suggested that they let me think about the shot overnight and propose a solution that I knew we could execute. They agreed.

Now, it is important to put this opportunity into perspective. This was to be a major motion picture by a major studio. It would, in fact, become a blockbuster. I had just been given the opportunity to design 60 seconds of this film. Complete novices never get such a chance. But I had it and knew it. I worked sleeplessly all night on a concept employing everything I had at hand—principally an extremely talented team, including Carpenter and his fractal mountains, Reeves and his new particle systems, Porter and his paint program, the 3D rendering skills of Cook and Duff, and the digital filming expertise of DiFrancesco. I was still under the influence of Blinn's *Voyager* flybys of the planets at the Jet Propulsion Laboratory. I threw all this together and sketched a short, six-card storyboard for what would become the genesis demonstration scene in *Star Trek II*. ILM accepted my ideas completely—even giving me another shot to do (the retina ID sequence). This was to be my first directing job for the big screen and our first real movie job, a very lucky break.

At the first production meeting for the genesis demonstration, I outlined the job to the team and made a statement something like, "We will do a first-rate job of satisfying Paramount and the story requirements. We will captivate the viewing public with our story. But the *real* purpose is to create a 60-second commercial to George Lucas, so that he will know what he has and what we can do for him."

I told them how we would accomplish this latter goal. I was acquainted with Lucas well enough by this time to know that when he watches a movie, he is completely aware of the camera—the choice of angle, distance, movement, etc. This is much easier to say than to do. A director's purpose, after all, is to engage the audience so thoroughly in the emotion of his story that they pay absolutely no attention to the technology creating the illusion. Lucas could resist the director's emotional pull while watching a movie and concentrate instead on the cameraman's choices.

So my instructions to the team were that we would create a camera move that would "knock George's socks off." It would not be a gratuitous 3D camera move that beginners in computer graphics sometimes implement on their first attempt. It would at

all times be relevant to the story and help build its emotional force. And, Lucas would know that it could not possibly be executed by a physical camera.

We proceeded to design a move based on the idea of a spacecraft flying by a dead, moon-like planet with a camera attached to the craft. Carpenter was the main contributor to this move, designing a 6D spline (for the six degrees of freedom of a rigid body in 3D space) through about 150 points. It is a twisting, spiraling, accelerating, decelerating, sweeping, reversing, minute-long, continuous camera move that follows the sudden appearance of flame over the limb of the planet and tracks the wall of fire to build tension. The fire overtakes the camera and melts the planet. As the spacecraft pulls away on its outward-bound trajectory, the camera—now looking back—reveals that the molten sphere has been reborn as an earth-like, green and blue, alive, planet.

I will not go into further details of the production and its credits, because I have covered them carefully elsewhere ("The Genesis Demo, Instant Evolution With Computer Graphics," *American Cinematographer*, Oct. 1982). The important point to my story is that the day after the premiere of *Star Trek II*—the special first screening to its creators—the quiet, some say shy, Lucas placed one foot into my office and said, "Great camera shot!" and then was suddenly gone.

We were in Lucas's next film, *Return of the Jedi*, in a 3D holographic shot that Reeves and Duff executed. And Lucas's close friend, Steven Spielberg, used the team shortly thereafter in *The Young Sherlock Holmes*. The team had now been augmented with a spectacular new animator, John Lasseter. From there, digital filmmaking continued to evolve, reaching a recent crescendo with *Toy Story*, directed artistically by Lasseter and technically by Reeves. Our goal to produce a completely computer-generated film was achieved, after 20 years, in 1995.

Alvy Ray Smith helped begin such famed graphics facilities as Pixar, Lucasfilm, and the New York Institute of Technology. He has received a technical Academy Award for the alpha channel and has received the Siggraph achievement award. Presently, he is a graphics fellow at Microsoft Corp.

Alvy Ray Smith can be contacted at
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399, U.S.A.
e-mail: alvys@microsoft.com

My Inspiration for Dithered (Nonuniform) Sampling
by Rodney Stock

Back in the spring or summer of 1979, I had the good fortune to be dining with a longtime friend, Lynn Conway. I had not seen her in a while—in fact, not since she had gone to work for Xerox Palo Alto Research Center. At dinner, she excitedly told me about work she was doing on VLSI design and about a book she was coauthoring with Carver Mead of the California Institute of Technology, called *An Introduction to VLSI Systems*. The basic concepts boiled down to the ideas that transistors would be very cheap, wires would be expensive, and tools could be sufficiently automated to allow for design of large custom systems. At the time, I was doing hardware design at Ampex on a frame buffer for a video paint system, but I was still quite interested in 3D hardware design from my days in the early 1970s at Evans & Sutherland, where I worked on the design of ship and flight simulators (alongside two colleagues who also wound up at Ampex at the same time as me). Although I was quite busy with video design, in my limited free time I could not help but think about how one could apply Conway's and Mead's work to design high-performance 3D graphics engines.

Pathological cases would be far more unlikely if the sampling pattern were some nonregular pattern, such as a dither pattern.

If transistors were cheap, one obvious idea would be to have multiple processors working on rendering one image. Interleaving processors in image space looked like a decent approach to partitioning the workload, and for point-sampled (i.e., unfiltered or aliased) rendering, this looked like a good way to go. On the other hand, people were just beginning to get fussy about jagged edges and "crawling ants" on images, and the papers coming out of Siggraph were making it clear that spatial filtering had to be done in some fashion to antialias the synthetic images.

Filtering required that the interleaved processors communicate with each other if they were to avoid recomputing their neighbors' pixel values. Unfortunately, using lots of communications (i.e., wires) was anathema to VLSI design. This led me to think more about supersampling, where multiple subpixels are computed and averaged together to get a pixel value. This approach had been tried and usually worked fairly well—better than just increasing the pixel resolution—but too often would encounter pathological cases. Ed Catmull had pointed out at Siggraph that even with subpixel averaging, a receding picket fence would always alias at some distance away unless prefiltering were done. I realized that with high-resolution subpixel averaging, the error from one sample could usually be made quite small (with reasonable restrictions on the data input), but that the problem arose when the small errors added up to large ones—I thought of it as the errors being correlated, because the sampling pattern had a high correlation with the image.

That led to the *flash*—the insight that the correlation of the errors was the problem with the pathological cases—if the errors could be uncorrelated (i.e., if the sampling pattern did not correlate with the image being sampled), then the most common pathological cases were far less likely to cause major problems. Pathological cases would be far more unlikely if the sampling pattern were some nonregular pattern, such as a dither pattern. Dither patterns had been specifically worked out to minimize any 1D or 2D patterns the eye might detect. For that reason, it was also clear that random patterns were not what was called for here, because truly random patterns can allow for both degenerate sampling and regularity in the sampling. While not an analytically perfect solution, dithering the sample points surely seemed like a

great practical way to generate antialiased images using many processors that did not need to talk to each other. This was far better than what was being done at the time.

Wow. This was big time exciting, I thought. Eager for confirmation, I asked our algorithm guru, Tom Porter, to look into it. Deeply focused on painting algorithms at the time, he returned a couple of hours later and said he did not think it was particularly useful. I was crestfallen—it had looked like such a good idea to me. But he knew a lot more about math and algorithms than I did, and I had a lot of confidence in him. (I still do. As it turned out, at Lucasfilm, Porter went on to make major contributions to the algorithm and became a coauthor of the famed 1984 Siggraph paper, "Distributed Ray Tracing.") Nonetheless, despite Porter's discouragement at the time, I could not get it out of my head that this was a good idea, so I looked for prior research.

The government of Quebec, Canada, was so proud of us three Quebecers that Rene Levesque and his government ministers interrupted the proceedings of the National Assembly to telephone their congratulations to us at the University of Montreal.

I then spoke with Dave James, an audio digital signal processing talent, who said the closest he had heard of such ideas concerned the addition of pseudorandom noise to audio signals in order to mask quantization errors and some research on the error effects of unintended nonuniform audio sampling rates. However, neither seemed directly relevant to my research. Some time later, at an unforgettable dinner at Charlie Brown's Restaurant, I asked Steve Gabriel, our video digital signal processing guy and a creative whiz at sampling theory, what he thought about the idea. After five minutes of pondering, his face lit up, and he excitedly proclaimed that not only would it work but, in fact, it was an excellent idea. *All right*.

The next year, I went to Lucasfilm's Sprocket Systems (now Pixar) to lead the Pixar image computer hardware project. Once again, synthesis was not my focus, but I could not get the dithered sampling idea out of my head. I soon began suggesting that the 3D guys check it out. Eventually, Rob Cook, who had the needed math and software background, agreed to look into it, despite his heavy workload. He generated intriguing, exciting test images without aliases and established that random sampling was not the right approach—validating my initial insight that dithered sampling would be superior to random sampling. At the same time, in an amazing coincidence, Alvy Ray Smith appeared with a just published *Science* article concerned with the distribution of rods and cones in a monkey's eyes—they were nonuniform. Holy smokes, this was the way nature did it. What a validation. It was one of the most exciting moments of my career.

From there, Poisson disk distribution, time distribution for motion blur, lens distribution for depth of field, and formalization to stochastic sampling and Monte Carlo integration were added by Cook and others—see the 1984 Siggraph paper by Cook, Porter, and Loren Carpenter. Cook's elegant development of stochastic sampling methods is in the January 1986 *ACM Transactions on Graphics*.

Rodney Stock engineered Evans and Sutherland flight simulators in the 1970s. He led the hardware design effort for Ampex paint systems and for the Lucasfilm Pixar image computer. In 1985, he founded the Computer Arts Institute, located in San Francisco. He has participated in several Siggraph tutorials and is an editor for *Computer Graphics World*.

Rodney Stock can be contacted at
50 Cypress
San Anselmo, CA 94960, U.S.A.
e-mail: rstock@microweb.com

The Virtual Humans Story
by Nadia Magnenat Thalmann and Daniel Thalmann

In the early 1980s, we developed MIRA, one of the first abstract data-type languages for graphics and a forerunner of modern object-oriented languages. MIRA was designed to facilitate the animation of virtual worlds and 3D characters. Despite its apparent antecedents in Latin and Romance language words relating to vision, the language was actually named after our dog. More recently, our dog has been commemorated in MIRA Lab, the first author's lab in Geneva. Using the MIRA software, we have pursued our goal of synthesizing human figures, which we call "virtual humans." In this essay, we will briefly recall some social moments of our research.

In 1982, in collaboration with Philippe Bergeron, we produced *Dream Flight*, a film depicting a person (in the form of an articulated stick figure) transported over the Atlantic Ocean from Paris to New York. This film won several awards, including first prize at the Online Conference in London. The government of Quebec, Canada, was so proud of us three Quebecers that Rene Levesque and his government ministers interrupted the proceedings of the National Assembly to telephone their congratulations to us at the University of Montreal.

In l987, the Engineering Society of Canada celebrated its 100th anniversary. A major event was planned for the Place des Arts in Montreal. The main sponsors, Bell Canada and Northern Telecom, were interested in simulating Alexander Graham Bell in a sequence that would showcase both high-tech and art in Canada. Instead, we proposed depicting figures with a wider appeal. Eventually, the idea emerged to simulate Marilyn Monroe and Humphrey Bogart meeting in a cafe in the old town section of Montreal. The development of the software and the design of the 3D characters (now capable of speaking, showing emotion, and shaking hands) became a full year's project for a team of six. Finally, in March l987, the actress and actor were given new life as virtual humans.

The virtual Monroe and Bogart are now 10 years old. Monroe has acquired a degree of independent intelligence; she even plays the autonomous role of a referee announcing the score of a real-

time simulated tennis match on a virtual court, contested by the 3D clones of two real players situated as far apart as Los Angeles and Switzerland.

During the 1980s, the academic establishment paid only scant attention to research on the animation of virtual humans. Today, however, almost every graphics journal, popular magazine, or newspaper devotes some space to virtual characters and their applications. Hundreds of researchers work in the area, and all manner of situations are being simulated. During the years when we worked out of the mainstream, we persevered because of the widespread public appeal of our topic and the satisfaction of following our own vision.

Nadia Magnenat Thalmann and **Daniel Thalmann** have developed numerous techniques for modeling and animating human figures. They have published and lectured widely. Presently, Nadia Magnenat Thalmann directs the MIRA Lab in Geneva, and Daniel Thalmann is with the Computer Graphics Lab of the Swiss Federal Institute of Technology.

Nadia Magnenat Thalmann and Daniel Thalmann can be contacted at

Computer Graphics Lab (EPFL-LIG)
Swiss Federal Institute of Technology
CH-1015 Lausanne, Switzerland
e-mail (Nadia Magnenat Thalmann): thalmann@cui.unige.ch
e-mail (Daniel Thalmann): thalmann@lig.di.epfl.ch