Kurt Akeley Interview

Interviewer by:
Dag Spicer

Recorded: November 5, 2010
Mountain View, California

CHM Reference number: X5984.2011

Dag Spicer: Okay. So today is Friday, November the 11th, 2010 in Mountain View, California at the Computer History Museum and we're delighted today to have Kurt Akeley, one of the co-founders of Silicon Graphics--a world leader in computer graphic systems, with us today. Thank you for being with us, Kurt.

Kurt Akeley: You've very welcome.

Spicer: I wanted to ask you what it was like at SGI [Silicon Graphics, Inc.] in the early days in terms of the work ethic and company culture.

Akeley: Yeah. Well, we … there was a strong work ethic. We all worked really hard, but I think, right from the beginning we also worked hard and played hard. And it was a very collegial environment. We were-- a lot of us weren't very old-- I mean, Jim Clark, the real founder of the company was, I don't know, 35 or something I forget how old at that point. But the rest of us were quite a bit younger. A lot of us were just either graduate students or very recently graduate students. So young, not so much family, and so I think the company became our home as well as our place of work. And Jim, at least to me, [became] very much of a father figure. He was out to do something great and he was very ambitious about it but he was-- I think he really enjoyed building this little family and kind of bringing us with him where we were going. So we did things together as a group in addition to being at work a whole lot of hours. Friday afternoon [we'd] go out to what we called Building 2, you may not recall but the St. James Infirmary up the way here. And it was part of the culture to kick back on a Friday early evening and drink a few and eat some peanuts out of the shells and just relax and have a great time. So, I think, it was a great combination of interesting people and a lot of hard work and a lot of great fun.

Spicer: Tell us what a workstation is.

Akeley: Okay. Well, you probably have seen a lot of workstations recently because at least the way I think about it if you've got a modern laptop or a modern PC you've got everything that a workstation was many years ago. It was a computing platform that was powerful enough to do serious things with, serious things like design airplanes, or solve research problems in medicine. And back 20-some years ago, I guess we're talking closer to 30 years ago now, the really-- anything that was PC-like just wasn't up to the task at all. It didn't have virtual memory. It didn't have a large memory. It couldn't run big programs and it just wasn't a serious piece of computing. And workstations and the way I think of it Sun Microsystems is maybe the perfect example of the ubiquitous workstation. It's got a Motorola processor in it. That means that it has enough horsepower to do some real computing. It has enough memory to do some real computing. It has enough disks. And it's got a display that isn't just a Teletype kind of a thing, but it actually has the kind of display that Chuck Thacker and his team at Xerox PARC worked through shortly before the time that SGI was formed. Then together it was a powerful computer. But, again, all of those things and far more are available for $500 at the, you know, Fry's [a local Silicon Valley electronics store] or the local business store. You can have it anyway you want it. And there's a story there we could get into but basically workstations are a dime a dozen now but it was a very special and exciting thing in the early eighties.

Spicer: What was SGI's first blockbuster product?

Akeley: Okay. Well, this one is a little bit sensitive but I have to say I think it was the Silicon Graphics GT and I'll tell you a little bit about why. So SGI's first product shipped late '81, '82, it's been a while, but right around there. And those products made a huge splash in a very small pond. You know there were a handful of people whose needs were desperate enough that they were considering these really expensive products to do their work. But they weren't blockbusters. They didn't put us on a path to huge sales. The product I'm describing, the GT, first shipped in late '87, or early '88, right around there. And interestingly enough it was the first product that did not have the "Geometry Engine", the special piece of technology that Jim Clark had worked with Marc Hannah to build right [] before the company was formed and around which the company was formed, and around which the name was chosen and a whole lot of things.

So there's lots of stories behind that that we may [get] a chance to get to. But this was a product that solved kind of both problems in computer graphics in a very core sense here. Up until that point, we had built machines that had enough 3-D transformational power that we could take a model and re-project it so you could look at it from here or from there or however you wanted to look at it. And in that sense it was a 3-D system but some fundamental … that model probably had to be drawn with lines. If it was filled in with colors it didn't look like it was lighted. We didn't know how to have it arrange things so that the thing you saw was whatever was closest to you. You might see the back instead of the front and that meant it wasn't at all a convincing real 3-D experience. And the GT product that shipped late '87, early '88 had enough power to do those transformations to re-project from different views. But it also could do the image work, compute the pixels correctly so it solved two really fundamental problems.

One was you could draw things that were solid. And what you saw was whatever part of the solid thing was closest to you. That's the way the world really works. It's kind of a freebie when we look around but it turns out to be a difficult computational problem.

And then the second thing was that the color that was on those surfaces was computed per pixel, roughly speaking, based on where the lights were. So if you drew a sphere, it didn't just look like a constant color disk, it looked like a sphere. It would be bright where the light was shining on it and dark around the edges. These two things together were huge. So in conjunction with being able to move your viewpoint and interactively fly through or manipulate or change and then being able to see solid objects like they're supposed to be that's what, I think, was the first really blockbuster product that set us on a path to changing the world.

Spicer: And was the GT an add-on board or a standalone workstation?

Akeley: It was [a] system. So called the GT. And it was a workstation, but it's an expensive workstation compared to what you would have expected to pay for a Sun workstation. But it was a workstation and actually the GT shipped with a single processor. But shortly thereafter (October '88) we shipped what was the intended real product which was a multiprocessor system with the high-performance graphics sped up another factor of two or three, you know, [give] you a half year or a year, and you could do that.

And together that really set SGI on the course that it followed throughout the rest of its history, which was being a very high-performance graphics company. But also a very high-performance compute company and we solved the problem of marrying those two things. And, I think, that's a lot of why people liked to buy our systems. You didn't have to go by choosing a high performance computer, find some kind of graphic system, and then somehow try to get them together. And frankly it's not easy to get them together. We solved that problem so that they were very tightly bound. And as a result, you could write high-performance applications. You could write a flight simulator. We sold a lot of systems to solve [the] problem of training people to fly.

Spicer: Did you work on the project?

Akeley: I did.

Spicer: And what did you do? What was your role in that project?

Akeley: Well, I had a hand in all of it. You might be able to infer this … that it was a revolutionary product in the sense that it did things differently than had been the vision of the company at its start. It didn't have this Geometry Engine in it as a piece of custom silicon. Instead, it used-- well, actually, I can't resist. Here's the very first board and we used this part right here, this is a Weitek 3332 processor. It's a little floating-point accelerator. These days floating point, real numeric calculations, occupy a couple of percent of a microprocessor's die area. Those days-- shall I keep going?

<crew talk>

Spicer: Could we do that? Okay. Great. Okay.

<crew talk>

Akeley: So these days you can buy high-performance floating point in that little workstation you're paying $500 for at the drug store and it's two percent of the microprocessor die. Back at this time getting some floating point, some real numeric calculation, was a difficult thing. And this Weiktek part was a very special part but it wasn't custom. We didn't design it. We bought it. And we built a system around it. And the impetus to do that was something that I led, that it seemed to me that for a variety of reasons building our own technology from the absolute bottom up didn't serve us [as] well as getting more leverage in this area. But that wasn't an easy decision to make for a company called Silicon Graphics, right. So there was a strong internal sense that our mission was to design the silicon that would do the high performance graphics. In fact, where we ended up and where we had the most success was marrying compute and graphics as a systems company. "Systems Graphics" would be a better meaning for SGI looking back 30 years. That wasn't what the name meant but effectively that's what we were doing. And when we realized that that's what we were really doing it makes sense to re-appraise some of the ways we got there. And one of those was: should we really be doing full custom silicon? The answer at that time was 'no'. Interestingly, if you follow the course of the industry the answer became 'yes' again. Right? What NVIDIA does now, what ATI (AMD) does now, is definitely full custom silicon for graphics. And that became the right thing, again, and that's a turn that SGI missed. But it was the right thing back in '87 or '88 to stop doing that.

Spicer: That's very interesting. Who were some of the superstars at SGI beside yourself? And what did they do?

Akeley: Well, I'd have to say it's hard to answer those kinds of questions. The place was full of unbelievably capable people and a lot of the joy of working at Silicon Graphics for all of the years I did, which was almost 20, was that you could—if you had questions about high-performance file systems or questions about networking or questions about multiprocessing or questions about compilers or questions about anything--there was an expert, often *the* expert. Our sales organization, I think, was by far the strongest sales organization in the technical community, full of passionate Ph.D.s who knew their subject area and knew our equipment. You could get consulting for free from SGI to do amazing things like the Hayden Planetarium in New York. We didn't charge a penny to basically design the first real computer graphics planetarium. All kinds of unbelievably strong people. But if you trace it all the way back, where did it come from? It came from Jim Clark and his vision and his-- this care that I mentioned earlier of taking care of a group of people as we went off and did these really exciting things which were entirely, in so far as I could tell, what he saw the future to be. And Jim was a remarkable guy as you almost certainly know, when he left SGI he formed Netscape with Marc Andreessen and caused even a much larger stir than SGI had ever caused, in the business community as well as in the technical community. I think you could fairly say that he changed the culture of Silicon Valley in a pretty profound way. So out of all those extraordinary people I just have to come back and say that Jim is the remarkable human being who put together some pretty amazing things.

Spicer: Okay. Thank you. Tell us a funny story that typifies working life at SGI in the early days?

Akeley: Well, it's pretty darrn early. Our first building back in the way-back days we had more-- it's over in Clyde Court, the south entrance to Moffett Field, and it wasn't a particularly large building like the one we're in now. But it was quite a bit larger than what we needed for our teeny little company. And so we occupied some of the smaller rooms but the back two-thirds of it was not even built out, in the sense that it was just one cavernous room. And we got to calling it the skating rink after a while because that's what it looked like and, of course, eventually, somebody figures out that we could use it as a skating rink. So we decided to do that-- we had our first big party. We rented a bunch of roller skates, brought in a big old stereo system and got one of these disco balls and had at it and had a wonderful time. There was only one problem and that was that management had noticed that it was a big empty space too and they had made the decision to fill it in. And all that had happened was that, you know, the walls around here on these buildings are made with these steel studs. The first thing they do is use those nail guns and nail down a steel u-strip to the floor. So what was in this big dark room full of partially drunk employees having a great time are these steel u-strips nailed to the floor. So we had one broken leg and one broken wrist. And this is in that theme of working hard and playing hard. We had an awfully good time. It was clear that it wasn't an entirely successful idea, so we replicated it each time we got a new building and it became part of the company culture. I don't think there were any further broken legs.

Spicer: That's great. Who were typical SGI customers?

Akeley: Well, I mentioned this a little earlier that in a sense you had to be desperate. When you're going to spend $50,000 or $100,000, which is the kind of thing we were charging early on for machines, you had to have a problem to solve that you just couldn't solve any other way. Or there was something you wanted to do that you just wouldn't have been able to do otherwise. So who were those people? Well, our neighbors right here at NASA were sort of the prototypical folks. They wanted to do huge simulations. They wanted to run computational fluid analysis of air flow through a jet engine or figure out how the reversers will work or won't work through simulation, a bunch of really interesting problems that were cutting edge and simply couldn't be solved until these really high-performance systems came along. And even if they could be solved, they couldn't be understood unless you had the right kind of visualization to see the results. You know, its one thing to get a whole pile of numbers back. It's another thing to make some sense out of it. And we actually solved some of the graphics problems a little earlier but we became a high-performance compute and a high- performance graphics company, everything married together. So [if] someone's trying to solve a ridiculous computational problem like the air flow through a jet engine and then also be able to *see* what the result was, we're your guys. We're here to help. And frankly, in addition to having these amazing capabilities we had this very engineering customer-centric culture. We were very early on with using the Internet back before it was really called that. Our customers--the ones that we favored, at least--had direct access to our engineers by e-mail, by newsgroups. So we worked closely with them to solve the problems that they had spent whatever significant chunk of change on to solve. And so that's one, you know, the aerospace industry. Boeing was a huge customer. We did this wonderful ad at one point--I think it was the 767 but I could be wrong on the number here--with this wonderful thing where there's a picture taken from the ground, a little SGI machine in the front of the camera and then this huge 767 behind [the SGI box] and the little tagline says, "This is the 767. This is the box it came in." Right, and the box it came in was the capabilities that we were delivering for dramatically less than what other companies--like IBM for sort of lesser equipment--were able to do. And the result was, again, the number may be off here, but this aircraft was the first one where the very first mockup flew. I believe that Boeing's practice prior to then was to work out all of the errors and the sheet metal and the alignments on one, and to build another one that you tested. That was my understanding. In any event, it significantly changed the workflow to be able to do enough simulation and enough visualization--literally enough compute and graphics--to get the thing a whole lot closer to right in design and not have to literally hammer it out, if you will, in the physical world.

Spicer: A lot of the early supercomputers were sold to the U.S. government, typically for military applications. What fraction of military customers did SGI sell to?

Akeley: It was a lot. By the end all of the major labs, all of the contractors who were, I guess, they were part of the military industrial complex, the government, per se or the Aberdeen Proving Ground. We sold to all of these places. That said it wasn't the bulk of our business. So it was an important component of the business but I believe it was true throughout that the private sector, non-military, was a larger part of our sales. We had a huge business in medicine, significant business over the years in entertainment, so it totaled out. The military was important but it wasn't the thing around which the company was organized, for example.

Spicer: You've partially answered this but what attracted these customers to SGI products? Anything you would like to add?

Akeley: Yeah. I like to think that in addition to this neat marriage of technology and the kind of hand-holding support that we could give both with our strong field sales organization which, by the way, this building was the headquarters of. And the access to the engineering, I like to think that in some significant ways they were engineered better than things you could buy other places. And I like to think that because I was a part of doing that work, but also because we brought a real systems kind of thinking to what we did. And so it wasn't a bunch of pieces that were kind of stuck together. We thought hard about what ought to be done where and how things needed to be married together. How the architecture of the graphics could be designed so that it was both able to solve customer problems--both the problems we knew about and the ones we didn't (which is, by the way, the test of an architecture: if it solves problems elegantly that weren't in existence when you were doing the design)-- but also equivalently franchises the implementers of the system--which at the start were us, but we also drove the graphics architecture as an industry standard and so now it's everybody--allows them to do great work, to be creative and to do new things. And yet, present those in a structure that is consistent enough that the customers can continue to make great use of it. This is a really interesting and important problem and, I think, SGI made significant contributions to the industry organizing things that way. And since we were leading that it made sense to come to us.

Spicer: I read somewhere that beginning in the 1980s, and for the next ten consecutive years, every technical Oscar awarded involved an SGI machine. Can you tell us about the influence SGI had on Hollywood?

Akeley: Oh, that's great. I didn't know that. SGI was an important enabler for Hollywood to go in this wonderful new direction of using high technology to add to entertainment. Early on, probably people were using SGI graphics to do that. Our workstations allowed them to do parts of the flow, to work out animated models and to plan the motion paths of the animated models and to experiment, maybe, with some lighting. But the truth is that where those blockbuster movies came from, I'm thinking *Jurassic Park*, the ones that really put us on the map, by the time that happened our computer graphics weren't that important to Hollywood anymore. They were using our compute power. The fact is that in those movies you have to compute enough frames to show them at 24 per second for two hours. It totals out to a lot of frames and if you multiply that by a few million pixels each, it's an awful lot of calculation. And it's a calculation that parallelizes really well. And SGI had developed early on expertise in parallelization in computing, I think, more than-- certainly more than anyone else in this kind of workstation lower-end supercomputer space had worked through. And it was that that was the engine that really changed Hollywood, I think. So it's attributed to our graphics but, in fact, it was their graphics running on our very high performance parallel systems that, I think, was the breakthrough. Again, our workstations mattered, but I think if you total out the dollars in all, it was selling compute systems.

Spicer: More the floating-point performance than graphics.

Akeley: Exactly.

Spicer: Interesting. I had always assumed it was because of the graphics and the visualization part, but no.

Akeley: But the company really was about both compute and graphics. And, I guess, the answer is it's both, but I emphasize the compute because people don't really know. It's an interesting, I believe, fact.

Spicer: How has computer graphics changed since the founding of SGI?

Akeley: Well, you know, it's funny. In huge ways in terms of the average person's access, you know, when I met Jim Clark and I walked in his office and he pointed to a-- it was a depiction of an aircraft on the wall--and he said "We're going to make that move, that's what we're going to do" and we did. And so in one sense there was this very clear vision of what was going to happen and it happened. And it happened for $50,000 in 1988. And it happened for not much money at all in the '90s. And now we're so far past that that graphics has become the model for how high-performance computation is done. If you want to buy a supercomputer you go to a company like NVIDIA and you work with them to put as many GPUs [Graphics Processing Units] as you can [in]. You know that the Chinese, the fastest computer in the world right now [Tianhe-1], what's it full of? It's full of graphics chips or actually graphic boards because you can't get the chip-- it's too hard to put a chip on a board. You need to know what you're doing. So you buy boards and you put them in a computer. That's the most effective way to get an unbelievable amount of computing power for a moderate amount of dollars and a moderate amount of power. So in a sense the vision extended from just how can we best organize computing to solve this relatively constrained problem of doing interactive 3D computer graphics (through the standardization I mentioned earlier, which had the effect of broadening the space that all this computing power could be pointed at) to a fundamental level that you could argue Pat Hanrahan began with his work doing the RenderMan Shading Language that eventually caused GPUs to be programmable high-performance computers. And then generalize that a little bit more and you find out that all of those years of thinking and hard work and competitive struggle to build the best high-performance parallel system to solve graphics and shading problems turns out to be the best thing to solve an awful lot more than that. And it didn't happen-- I make it sound like it happened by mistake. It didn't. But there was a very straightforward progression to do this. So, in effect, computer graphics is ubiquitous. Everybody can have that airplane flipping around or play a high-performance game or whatever they want to do. But it's more interesting than that. It's actually, I think, changed everyone's understanding now of how to architect the best way to build a high-performance computing platform to solve pretty much all of the interesting problems that computing is going to be addressing in the next who knows how long. So [it's] a pretty interesting trajectory.

Spicer: That's great. You mentioned the Chinese supercomputer. I think the IBM RoadRunner supercomputer also uses [Sony] PlayStation Cell broadband processors.

Akeley: That's right which is-- it wouldn't be fair to call those GPUs exactly, but close, right. I mean they were designed to solve graphics problems at a system level. So the processor was able not only to do the graphics but the work that drove the graphics, but yeah, it's the same story. The hard high-performance computing problem that was tractable was graphics. And magically there was enough market in it that it could get over this hump that a lot of other technologies never got over of becoming ubiquitous enough that they're sort of self-sustaining. It isn't just the special thing that a few people buy. There's a market that's established with enough money to keep this thing going. That's what ultimately ATI and NVIDIA have accomplished. And, of course, Intel. We shouldn't-- you know, they're not thought of as, sometimes, as a high-performance graphics vendor but they are. They lead the market in a lot of ways. So those three companies have created computing out of graphics that is going to be, I predict, the center of all computing in the coming years.

Spicer: Now, we've been talking a lot about hardware, but let's break into software. You led an effort known as OpenGL. Why don't you tell us about that and why it was important to come up with this system?

Akeley: Well, the fact is that I actually have and that's part of the trick here. I talked about OpenGL as an architecture, as an interface, because that's what an architecture is. It's an abstraction that allows implementation to be done in clever ways *underneath* and applications to be done in clever ways *above*, but to have a meeting ground so that all the different implementations expose the same interface and all of the applications can do the same thing. The right word for that, I would assert, is architecture. OpenGL is an architecture. We didn't know it at the time. We didn't call it that. We made, I think, all of the right decisions to make it an architecture. We made this contract that everyone underneath would expose something that would look the same so that everyone writing applications above could take advantage-- could make the large industrial investments that are necessary to do interesting things--with some confidence that they could use different products at *one* time and then continue to use products *over* time without having to change what they did. That's what an architecture buys you. But often, OpenGL is thought of as software because, indeed, the way you get it on your workstation or PC is it shows up as a little library that you would compile against or run a program against. But the fact is that it's an architecture represented as an applications programming interface [API] instead of as an instruction set. So people talk about ISAs, instruction set architectures. That's the classic way to expose an architecture. It turns out to be the wrong way to expose a graphics architecture for a whole bunch of reasons. But the easiest one to understand is, if you make the architectural specification too low you disenfranchise the implementers of it. It's hard to make microprocessors which have an instruction set architecture really different from each other. Too much of how they have to work is exposed. The graphics architecture, by being at a higher level and by being shown as a programming interface instead of as an instruction set architecture, left more room underneath. And that more room underneath has been filled in these exquisitely wonderful ways by building amazing high-performance parallel systems. And by the way, you can take advantage of all of that parallelism--and it's a lot--without having to think about parallelism. That's magic, because it's hard to think about parallelism. It's hard to write parallel programs. It's not just common wisdom. It's true. But if you provide the right interface-- that's another thing that graphics has given the computing world is at least a really solid example of doing interfaces, architectures that are at a level--that makes it possible for implementers to provide a whole lot more interesting value and, in particular, to provide the value of parallelism in a way that consumers don't have to deal with the complexity. [It's a] big deal.

Spicer: What happened to SGI?

Akeley: Well, you probably know that it declared bankruptcy a few years ago and then it popped back and, I guess, it's been sold to Rackable [Systems] or something. I actually don't follow it very closely. I was so emotionally attached for all of those years that when I finally became unemotionally attached I pretty much just let go. But that said, we can speculate a little bit about what kind of things went on.

So here's one way to think about it. Back in 1982 we made a business decision. The business decision was to go for high margin, provide high service, and don't worry too much about volume. I think you can make a reasonable case that that business decision has a-- what's the word I'm looking for--a trajectory. It's a little bit like tilting your gun up to a certain angle and shooting something [hand moves through an arc] but it's going to come back down. And the history of computing just says to me [if you] don't go after volume, if you don't sell millions of GPUs, if you don't sell hundreds of millions of PCs, if you don't sell billions of phones, whatever it's going to be, you're going to be on the losing end of history. So one case you could make is it was as simple as that. That the trajectory was set essentially when the company started and then it had a glorious assent at a pretty high apogee but it was going to come back down. That's an argument that gives up all responsibility for the rest of the path and so it's almost certainly not fair, but, I think there's an element of truth to it.

I mentioned earlier that we made a really tough call as a company back in '87 to change our fundamental technology approach, to walk way from what it was that made Silicon Graphics, Silicon Graphics: these custom transformation engines that solved an important part of the problem of interactive 3-D graphics. We walked away from that and instead started buying technology and then thinking of ourselves more as a systems company and less as a silicon company. I think that saved the company. I don't think it would have-- the trajectory would have looked different, right. Obviously nothing saved it but instead of it peaking somewhere around '95 at a $5 billion valuation or something, a pretty interesting thing, it would have peaked a whole lot earlier and gone down. But I also mentioned that we didn't make the subsequent call which was to figure out that that situation had changed again. And that it was now-- the technology was ripe for going back to doing full custom or at least reasonably custom. We could argue about what's full custom, but the kind of thing that NVIDIA and ATI and actually a whole lot of other competitors were doing there for a while. If we had made that transition, maybe the trajectory would have been a little longer, who knows? So, I think, it's fair to say it's a combination. That we made a fundamental business decision early on that meant we didn't get volume. That meant we didn't get the volume apps. That ultimately meant that it just couldn't play out. But we also made mistakes along the way. We did some great things, made some mistakes, and that changed the shape. But my gut is that the thing was on a trajectory.

Spicer: Tell us about the aftermath of SGI? For example, a whole bunch of people went from SGI to NVIDIA, for example. Could you tell us a bit about that?

Akeley: I guess the thing that's fun for me seeing this is that I'm not a networking kind of guy in the human networking sense. So when I worked at SGI for years and years we had less than five percent regretted turnover, essentially nobody left, out of our engineering organizations at least. It was amazingly constant and so you got to know a bunch of great people and you just kept working with them. It was perfect. But a side effect of that was that, at least for people like me who didn't make an active effort, there wasn't really much that I knew about the world outside of SGI in terms of personal relationships. And actually with OpenGL that changed significantly because we were building strong relationships with other companies. But as a sort of general rule, my experience was that most of the great people I knew were right with me. And, of course, that changed in this unfortunate but dramatic way when SGI-- it didn't quite explode but, you know, if you played it-- filmed in slow motion and played it back it would look really stable and then all of a sudden 'blam,' all of these great people we were talking about earlier scattered out and they're all over the place. And, of course, an important chunk of them went to NVIDIA and an important chunk went to ATI, and important chunks [went to] Microsoft. And important chunks at all of the major technology companies, venture capital firms, you know, just all around the Valley there are these graduates of SGI. And so I find myself actually more connected than I ever would have imagined because now all of my old friends are here, there, and everywhere else. So there's always a little golden lining in even really unfortunate things and for me that's sort of the sad positive part of it breaking up.

Spicer: Where do you see computer graphics going in the future?

Akeley: Yeah. Well, I may have tipped my hand a bit about that but, of course, the $64 or 64 million or billion dollar, whatever question is can computer graphics-- will it continue to exist as a specialized hardware activity? Will there be GPUs in the future is one way to phrase that question. And the answer that I predict, for whatever it's worth and I've certainly been wrong a fair amount in my career, is that the technology, the approach to doing high-performance, highly parallel computing is going to continue to exist in the future. It's going to be around for a good long time because I think it's just inherently right. And it may, in fact, be one of the only ways to do highly parallel computing. But I would also predict that in the process of doing that, it's going to move in and take over mainstream computing. And so when someone who doesn't have a sense of the history pries open a box and says oh, I see, there's 256 processors and off on the side there's a little place that they can tickle that causes pixels to go out to some display device, whatever that looks like--there's no GPU here. It's just a high-performance computer with a little thing on the side to dump pixels out to a display. And they'll be right looked at through a sort of naïve frame, the rest of them is saying -- the way those processors are organized, their instruction sets, the vector units that they have and the way they can do scatter and gather and a bunch of highly technical things to do with parallel computing--that's all about the way graphics evolved computing. And so the whole thing is a GPU and it's the CPU that's missing. It's sort of the opposite way to look at it. And, of course, neither extreme is the truth. But so my prediction is yes, the technology that graphics people and graphics companies have developed over the years will live on as the core of high-performance computing, but there may very well be a year not too far in the future when the thing we call a GPU as distinct from the rest of the computer isn't there anymore.

Spicer: Related to that question, do you see any progress being made towards helping people, programmers, especially, adopt the parallel paradigm? It seems like the single biggest problem facing the adoption of multi-core processors.

Akeley: Well, you know, there's people chipping away at it in a whole bunch of ways and I'm certain that progress is being made. My futurist hat is not something that you want to pay me for, but my gut is that one of these years-- the way progress is accelerating, kind of the Kurzweil way of thinking of things that things are changing and they're changing faster and faster--that before we ultimately solve the problem, having people really deal with all of this complexity, we will solve the problem of machines dealing with the complexity. I mean in some sense it's wrong to pin this problem on individual programmers. And graphics has certainly offered a kind of a respite from this by doing this higher level architecture that allows an awful lot of things to be done without having to think as much in parallel, right.

When you write an OpenGL program--even back in 1992, let's say in '96, when you can buy a machine called Infinite Reality, an expensive machine from SGI--there were on the order of 300 fully independent processors under that interface. That's a lot, right. And you as a programmer didn't need to know anything about parallel programming to take advantage of all of that power. And this was a really good idea and it's going to get more legs but it doesn't solve the whole problem. You need to have something well enough defined that you can define such an interface.

But what a really neat thing that's come out of it is this idea of shaders. Again, Pat and the RenderMan interface kind of writ large into all of computing now is that when you write a shader you're also taking advantage of massive parallelization but you don't need to think about writing in parallel. A shader is a sequential program. So the person who writes this and this is just-- in graphics we call them shaders because sometimes they're used to compute color for a pixel but they can be used to compute anything. And no matter what you're using it to compute, you don't need to think about it in parallel. You just think about the one running and the system distributes out, thousands, millions, you know, whatever it amounts to, and takes care of a lot of the problems. So that wasn't the original graphics model in hardware, but it was the original-- near original--software model for graphics and it's made an awful lot of interesting parallel problems very tractable for people who don't want to or aren't able or it's just too hard to think about all of the parallelism. So, you know, in the short term some of those things will help. There's a bunch of other --transactional memory--there's all kinds of ideas for helping but, I guess, I kind of feel like by the time we're somewhere near really sorting it out the machines will be able to do it for themselves. That's my 10-cent guess.

Spicer: That's great. I think that's it. And anything else?

Akeley: No. I'm all set.

Spicer: Okay. And then we wanted to shoot the board as well.

Spicer: Can you also talk a bit about it? Is that okay? Just a couple of sentences.

Akeley: So what we're looking at is the sort of first prototype board of a Geometry Engine that doesn't have a full custom Jim Clark-conceived integrated circuit Geometry Engine. Instead, it has a Weitek 3332 part in the upper right of the board as you're viewing it. And together with a bunch of microcode control it became collectively a Geometry Engine that had far more capability than any custom geometry engine that SGI had done before. And it was the start of a set of machines that had a huge impact on the industry.

Spicer: What were the machines?

Akeley: The opening-- the first--machine was the Silicon Graphics GT shipped late ’87, and then the whole GTX series that included the parallel processing.

Spicer: Did you want any other views of the board? Any close up on any part of it?

Akeley: No, I think we’re set it. I don't know the back is kind of interesting. There’s a story about the back too if you want.

Spicer: Sure. Wire-wrap. This is nice.

Akeley: So this is the back of that same first non-SGI silicon Geometry Engine. What you notice is that there’s a large area of wire wrap in one color and then a smaller area in another. I was in such a rush to put this prototype together that I took a board that had been used for something else. I ripped off the wires in the places that I needed to add the new circuits. I left the wires in the places I didn’t care about and this prototype was designed and constructed and tested in about a week. Those were the days. You can’t do it anymore.

Spicer: I’m glad I didn’t have to test that.

Akeley: Yeah, it is kind of a fun story. I forgot which color was which so I just-- and I didn’t want to turn it around.

Spicer: What’s that other gizmo you’ve got there?

Akeley: I brought along-- this is another-- we haven’t gone into this story at all but this-- there’s a lot of-- it’s fun but there’s a lot of pain associated with this one as well. It’s a little board that plugs into the mouse interface on a Silicon Graphics workstation.

<crew talk>

Akeley: Sure. This is a little board that generates a waveform. It’s just a square wave generator. And it plugs into the mouse interface on an SGI workstation. This would have been late eighties, probably. Those workstations had memory problems. They were failing in the field and we couldn’t figure out why. And one of the ways that I worked out to do the experiment-- you couldn’t tell when the memory got corrupted you could only find out sometime later when you tried to use the corrupted memory. So the question was can we catch it actually happening? And one way you can do that with computers is you can write a little test program and run it as a very high priority interrupt. We’re getting every teeny little bit of time will run this thing and check and see if that location is corrupted. And then it’s kind of like a logic analyzer trigger, you don’t know exactly when it happened but you know it happened really recently. It happened since the last time you ran the little program. So how do you generate those high priority interrupts? Well, it turned out that this [Motorola] 68000-based workstation was designed by a guy named Bart Locanthi who was at Bell Labs before he joined SGI. He had figured out that the way you could interface to a mouse didn't take much hardware at all. You just generated a high level interrupt every time the mouse quadrature changed an edge and go read it in the microprocessor and figure out what had happened, right. So all of the hardware that was required was kind of a single register and an XOR gate to generate the interrupt. So when I wanted to interrupt the system really rapidly I just unplugged the mouse, plugged this little gizmo I built with just a square wave generator, jam it in the mouse interface and it's generating a high level interrupt, a level seven. So it couldn't be ignored. And we used that as part of the many attempts to debug this horrible problem which we eventually solved but it was hard to work.

Spicer: That's a good story. Did you have error correction in your…

Akeley: We did not in these early ones. Certainly, later on we did, but it did have parity. And I invented a clever thing which I subsequently learned everyone who has ever done a parity system invents which is the ability to load in bad parity so that when a location would be accessed you'd get an interrupt. It's one of those inventions that happens over and over.

Spicer: Well, I think that's it. Thanks so much.

END OF INTERVIEW