Palo Alto CA
0:47:06
m4v

Gary Bradsky

An interview conducted by
Peter Asaro
with
Selma Sabanovic

9 June 2011

Q: And could you just start by telling us your name, where you were born, and where you grew up?

Gary Bradsky: I am Gary Bradsky. I was born in Arizona but basically grew up in Silicon Valley in Palo Alto. A long time ago.

<laughter>

Q: Where did you go to school?

Gary Bradsky: At Palo Alto High School.

Q: And your undergraduate?

Gary Bradsky: Berkeley.

Q: What did you study there?

Gary Bradsky: Electrical engineering.

Q: What motivated you to do that?

Gary Bradsky: I was originally a Physics major, and I decided, at that time, I didn't want to have to go all the way through to get a Ph.D. to do anything sort of useful. So, I thought that with EE I could get work for a while. I always intended to go back to graduate school, but I didn't want to go right away.

Q: And were you involved in robotics applications and projects at that time?

Gary Bradsky: No. <laughs> I was involved in wind-turbine design and Professor Otto Smith and...

Q: And when did you first sort of encounter robots?

Gary Bradsky: In a way, so, my background as – well, my graduate studies were in sort of biological vision, modeling the human perception system. And, again, when I got out, since I took my time, I was an older student at the time, I decided I didn't really want to hang around and do post-docs and keep getting paid little. And so, I decided that I'd switch into computer vision sort of formally. And I ended up, after a bunch of detours, I actually worked as a quant in a Wall Street kind of operation for a while. I didn't like it, although it paid well. And so, then I went back to research, Intel Research, and that's where I really started computer vision, because I had some background and they wanted to work on this image-processing library, and so that's where I started there, with this library group on the promise I would go into the research group and that happened a year later and it turned out to be a great connection, because that's where I started the open-source computer vision library. And, you know, that's one of the things I've been running or it's become very popular over time. But I wouldn't have been able to do that without the sort of knowing the people in the library group, who weren't doing anything like that, but I just started this as my own initiative. And so, where I met – finally intersected with robots was Sebastian Thrun, who came by – well, he was coming to Stanford. I knew who he was, so, I don't remember how it happened, actually, but he came. He split a sabbatical and spent some days of the week at Intel Research. And so, he had a little – an erratic that would wander around and we were doing some things there. But then I had a sabbatical coming up, and so I – he talked me into going to Stanford for the sabbatical. I was going to go to Switzerland. And I ended up, not during the sabbatical but right after, you know, the DARPA Grand Challenge had run and Stanford wasn't in it. Thrun didn't go the first time. And the second time was then cueing up and I was just visiting him and he said, "Oh, let me show you this Stanley." Whatever. I mean, it wasn't Stanley at the time. It was a Volkswagen SUV. And so, I looked at it and said, "Well, I'm in. This looks like fun." So, that was really the first robotics project I had been on, although I had been doing computer vision and had, you know, dabbled around in little stuff that is robot-capable. So, I organized the computer-vision team, and we won the race. <laughs>

Q: So, where did you do your graduate work?

Gary Bradsky: In Boston at Boston University under Grossberg. This was biological modeling of the brain. So, I guess Grossberg had broken off from MIT, and I decided, like I'd become interested in the brain, because prior to, way back, I had worked at this place called Neuroscience, which was a computerized, one of the first computerized EEG companies. And then, I got interested in the brain and I figured this – he was doing neural networks, biological neural networks, sort of modeling perception, and I thought, well, this is the best way to combine these interests. And so, you know, I went to him. I was on a business trip and I talked to him and I go, "Okay, I'll just go to graduate school here." So.

Q: So, did you really learn a lot from the biological systems about how to build better computer vision?

Gary Bradsky: No. <laughs> I mean, I learned a lot. I keep hoping to put it to use. There's maybe a few principles, but I don't think biology has informed computers much. I mean, in a general sense. You know, von Neumann, whatever, his first things he was thinking of were neurons, and then there is, you know, some things now. There is this deep-belief learning, and some things in AI, you know, are influenced by some conception of what the brain is doing, but it's been a pretty weak link. And everything I found to be successful, sadly, <laughs> with robotics, is you really engineer the system. So, I haven't really done, you know, the bio-inspired robotics and whatever. I also don't see them as being such a big success yet. I mean, it hasn't happened. You know, what's a big success? These industrial arms and Stanley and, you know, the SLAM and all these are very sort of engineered systems. So, you know, it seems like that's what we're able to make successful, and, whatever, maybe the hardware that we have is just too different from the brain to make some connection. I can go on and on, you know. Like, people in AI, they're always thinking if we had enough processing power, that that's when AI is going to happen. Well, we have essentially equivalent in different modes to, you know, to compete with the brain or at lease approximate it. So, that hasn't magically made AI appear. And then, you know, they go, "Well, if we had enough data." Well, often, you know, if you calculated out when Google trains on an image, they often train on, you know, to get some object recognition. They often train on more images than a human sees in a lifetime. And yet, you know, we're arguably quite a bit better. So, none of those things have panned out. I mean, there's fundamental things we still don't understand, and, you know, kind of people on the cutting edge are trying things, so, you know, they have this sparse coding theory and, you know, like Andrew Ng at Stanford is working on these things and maybe some of that will pan out.

Q: And so, when you're at Intel, what motivated you to go into the open-source computer vision library? Why was it important to make it open source?

Gary Bradsky: Because, so, I joined Intel, and at the time, I guess their charter was, you know, write papers and get – the thought was, let's get this on the map, Intel Research. But I looked around and said, well, you know, I could be in a post-doc or whatever and write papers, maybe even better, you know, and people in academia are pretty good at writing papers. So, I said, what can I do here that's more impact or more leverage. And I was also, you know, at the time, visiting MIT Media Labs, and I saw – I was looking at the students there. You know, they were doing a lot more, because they had so much infrastructure built up. Like, when you come into the Media Lab, you weren't just, okay, I'm going to write my own, you know, gradient detector and then I'll build that up. They had like this whole system. They had stereo working. They had everything working. And then they come on and built on top of that. And so, you know, one of my ideas was, I go, well, you know, what Intel really wants is to drive the need for, you know, processing power in the world. You know, like computers are getting fast enough for spreadsheets and whatever. And so, I go, well, that combines, you know, that sort of charter if I can speed up people doing computer vision by giving them the same kind of infrastructure that a student at the Media Labs would start with. So, I would make that universal. And so, you know, that was the idea at the time. And, you know, in life, very few of your ideas pan out, but that

original conception for OpenCV, I think, has really panned out. Like, it did boost computer vision. It did get increased need for MIPS or, you know, out in the world and, you know, it spread around. So, I think, you know, it did answer that problem and still does, you know, so.

Q: When did you start on that?

Gary Bradsky: 1999. Well, no, no, no, no. Yeah, 1999, I think. I think, you know, it started as a little project on my hard disk, and then at the time, Intel had these Russian contractors who then became employees and I sort of cobbled them in, you know, before I actually had permission. When someone went on sabbatical, I took their group and started them working. By the time they came back, it had too much momentum behind it, and so they made it an official project. And I think 2001 was when we actually came out with it at CBPR.

Q: And what were some of the challenges of building a unified vision library?

Gary Bradsky: The same challenges we had. <laughs> I mean, so, I wouldn't say unified. I mean, that's the challenge. I mean, so, there's a certain, you know, how complex you should be and how much you should include. So, I mean, OpenCV is over 2,500 functions now, and, you know, not all of them are used and not all of them should be in there. But, you know, so, what to put in and what to keep out is a challenge. I think, you know, how to design and architect a library – at the time, you know, what the – C++ had this name mangling problem, so we stuck with C. Of course, you have a lot of more tools in C++, and we finally switched the whole library over to C++, though the C interface remains, probably forever. There is no reason to get rid of it, really. But, you know, I've seen other libraries that just went – that came later, and so, they had more like boost tools and they just went what I think was overboard in templatization. And so, what I really – I think one of the good things in the library and what I wanted is, I didn't want – you have to learn like a system before you learn the function. If you wanted to do Canny edge detection, you just take and do Canny edge detection. And so, I think to that extent, that was good, and that's one of the reasons the library spread, because a lot of other libraries are really frameworks. And then you go, well, I have to learn the framework and get all this buy-in. And OpenCV, you know, you just, I mean, people use it in bits and pieces, and that's how it was intended. But we have now tried to organize it into modules, so it's getting more categorical, but there is no – there is still no overriding framework that you have to learn, though.

Q: And you were also involved with the open-source machine learning?

Gary Bradsky: Yeah, so, well, there's a machine learning library in OpenCV that came out of this – it's statistical machine learning that I did with – when the dot-bust happened, I figured, hmm, <laughs> like everything is falling through to the floor. Maybe computer vision, like who

are you going to cut, you know, the microprocessor designer or this computer-vision guy? I thought, I better get a little more relevant to the core of Intel. And so, what I did is I started doing projects with Intel Manufacturing, and it involved the statistical machine learning. So, I sort of do both things. And we were – I actually came out with the Bayes Net Learning Library that's still out there somewhere, but I just ran out of cycles to run all those projects. And I guess the guy who was on it never really drove it as much. It takes a certain personality to keep open-source going. But in manufacturing, there was just no way. We had to get them onto decision trees and things, not the Bayes Net.

So, we gave up on that and we just used these statistical techniques and, you know, there was a pretty successful project in basically reducing the need for testing so you'd have faster throughput and subject to some pretty hard constraints on they didn't want to see – they didn't want to lose the statistics, so we had to impose random sampling on it, and they didn't want to see the numbers change at the end, despite the fact that we show that it's sort of counter-intuitive, that it's actually better in some cases to have the parts fail at the end and just run them through and then bake them in. Even though that seems like it's more expensive, it actually turned out with all these steps not to be in the slow-downs, but they wouldn't hear of it, so. <laughs> But, I mean, that thing is now running in the manufacturing plant, so. So, when I – after I went back to the computer vision library, I drew in those techniques. We actually had to rewrite them, because now they were, you know, part of the manufacturing process and so top-secret, right. <laughs> So, we had to rewrite all the code for the decision trees and all that stuff. And so, that library is still in there. It's meant, now, it's focused on object recognition. But it has a whole bunch of statistical routines, and lately, it's added from Marius Muja at David Lowe's group, which it's added FLANN, which is a fast approximate nearest neighbor, so we use that quite a bit now.

Q: And how is that all being used within ROS?

Gary Bradsky: So, I mean, OpenCV stands alone, but it also is wrapped very lightly in ROS, as all things are. So, if you're going to call in and build the camera modules or the things that depend on calibration or anything like that, the CMG file will call in and build OpenCV, and so you'll just have it, seamless to you. You know, if you need it, it will be built and pulled in. And so, it's the people who are using the perception parts will end up with OpenCV.

Q: So, let's jump back to Stanley, and what were some of the challenges for the vision system in being a leader of that project?

Gary Bradsky: So, I mean, the main challenge is this is a car. It's going to run in the desert for an hour, seven hours, or something like that. And so, when you look at the failure rates, what can they be? It's really like 0. <laughs> So, we tried. I mean, we started meeting at night and

hashing out a bunch of approaches. And we tried a lot of stuff, and so, you know, I did these convolutions with line finders and then, you know, roads. Dirt roads tend to have lines, and paved roads with lines. So, you'd find these converging lines and you'd have some idea of the horizon and so we could identify like these things seem to go to horizon points. And so, we could identify roads. And we did many things, like actually just doing machine learning, you know, what is a road? And those things kind of worked in, you know, sometimes in high 90s. And actually, one of the machine-learning techniques beat, as far as we could tell, what we actually used. But the thing is, you know, Sebastian is very pragmatic about what to do, and so, the vision system on Stanley, there was a laser system, and the lasers had enough range and enough range pointing out sort of straight. You know, as you glance at the road, at some point, the lasers do you no good. They're too flat to the road, so every little bump seems like you're changing, you know, many meters. So, the vision system was really designed to extend how fast the car could go and still see. So, the lasers could go to about 25 miles an hour, but to do this race, you know, we realized that you – and the flat sections, you have to go faster. So, it was really – the vision system really came and clicked on when it could see enough road and say, "Look, you can really go faster. Then you can stop with the lasers, because I see it clear for a long ways." And so, it allowed the Stanley to speed up and go fast when it could go fast. And the tight-turning mountain roads, it wasn't really – the vision system was on, but it wasn't able to say go, you know?

So, when you look at the race, the vision system was on a fairly small percent of the time, which is true in any race. Whenever you have a race and you have the fast parts, you don't spend much time in the fast parts, because you go fast. <laughs> Right? You spend most of your time in the slow parts. So, you know, we backtracked that out, and it was absolutely critical for Stanley to win. I think it would have come in third place without doing those speed-ups. And so, that was the role of the vision system, is, one, to see further than the lasers and, you know, these were both merged into a bird's-eye view of the laser map, and then the vision system. What we ended up doing after thrashing around – and before, these two groups were separate and then they were put on the car. Now we actually had the laser data available to us. And so, we decided like the lasers would say, well, "This is flat." You know, the laser thinks the car can drive on this. And so, we just took that chunk and learned a color module and extended that out. And so, it's sort of like the laser would say, "I like this," and the vision system's job was to find more of this. And that's how it worked. And in the end, we had just a very simple color model. Our system got simpler and simpler. And we had this machine learning model that was kind of a regression system. That actually did better. But I think – well, partly – I don't know if it was the intersection or the code freeze, but I also think Sebastian thought, you know, like this color system was kind of – came down to a decision tree, and what does it mean that like the color is one, you know, is above 127 or below it? It's hard to tell, whereas we did a Gaussian color model, and you could make some reasoning about that, that I am this far away from the color distribution, so I'm going to trust it less and less. And so, yeah, he wanted to go with a simpler system, and that, you know, because he figures we can always slow the car down, you know, but we want to be really sure when we can go fast that we're not going to run into

anything. So, that's how it worked. But, and, you know, so, essentially the vision system didn't fail, you know, seven hours running full-time, even though the laser system did fail, I think, three times in the race, because of some timing problem where things got out of order. It would think there was a wall in front of it.

Q: What was Sebastian like to work with?

Gary Bradsky: He was great. <laughs> I mean, he was a lot of fun. It was actually, you know, I guess there was this competition between Red and Sebastian, and I like Sebastian's style more. And it was kind of two styles, and I thought like – so, Red would agree, it was he's more command and control. He's like he's an ex-Marine, he tells you what to do, you do it. And that works often, if the person is smart enough or whatever. It doesn't work if the person is not smart. But in this case, both leaders were smart enough, right? And so, that works. But Sebastian was more like, you know, just keep up with me. He's not directive in that sense. It's just like, you know, well, you know, you're not keeping the pace. I'm going fast. You keep up. And so, I wanted – part of my joining the team was like I want to just establish if that approach also works, <laughs> right? And so, in this case, it did, you know, that this more open style of leadership. Yeah, not that there weren't occasions of falling out and screaming. <laughs> But it all, I mean, that team, when it was coming together, you just had this feeling that we're going to win. I mean, it's just like it clicked in many ways. So, you know, Mike Montemerlo, he was sort of, you know, the guy who was the type-A personality holding everything together. But, I mean, all the personalities, this Hendrik and whatever, we had a good mesh, and you just felt like this team is just hitting on all cylinders. So.

Q: Any funny stories that you want to share?

Gary Bradsky: There's many, but I don't know. I guess the funniest one is that I'm actually friends with this reporter, John Markoff, and so, I was saying, "Cover this, cover this," <laughs> whatever, so he came in and he took a ride in the robot and that was the one time it crashed.

<laughter>

Gary Bradsky: And I think what it was was an overreaction. I think it was. I might get the story wrong. But I think Mike Montemerlo wanted to be extra careful, and so he – I think he put a limit on the speed to below 15 miles an hour, but I think it ended up introducing a bug, because I guess they went down this berm and – I'm not sure exactly what the sequence of things happened, but I think it slowed down to go down and then saw that it's not 15 miles an hour and it sped up to go 15 miles an hour and then crashed down into some bushes or whatever. But no one got hurt. <laughs> The car didn't get hurt, but – I don't know. There were other funny

things, like I think the CMU team had professional drivers because they might have asked their lawyers, like can a student be in a robot car? But Sebastian is the kind of guy, he's not going to ask that question, because you know what the answer is going to be. A lawyer in the school will say, "No." Like, what's the lawyer's benefit in saying yes? <laughs> Only a disaster could follow. So, but it was never asked, and I think it was important that the students were in the car, because, you know, professional drivers, some glitch happened, the guy – just ignore it after a while. "Oh, well, the car straightened out." But the guy who is coding that will say, "What?" You know, like why did that glitch, and go look into it. But, so, you know, as far as debugging, I think that was important. But, I don't know, the whole project was a lot of fun.

Q: And how did you get involved with Willow Garage and decide to make that career move?

Gary Bradsky: So, in a weird way, I had that – it sort of was an incentive, I don't know at what level, too, forming Willow Garage, but not in the way – so, at Intel, I had a DARPA subcontract with cognitive architectures, and I thought, oh, that's a crazy program. Like, I don't even know what a cognitive architecture is. I don't think anyone really does, <laughs> so, but, and then, you know, I saw that the program manager was going – so, this was an Intel subcontract. But I saw that the program manager was going to leave and was retiring or whatever. They only keep them for a certain time. It's six months. And I go, there's no way this program is going to survive the six months. The new guy is going to come in. No new guy says, "Oh, I'll take the old guy's stuff and run with it." And so I go, this is only going to last six months. And so, what I did is I had money to hire some students to work on cognitive architecture. Instead, I bought a robot arm, <laughs> which caused some consternation between the lawyers at USC who we were subcontracting and Intel. But I smoothed it over somehow. And so, I had this robot arm, and one of the things on the robot arm is it has to be transferred off as government property to some other DARPA-contractor government entity, and I go, well, Stanford is full of DARPA contracts. So, I went to Andrew Ng, and I said, "Well, here's an arm," and Stanford had all this early history and it was shaky and whatever and there ought to be some learning-based, perception-based robotics. And my idea was, let's put some cameras and do it with the arm and start working a project. But Andrew took, I guess, that as a kick and expanded it into the STAIR program, Stanford AI Robot, that's still going on. All right, and I guess Ken Salisbury – I don't really know the causality chain here. I went off. Soon after that, I quit and went to a start-up. I quit Intel. So, the contract ended, I transferred the equipment. That arm is still there, <laughs> in Andrew Ng's lab, and it's a Harmonic Arm from Canada.

And so, we started this whole project putting wheels under it and making it basically a learning robotics project and got a bunch of the department involved. And I guess that triggered Ken Salisbury and some of his students, Eric and Keenan, who were originally here, to start building a robot torso. And I, while I was at Stanford, I was hearing about, "Oh, there's a robot torso." And I go, "Oh, wouldn't that be cool to see?" You know, like instead of this, STAIR was like sort of bolted-together little pieces. And then I went off to the start-up, and I guess during

that time I was away doing this start-up – which is Video Surf, it's still there, video indexing – I guess they built up PR1, and then Scott, our founder, came in and said, "Well, that will be one of the robotics projects," and brought those two in that formed the nucleus here. Now, later on, after a year of doing the start-up, I decided it's not the right place for me, because I wanted to keep working on OpenCV and other things. And so, I was looking around. And then Andrew Ng called me up, and I was starting to interview. I interviewed with Facebook and all these places. And I was going to get ready to take a job, and Andrew Ng said, "You know, I really think you should see this new place." And so, I said like, "I can't. Like, they're getting onto me. It's like a 15-person start-up." You know, it was pretty obvious. I was spending a lot of late lunches and whatever while interviewing. And he said, "No, I think you should go."

So, I looked at the map and I said, "Well, I could get there in 15 minutes, have a sandwich, shake some hands, and get back, and within a half an hour, they'll never know." And so, I came here and I think I stayed for like 3 hours and then I just – you know, I got to work here. So. Since then, you know, they have supported OpenCV, and now we're working on the – so, we had a lot of perception packages. Now we're putting it into a framework that I think will be pretty efficient. So, you know, we're looking for a way where it's very fast to code object recognition and will also keep the code in shape and maintainable. So, we want fast development speed, but we also want that the code itself is fast. And so, we have kind of a Python framework that wraps modules which wrap OpenCV. And basically, you know, when you're done, you build up all these modules and you can just put them together, mix and match. So, I'm doing a color-based object recognizer, but once it's done, we have this texture-based object recognizer, and it will be simple just to say, "Put them together." So, you know, that's what we're doing here, sort of the perception side of robotics. Or, that's what I'm doing.

Q: What are some of the real challenges for object recognition?

Gary Bradsky: Just that it's an unsolved problem. <laughs> I mean, yeah, I mean, it's actually kind of an insane – I've always worried about being in computer vision. Like, you know, I mean, a field that doesn't really work and I get these – encourage these students and they go, but they all stay employed. So, I mean, it really is making progress, but it's daunting in many ways, because humans have this sense of vision. They think it's easy. But when you look at a camera, it's just a flux of numbers, and it really is this flux of numbers, because it's the world is filled with noise and, you know, how can I segment me from the background and, you know, how do you be discriminate enough. So, if you have something that is very tight-matched, well, you're not going to match anything, because you never, ever see the same view twice, right? You just don't do it unless you're some very precise industrial machine. So, you need flexibility in the templates. Well, that means they're sloppier, and that means they get confused. And so, you really have to put on a lot of modalities, and so some computer-vision techniques that do well are extremely slow, and it's like, well, when will they be fast? Sort of like never. <laughs> And so,

we're looking more at, you know, things that are real-time rather than just trying to get out academic papers that do good jobs in recognition.

So, I also started this contest that's called Solutions in Perception, and we ran the first one in April, a month or two ago, and this is to establish what problems are solved in vision. We start with easy objects and work our way up. Unfortunately, no one got about threshold, <laughs> but I think – so, we started with textured objects that are things you might find in a grocery store, like a box of tea or a cereal box or detergent bottle, not things that were transparent or shiny, and so basically things with a lot of texture on it, texture could be anything, manufactured pictures or actually text. And so, I think that is solvable, and it was just that the contest didn't have enough time. It was sort of a rush. We were doing it with NIST, and the White House Office of Technology liked it. Thomas Kalil was spurring it on. And so, it was a bit of a rush. We did it in four months. So, this time, we'll have a year, and so, I hope we'll solve some categories of vision. We're actually going to loosen up the problem a little bit to be textured and textureless, rigid, non-shiny, non-transparent objects, so lots of the stuff around here, the tape dispenser over there, that would be part of this. We also were a little bit limited by the Kinect. So, the Kinect is a great device. It's really a price breakthrough, revolutionary in that sense. <laughs> But it's got a really lousy camera, because it was designed, not for robots, but for, you know, consumers in a living room, and they don't really care, you know, that the camera is full of noise and whatever. It's a really cheap camera. So, now, you know, we have calibration to put a better camera with the Kinect. Ultimately, we want a Kinect device for robots that will, you know, have higher resolution. So, because it's so low-resolution, that hurt the texture recognition. We decided to standardize around this Kinect device. And it's a struggle, because you have to turn it on. It's 1 megapixel mode, which is sort of hidden in it if there is such a mode. And it's really, really noisy. So, this texture, they're trying to find gradients, and yet, half the image is just fictitious gradients of noise. So, that was a big challenge. And so, this next year, we're going to lift the – say, "Bring whatever you want, just whatever sensor you want, whatever computer you want. Just get the job done." So, that's one of the learnings for next year.

Q: Do you focus those mostly on the static images or, I mean, robots have the ability to move through the world and use optic flow.

Gary Bradsky: And so, in this case, it was multiple views. It was also – you know, we made a lot of mistakes. We were going to be the masters serving the data and checking the codes so that other people – my fear was things like the PASCAL Challenge, it's self-reported. It's not that I think people lie, but I think there is the possibility that you run it once and you go, "Oh, I forgot to set the threshold," you know, and then you set it. Well, now – you can't really run things twice and get an honest score. So, I was afraid of that. But now I'm going to say, well, I've lost my fear of that. <laughs> It's too much work, and also, it's too constraining on the people. So, we're just going to put out here's the kind of objects that we're going to do. And at the

conference where this is going to run, we're just going to bring in a pile of these objects, of new objects but of the sorts that people saw, and they're going to have to learn them there and recognize them there. They'll have two days to train and test and then the third day will be a contest. And they'll be able to move around. So, we think – we didn't – right now, you know, robots are still expensive and whatever, and so, we're just going to have however you want to transport your camera around. It can be on a robot or you can just move it by hand. But the idea is you will have the ability. So, unlike this last time we were asking per view, we're going to set up stations where you'll have a certain fixed amount of time to run around the room and recognize objects, and at these stations, you can do whatever you want. So, we just want the report at the end, what objects are there. So, one of the problems we had while making these fixtured scenes was things was occlude. Well, now, how do you code that occlusion? And sometimes it's half an occlusion. Is it fair to ask them to recognize the back objects? I mean, these were all such judgment calls that we didn't like. So, now we're just going to say a robot could move around, but the human might actually be moving the camera, but it'll just be – do whatever it takes to recognize this.

And the idea for this contest is, once we establish like plates and cups, let's say, are solved, then this – then people can rely on, you know, the planners and can just say, "Oh, that part is solved." We can go on and think of more complex things. I got that from like the DARPA Grand Challenge, which I think, you know, people thought of doing that, because things like navigation – I mean, GPS was mature and SLAM was getting mature, and so that allowed people to think, well, we could actually run for hours in a desert. Before then, I mean, it would have been ridiculous. I mean, the early robots could barely get across a room. Now I can hear robots wander around all the time. They don't bump into things or people. And so, that allows you to think, well, you know, I can think of them now operating around people. So, I want with objects to say like these classes of objects are solved, so now we can move on. You can say, well, can I clear a table? Yeah, if I can get cups and dishes and whatever, that would be something someone would think of. Having a Grand Challenge or maybe building, you know, a robot that can clean a cafeteria or something like that. But I think it will be unleashed when the perception is unleashed. You know, that's one of the gating functions. Yeah, there are many other gates on using the arm and planning and stacking, you know, but we're going to see what we can do.

Q: How do you interface with other parts of the team?

Gary Bradsky: We argue. No. <laughs> So, I mean, more or less, there are the manipulation people, and planners are consumers of the perception. So, we've had a long-running dev-mo that sort of gets some bowls and whatever from the early stuff I did with Marius Muja. And now we're trying to replace it with more sophisticated stuff, you know, that – and so, their sort of our customers or whatever, you know. Our user base is the people who are going to use it for manipulation. You know, we have a small team. So, I run this OpenCV team that Willow

supports, and those are five Russian developers. But they are mostly responsible for OpenCV, not the object-recognition in particular. I mean, there is overlap. They did help on the plugging in and many other things. But mostly, they are on a library maintenance and debugging and advancing certain techniques. So, what we have here in object recognition, the real core team is me and two others, so Vincent Rabaud and Ethan Rublee are working on this infrastructure. And then, there's a programmer here, Troy, who is helping us with trying to get this seamless infrastructure of these modules and things that are easy to string together. I was programming on it just before the interview. <laughs>

Q: And you mentioned, I mean, you mentioned object recognition, you mentioned the plugging in. What are some other, you know, applications of PR2 that you have worked on?

Gary Bradsky: Well, I mean, there was these milestones, right? So, there were a couple phases of plugging in for the first mile – well, milestone two, that was the big one. We didn't actually have the cam – well, the camera in the forearm just came, but way too late for the effort. So, we stuck with the high-resolution camera. But later on, we did another effort where we moved it down. Now it uses the forearm camera. And doorknob recognition. But the perception has been used in absolutely every video. You can look on YouTube. There is the beer-getting. Well, it was using the stuff to find the beer. There is the pool. Where it was using, you know, finding balls and whatever, you know, using OpenCV functions and whatever. And on and on. We've had picking up demos and clearing demos and, you know, they've used this on and on. The problem has been that this object recognition has been kind of fragmented. Like – I teach a class at Stanford which is Perception for Robotics, and, well, the first year, we called it Perception for Manipulation, and, you know, I wrote a module that was pretty good at recognizing hands and things. And it just wasn't maintained. So, that's why we're building this infrastructure as a kind of polished piece of software engineering where this stuff, once it's locked in, it stays usable and stays alive.

So, you know, that's what we're aiming at now. And that, hopefully, will become more consumable by other people as – I don't know if a perception person – I mean a manipulation person will actually use this module. Sophisticated people like Mattei here might, but they would have the ability to actually construct their own object recognition by putting, oh, I know how to filter the background. That's just this. What we're building you can actually do graphically or in Python, but when you're done, it's all C++, because these are just very like Python wrappers over C++ code. But they can string this stuff together, and then they have a stand-alone C++ module, you know, that's for speed that you can run as a Python script. But you can run it stand-alone. And then, we hope to actually make it even easier for ROS developers. At the end we can say, "Well, once you're done designing this module, it can either stand alone or it can – you can just say I want this as a ROS node, and it will make a message type for it automatically, or a ROS service." So, that's the goal.

Q: Have you had feedback from people who've been using it with ROS?

Gary Bradsky: Well, I mean, no, this framework is new, so the feedback is all the people – is internal users. But from – well, I have plenty of feedback on OpenCV and, you know, feedback from the other systems, as I think they want a more modular system that's more of a package. You know, like we are researchers, so we're moving ahead, and we're not the software engineers. So, there hasn't – we haven't put enough effort into released code. That's this released object module. And so, I think that would probably be the complaint. The answer to that complaint is the system, because every module is very simple and will have test code and be essentially released. And so, on a new system, you'll just string together modules, and then there will be the core one you're working on. Well, you write test code for that. And the documentation is forced, because we have these – getting into detail – virtual functions you have to fill out, and one of them is a documentation function. So, the documentation is just another method in the class. As a result, the Python can say, "Document yourself." You can put all this stuff together, and you'll have this whole flow of what this thing actually does. And so, that's another thing. Like, and it spits this out in SWG. And, you know, so, it's a language that can be turned immediately to an HTML or a PDF document.

Q: So, I know you have to get to graduation. <laughs>

Gary Bradsky: Yes.

Q: So, we should probably wrap up. Do you have anything else you want to add other than our fabulous wrap-up questions? Okay, so, <laughs> so, we just ask everyone this to wrap up, but do you have any advice for young people who would be interested in robotics or computer vision?

Gary Bradsky: Do your math homework. <laughs> I mean, you know, it's not always so mathematical as people think. I mean, some of the things are extremely mathematical, but I'm not sure if they have as much payoff as just simple intuitions and sort of persistence. So, I think, you know, there is a big role for persistence and creativity. There is, you know, getting into robotics and computer vision, there is always this thing, like you sort of want to know what you're doing, and by that, I mean, is how do you know if something is working or not? And that's sort of a talent or skill that you have to build up. A lot of times, you have these complex systems that have lots of tradeoffs and you can't test everything. So, you have to develop a lot of intuitions. I'm building a color-based detector. Well, there's a million things I can do. I'm using color ratios, because that's the only thing that's more or less stable, but it's not so stable, so how am I going to counter that out? And the thing, you know, I run an early version. It kind of works. But it has a bug in it. How do you know that it has a bug in it? Because a lot of the time when you're doing pattern recognition, things will work, but not as well as it should. And so,

how do you tell, you know, that? You have to think of ways of testing that what you're seeing is really what you thought you wanted to do. And so, I don't know, creative play. I'm all for that, you know. I think you should just really go for what you're passionate about, find – if you find perception interesting, then just start doing it, hacking it. You know, there's a lot of code out there. I am now just relearning hardware design. I used to work in it but long since forgotten it. But now there's all this open-source, Arduino and, you know, you can just get in there and start designing chips. And now there's like EAGLE. You can lay out circuit boards. It's getting pretty easy, right? So, I mean, there's all these tools. I mean, that's the good thing with computer vision, is just start learning the tools and it gets you a long ways there. You know, I guess one of the easier ways is to pick a MATLAB and that's, you know, easy to develop in and, you know, start adding maybe Python and C and C++.

Q: How do you think the availability of these kind of more open, both hardware systems and also software systems, is affecting the development of the field?

Gary Bradsky: I mean, it's training up this new generation of, you know, people maybe we thought we had lost, these old like designers and hardware, because, I mean, people are just doing wild stuff. You know, you can find almost anything you think of. Like, I'm looking at spectrum analyzing. Well, people built this little chip into a spectrum analyzer, not in the way I really want to do it, but, I mean, it's getting easier to start combining these Arduino chips and all this stuff. It's getting easier to start doing hardware and software again. I mean, it's sort of like building up these packages. And so, now you can really start doing hobby robotics. You can make an arm pretty cheap with these little servos and controllers. And, you know, maybe that's what it takes. I have heard of, you know, very clever things that people are doing. So, you want force control and touch – you know, how are you going to do that? Well, people have taken off the touch screens on like the iPhones or whatever and used those. Like sliding together, you can detect movement and force and contact and, you know. So, I mean, a lot of creative stuff happening out there.

Q: Great.

Q: Thank you very much.

Q: Thank you very much.

Gary Bradsky: Yeah, thank you.